

LNCS 6845

Leslie Ann Goldberg
R. Ravi José D.P. Rolim

Approximate Randomization and Combinatorial Optimization

Algorithms and Techniques

14th International Workshop
and 15th International Workshop
Princeton, NJ, USA, August 2011

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Alfred Kobsa

University of California, Irvine, CA, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

TU Dortmund University, Germany

Madhu Sudan

Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max Planck Institute for Informatics, Saarbruecken, Germany

Leslie Ann Goldberg Klaus Jansen
R. Ravi José D.P. Rolim (Eds.)

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

 Springer

Volume Editors

Leslie Ann Goldberg
University of Liverpool
Department of Computer Science
Ashton Building, Liverpool L69 3BX, UK
E-mail: l.a.goldberg@liverpool.ac.uk

Klaus Jansen
University of Kiel
Department of Computer Science
Olshausenstr. 40, 24098 Kiel, Germany
E-mail: kj@informatik.uni-kiel.de

R. Ravi
Carnegie Mellon University
Tepper School of Business
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
E-mail: ravi@cmu.edu

José D.P. Rolim
University of Geneva
Centre Universitaire d'Informatique
Battelle A, 7 route de Drize, 1227 Carouge, Switzerland
E-mail: jose.rolim@unige.ch

ISSN 0302-9743
ISBN 978-3-642-22934-3
DOI 10.1007/978-3-642-22935-0
Springer Heidelberg Dordrecht London New York

e-ISSN 1611-3349
e-ISBN 978-3-642-22935-0

Library of Congress Control Number: 2011933808

CR Subject Classification (1998): F.2, E.1, G.2, I.3.5, F.1, C.2

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

© Springer-Verlag Berlin Heidelberg 2011

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.

The use of general descriptive names, registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

This volume contains the papers presented at the 14th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX 2011) and the 15th International Workshop on Randomization and Computation (RANDOM 2011), which took place concurrently in Princeton University, USA, during August 17–19, 2011.

APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally difficult problems, and was the 14th 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), and Barcelona (2010). RANDOM is concerned with applications of randomness to computational and combinatorial problems, and was the 15th 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), and Barcelona (2010).

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, mathematical programming methods, combinatorial problems in graphs and networks, game theory, markets and economic applications, geometric problems, packing, covering, scheduling, approximate learning, design and analysis of online algorithms, design and analysis of randomized algorithms, randomized complexity theory, pseudorandomness and derandomization, random combinatorial structures, random walks/Markov chains, expander graphs and randomness extractors, probabilistic proof systems, random projections and embeddings, error-correcting codes, average-case analysis, property testing, computational learning theory, and other applications of approximation and randomness.

The volume contains 29 contributed papers, selected by the APPROX Program Committee out of 66 submissions, and 29 contributed papers, selected by the RANDOM Program Committee out of 64 submissions.

We would like to thank all of the authors who submitted papers, the two invited speakers, David P. Williamson and Joel Spencer, the members of the Program Committees, and the external reviewers.

We gratefully acknowledge the support from the Department of Computer Science at the University of Liverpool in the UK, the Tepper School of Business

at Carnegie Mellon, USA, the Institute of Computer Science of the Christian-Albrechts-Universität zu Kiel, and the Department of Computer Science of the University of Geneva.

Finally, many thanks to Parvaneh Karimi-Massouleh for editing the proceedings.

August 2011

Leslie Goldberg
R. Ravi
Klaus Jansen
José D.P. Rolim

Organization

Program Committees

APPROX 2011

Julia Chuzoy	Massachusetts Institute of Technology, USA
Naveen Garg	Indian Institute of Technology, India
Michel Goemans	Massachusetts Institute of Technology, USA
Fabrizio Grandoni	Università di Roma Tor Vergata, Italy
Anupam Gupta	Carnegie Mellon University, USA
Prahalad Harsha	Tata Institute of Fundamental Research, India
Satoru Iwata	Kyoto University, Japan
Philip Klein	Brown University, USA
Robert Krauthgammer	Weizmann Institute of Science, Israel
Kamesh Munagala	The University of Pennsylvania, USA
Zeev Nutov	The Open University of Israel
R. Ravi	Carnegie Mellon University (Chair), USA
Guido Schaefer	Centrum Wiskunde & Informatica, The Netherlands
Chaitanya Swamy	University of Waterloo, Canada
Kunal Talwar	Microsoft Research Silicon Valley, USA
Gerhard Woeginger	Eindhoven University of Technology, The Netherlands

RANDOM 2011

Per Austrin	University of Toronto, Canada
Nayantara Bhatnagar	University of California, USA
Amin Coja-Oghlan	University of Warwick, UK
Josep Diaz	Universitat Politecnica de Catalunya, Spain
Benjamin Doerr	Max-Planck-Institut für Informatik, Germany
Devdatt Dubhas	Chalmers University of Technology, Sweden
Martin Dyer	School of Computing, UK
Tom Friedetzky	Technische Universität München, Germany
Leslie Goldberg	Carnegie Mellon University (Chair), USA
Mark Jerrum	University of Edinburgh, UK
Elitza Maneva	Universitat de Barcelona, Spain
Allan Sly	Microsoft Research, Theory Group, USA
Eli Upfal	Brown University, USA
Juan Vera	Carnegie Mellon University, USA
Osamu Watanabe	Tokyo Institute of Technology, Japan
David Zuckerman	University of Texas at Austin, USA

Referees

Anil Ada	Sanjoy Dasgupta	Ito Hiro
Saeed Alaei	Amit Deshpande	Kazuo Iwama
Aris Anagnostopoulos	Martin Dietzfelbinger	Rahul Jain
Alex Andoni	Jian Ding	Hossein Jowhari
Alexandr Andoni	Michael Dinitz	Valentine Kabanets
Anon Anon	Irit Dinur	Satyen Kale
V. Arvind	Shaddin Dughmi	Lior Kamma
Nikhil Bansal	Giuseppe Durisi	Daniel Kane
Surender Baswana	Zeev Dvir	George Karakostas
M. Bateni	Alon Efrat	Jonathan Katz
Tugkan Batu	Klim Efremenko	Akinori Kawachi
Siaavosh Benabbas	Charilaos Efthymiou	Mark Keil
Petra Berenbrink	Khaled Elbassioni	Iordanis Kerenidis
Shankar Bhamidi	Michael Elkin	Guy Kindler
Sayan Bhattacharya	Yuval Emek	Robert Kleinberg
Arnab Bhattacharyya	David Eppstein	Adam Klivans
Eric Blais	Funda Ergun	Yusuke Kobayashi
Anna Blasiak	Uriel Feige	Ronald Koch
Andrej Bogdanov	Anne Fey	Jochen Koemann
Glencora Borradaile	Yuval Filmus	Alexandra Kolla
Andreas Brandstadt	Eldar Fischer	Stavros Kolliopoulos
Mark Braverman	Ehud Friedgut	Swastik Kopparty
Joshua Brody	Tobias Friedrich	Guy Kortsarz
Harry Buhrman	Alan Frieze	Nitish Korula
Gruia Calinescu	Takuro Fukunaga	R. Krishnaswamy
Amit Chakrabarti	Peter Gacs	Michael Krivelevich
Deeparnab Chakraborty	Anna Gal	Janardhan Kulkarni
Sourav Chakraborty	Dmitry Gavinsky	Oded Lachish
Parinya Chalermsook	Konstantinos Georgiou	Michael Langberg
Timoth M. Chan	Ashish Goel	Gilbert Laporte
Arkadev Chattopadhyay	Oded Goldreich	Silvio Lattanzi
Ning Chen	Parikshit Gopalan	James Lee
Mahdi Cheraghchi	Lee-Ad Gottlieb	Asaf Levin
Flavio Chierichetti	Sudipto Guha	Shi Li
Eden Chlamtac	Sylvain Guillemot	Yi Li
Kai-Min Chung	Venkatesan Guruswami	Shachar Lovett
Andrew Collins	M. Hajiaghayi	Brendan Lucier
Matthew Cook	Moritz Hardt	Frederic Magniez
Graham Cormode	Prahladh Harsha	Mohammad Mahmoody
Mary Cryan	Johan Hastad	Rajsekar Manokaran
Marek Cygan	Elad Hazan	Russell Martin
Artur Czumaj	Brett Hemenway	Jiri Matousek
Peter Damaschke	Martin Hildebrand	Arie Matsliah

Kevin Matulef	Rajiv Raman	Zoya Svitkina
Andrew Mcgregor	Oded Regev	Suguru Tamaki
Colin Mcquillan	Daniel Reichman	Marc Thurley
Manor Mendel	Heiko Röglin	Hidetoki Tanaka
Julian Mestre	Dana Ron	Siamak Tazari
F. Meyer Auf Der Heide	Bill Rosgen	Orestis Telelis
Daniele Micciancio	Benjamin Rossman	Prasad Tetali
Peter Bro Miltersen	Aaron Roth	Fabio Toninelli
Vahab Mirrokni	Thomas Rothvoss	Madhur Tulsiani
Dieter Mitsche	Yogish Sabharwal	Paul Valiant
Matthias Mnich	Barna Saha	Fabio Vandin
Cris Moore	Mohammad Salavatipour	Anke Van Zuylen
Marco Molinaro	Rahul Santhanam	Santosh Vempala
Dana Moshkovitz	Mathias Schacht	Rakesh Venkat
Elchanan Mossel	Arnab Sen	Michael Viderman
Amir Nayyeri	Sandeep Sen	A. Vijayaraghavan
Alantha Newman	Maria Serna	Emanuele Viola
Evdokia Nikolova	Ronen Shaltiel	Berthold Voecking
Ryan O'Donnell	Asaf Shapira	Jan Vondrak
Neil Olver	Maiko Shigeno	Van Vu
Shayan Oveis Gharan	Igor Shinkar	Anil Vulikanti
Igor Pak	Amir Shpilka	Mark Walters
K. Panagiotou	Mohit Singh	Cenny Wenner
Gopal Pandurangan	Christian Sohler	Thomas Vidick
Chris Peikert	Jose Soto	Andreas Wiese
Ludovic Perret	Perla Sousi	David Wilson
Alberto Pettarin	Daniel Spielman	Carola Winzen
Ulrich Pferschy	Joachim Spoerhase	David Woodruff
Olli Pottonen	Srikanth Srinivasan	Yi Wu
Xavier Pérez Giménez	Nikhil Srivastava	Orly Yahalom
Yuri Rabinovich	Gautier Stauffer	Sergey Yekhanin
J. Radhakrishnan	Jeff Steif	Yuichi Yoshida
Prasad Raghavendra	He Sun	Rico Zenklusen
Sanatan Rai	Ravi Sundaram	An Zhu
David Richerby	Ola Svensson	Anastasios Zouzias
Matteo Riondato	Maxim Sviridenko	

Table of Contents

Contributed Talks of APPROX

New Tools for Graph Coloring	1
<i>Sanjeev Arora and Rong Ge</i>	
Inapproximability of NP-Complete Variants of Nash Equilibrium	13
<i>Per Austrin, Mark Braverman, and Eden Chlamtác</i>	
Sparse Recovery with Partial Support Knowledge	26
<i>Khanh Do Ba and Piotr Indyk</i>	
On Capacitated Set Cover Problems	38
<i>Nikhil Bansal, Ravishankar Krishnaswamy, and Barna Saha</i>	
Bandwidth and Low Dimensional Embedding	50
<i>Yair Bartal, Douglas E. Carroll, Adam Meyerson, and Ofer Neiman</i>	
$O(1)$ -Approximations for Maximum Movement Problems	62
<i>Piotr Berman, Erik D. Demaine, and Morteza Zadimoghaddam</i>	
Optimal Lower Bounds for Universal and Differentially Private Steiner Trees and TSPs	75
<i>Anand Bhalgat, Deeparnab Chakrabarty, and Sanjeev Khanna</i>	
Social Welfare in One-Sided Matching Markets without Money	87
<i>Anand Bhalgat, Deeparnab Chakrabarty, and Sanjeev Khanna</i>	
Primal-Dual Schema and Lagrangian Relaxation for the k -Location- Routing Problem	99
<i>Tim Carnes and David B. Shmoys</i>	
Scheduling Resources for Throughput Maximization	111
<i>Venkatesan T. Chakaravarthy, Amit Kumar, Vinayaka Pandit, Sambuddha Roy, and Yogish Sabharwal</i>	
Coloring and Maximum Independent Set of Rectangles	123
<i>Parinya Chalermsook</i>	
A Primal-Dual Approximation Algorithm for Min-Sum Single-Machine Scheduling Problems	135
<i>Maurice Cheung and David B. Shmoys</i>	
A $(1 + \ln 2)$ -Approximation Algorithm for Minimum-Cost 2-Edge-Connectivity Augmentation of Trees with Constant Radius	147
<i>Nachshon Cohen and Zeev Nutov</i>	

Periodicity and Cyclic Shifts via Linear Sketches	158
<i>Michael S. Crouch and Andrew McGregor</i>	
An Approximation Algorithm for the Tree t -Spanner Problem on Unweighted Graphs via Generalized Chordal Graphs	171
<i>Feodor F. Dragan and Ekkehard Köhler</i>	
Approximating the Closest Vector Problem Using an Approximate Shortest Vector Oracle	184
<i>Chandan Dubey and Thomas Holenstein</i>	
Opaque Sets	194
<i>Adrian Dumitrescu, Minghui Jiang, and János Pach</i>	
Exploring and Triangulating a Region by a Swarm of Robots	206
<i>Sándor P. Fekete, Tom Kamphans, Alexander Kröller, Joseph S.B. Mitchell, and Christiane Schmidt</i>	
Improved Competitive Ratios for Submodular Secretary Problems (Extended Abstract)	218
<i>Moran Feldman, Joseph (Seffi) Naor, and Roy Schwartz</i>	
Locating Depots for Capacitated Vehicle Routing	230
<i>Inge Li Gørtz and Viswanath Nagarajan</i>	
Satisfying Degree- d Equations over $GF[2]^n$	242
<i>Johan Håstad</i>	
Black-Box Reductions in Mechanism Design	254
<i>Zhiyi Huang, Lei Wang, and Yuan Zhou</i>	
Multiplicative Approximations of Random Walk Transition Probabilities	266
<i>Michael Kapralov and Rina Panigrahy</i>	
Approximation Schemes for the Betweenness Problem in Tournaments and Related Ranking Problems	277
<i>Marek Karpinski and Warren Schudy</i>	
Network-Design with Degree Constraints	289
<i>Rohit Khandekar, Guy Kortsarz, and Zeev Nutov</i>	
Improved Approximation Algorithms for the Min-Max Tree Cover and Bounded Tree Cover Problems	302
<i>M. Reza Khani and Mohammad R. Salavatipour</i>	
Algorithmic Extensions of Cheeger’s Inequality to Higher Eigenvalues and Partitions	315
<i>Anand Louis, Prasad Raghavendra, Prasad Tetali, and Santosh Vempala</i>	

Nearly Optimal NP-Hardness of Vertex Cover on k -Uniform k -Partite Hypergraphs	327
<i>Sushant Sachdeva and Rishi Saket</i>	
A Linear Time Approximation Scheme for Maximum Quartet Consistency on Sparse Sampled Inputs	339
<i>Sagi Snir and Raphael Yuster</i>	
Contributed Talks of RANDOM	
Viral Processes by Random Walks on Random Regular Graphs	351
<i>Mohammed Abdullah, Colin Cooper, and Moez Draief</i>	
Quantum Property Testing for Bounded-Degree Graphs	365
<i>Andris Ambainis, Andrew M. Childs, and Yi-Kai Liu</i>	
Lower Bounds on the Query Complexity of Non-uniform and Adaptive Reductions Showing Hardness Amplification	377
<i>Sergei Artemenko and Ronen Shaltiel</i>	
Testing Graph Blow-Up	389
<i>Lidor Avigad and Oded Goldreich</i>	
On Sums of Locally Testable Affine Invariant Properties	400
<i>Eli Ben-Sasson, Elena Grigorescu, Ghid Maatouk, Amir Shpilka, and Madhu Sudan</i>	
Limits on the Rate of Locally Testable Affine-Invariant Codes	412
<i>Eli Ben-Sasson and Madhu Sudan</i>	
The Computational Complexity of Estimating MCMC Convergence Time	424
<i>Nayantara Bhatnagar, Andrej Bogdanov, and Elchanan Mossel</i>	
Streaming Algorithms with One-Sided Estimation	436
<i>Joshua Brody and David P. Woodruff</i>	
Everywhere-Tight Information Cost Tradeoffs for Augmented Index	448
<i>Amit Chakrabarti and Ranganath Kondapally</i>	
A Canonical Form for Testing Boolean Function Properties	460
<i>Dana Dachman-Soled and Rocco A. Servedio</i>	
Independent Sets in Random Graphs from the Weighted Second Moment Method	472
<i>Varsha Dani and Cristopher Moore</i>	
Extractors and Lower Bounds for Locally Samplable Sources	483
<i>Anindya De and Thomas Watson</i>	

A Deterministic Algorithm for the Frieze-Kannan Regularity Lemma . . .	495
<i>Domingos Dellamonica, Subrahmanyam Kalyanasundaram, Daniel Martin, Vojtěch Rödl, and Asaf Shapira</i>	
Dense Locally Testable Codes Cannot Have Constant Rate and Distance	507
<i>Irit Dinur and Tali Kaufman</i>	
Efficient Probabilistically Checkable Debates	519
<i>Andrew Drucker</i>	
An Efficient Partitioning Oracle for Bounded-Treewidth Graphs	530
<i>Alan Edelman, Avinatan Hassidim, Huy N. Nguyen, and Krzysztof Onak</i>	
Inflatable Graph Properties and Natural Property Tests	542
<i>Eldar Fischer and Eyal Rozenberg</i>	
Fast Simulation of Large-Scale Growth Models	555
<i>Tobias Friedrich and Lionel Levine</i>	
Improved Inapproximability Results for Counting Independent Sets in the Hard-Core Model	567
<i>Andreas Galanis, Qi Ge, Daniel Štefankovič, Eric Vigoda, and Linji Yang</i>	
Proximity Oblivious Testing and the Role of Invariances	579
<i>Oded Goldreich and Tali Kaufman</i>	
Optimal Rate List Decoding via Derivative Codes	593
<i>Venkatesan Guruswami and Carol Wang</i>	
Public Key Locally Decodable Codes with Short Keys	605
<i>Brett Hemenway, Rafail Ostrovsky, Martin J. Strauss, and Mary Wootters</i>	
On Sampling from Multivariate Distributions	616
<i>Zhiyi Huang and Sampath Kannan</i>	
Almost Optimal Explicit Johnson-Lindenstrauss Families	628
<i>Daniel Kane, Raghu Meka, and Jelani Nelson</i>	
Correlation Bounds for Poly-size AC^0 Circuits with $n^{1-o(1)}$ Symmetric Gates	640
<i>Shachar Lovett and Srikanth Srinivasan</i>	
Clustering in Interfering Binary Mixtures	652
<i>Sarah Miracle, Dana Randall, and Amanda Pascoe Streib</i>	

Approximating the Influence of Monotone Boolean Functions in $O(\sqrt{n})$ Query Complexity	664
<i>Dana Ron, Ronitt Rubinfeld, Muli Safra, and Omri Weinstein</i>	
On Approximating the Number of Relevant Variables in a Function	676
<i>Dana Ron and Gilad Tsur</i>	
Query Complexity in Errorless Hardness Amplification.....	688
<i>Thomas Watson</i>	
Author Index	701

New Tools for Graph Coloring^{*}

Sanjeev Arora and Rong Ge

Department of Computer Science, Princeton University
and Center for Computational Intractability
arora/rongge@cs.princeton.edu

Abstract. How to color 3 colorable graphs with few colors is a problem of longstanding interest. The best polynomial-time algorithm uses $n^{0.2072}$ colors. There are no indications that coloring using say $O(\log n)$ colors is hard. It has been suggested that SDP hierarchies could be used to design algorithms that use n^ϵ colors for arbitrarily small $\epsilon > 0$.

We explore this possibility in this paper and find some cause for optimism. While the case of general graphs is still open, we can analyse the Lasserre relaxation for two interesting families of graphs.

For graphs with low *threshold rank* (a class of graphs identified in the recent paper of Arora, Barak and Steurer on the unique games problem), Lasserre relaxations can be used to find an independent set of size $\Omega(n)$ (i.e., progress towards a coloring with $O(\log n)$ colors) in $n^{O(D)}$ time, where D is the threshold rank of the graph. This algorithm is inspired by recent work of Barak, Raghavendra, and Steurer on using Lasserre Hierarchy for unique games. The algorithm can also be used to show that known integrality gap instances for SDP relaxations like *strict vector chromatic number* cannot survive a few rounds of Lasserre lifting, which also seems reason for optimism.

For *distance transitive* graphs of diameter Δ , we can show how to color them using $O(\log n)$ colors in $n^{2^{O(\Delta)}}$ time. This family is interesting because the family of graphs of diameter $O(1/\epsilon)$ is easily seen to be *complete* for coloring with n^ϵ colors. The distance-transitive property implies that the graph “looks” the same in all neighborhoods.

The full version of this paper can be found at:

<http://www.cs.princeton.edu/~rongge/LasserreColoring.pdf> .

1 Introduction

In the graph coloring problem we are given a graph $G = (V, E)$. A *coloring with t colors* is a function $f : V \rightarrow [t]$, such that for any $(p, q) \in E$, $f(p) \neq f(q)$. The smallest t such that a coloring exists is called the *chromatic number* of the graph, and the graph is said to be *t -colorable*.

Despite much research we still have no good coloring algorithms even in very restricted cases. This is explained to some extent because it is NP-hard to approximate the chromatic number of a graph up to a factor of $n^{1-\epsilon}$ for any $\epsilon > 0$ ([20], following a long line of work in PCPs). Therefore attention has shifted to the case where the graph is $\tilde{3}$ -colorable. In this restricted case known algorithms can color the graph using $\tilde{O}(n^c)$ colors for some constants c . Wigderson’s

^{*} Research supported by NSF Grants CCF-0832797, 0830673, and 0528414.

¹ Here and throughout the paper \tilde{O} hides logarithmic factors.

purely combinatorial algorithm [19] works for $c = 1/2$. Using more combinatorial tools Blum achieved $c = 3/8$ [5]. Karger, Motwani, and Sudan [13] used SDP relaxations to achieve $c = 1/4$, which was combined with combinatorial tools to achieve $c = 3/14$ by Blum and Karger [6]. Arora, Charikar and Chlamtac [2] then carefully analyzed the SDP relaxation to reduce c to 0.2111. Chlamtac [8] further reduced c to 0.2072 using $O(1)$ levels of Lasserre lifting of the basic SDP relaxation (Lasserre lifting is defined in Section 2).

The seeming difficulty in getting even small improvements in c suggests that substantial improvement to c (achieving $c = o(1)$ for example) is intractable, but few lowerbounds are known. Dinur, Mossel and Regev [10] showed that it's hard to color with any constant number of colors (i.e., $O(1)$ colors) based on a variant of Unique Games Conjecture. Some integrality gap results [11,13,18] show that the simple SDP relaxation has an integrality gap at least $n^{0.157}$.

Arora et al. [2] suggested that using $O(1)$ or $O(\log n)$ levels of Lasserre lifting on the standard SDP relaxation should allow us to find an n^ϵ -coloring (running time would be $n^{O(k)}$ where k is the number of levels). In general researchers have hoped that a few rounds of lift-and-project strengthening of SDP relaxations (via Lasserre or other methods) should allow better algorithms for many other problems, though few successes have resulted in this endeavor.

The current paper is related to recent developments about the unique games problem. A surprising recent result of Arora et.al. [1] showed that unique games can be solved in subexponential time using the idea of *threshold rank*. More recently, Barak, Raghavendra and Steurer [3] showed that the surprising subexponential algorithm for unique games can be rederived using Lasserre lifting. Their rounding algorithm involves a new convex programming relaxation for threshold rank which we also use in a key way. It gives a way to round the SDP solution by showing that the solution vectors exhibit “global correlation.”

We extend the techniques of Barak et.al. to show that low threshold rank also helps in coloring 3-colorable graphs with fewer colors. Our algorithm is also derived using Lasserre liftings. In general we think our approach may lead to n^ϵ -coloring in subexponential or even quasi-polynomial time.

1.1 Our Results

The difficulty in using Lasserre liftings for colorings as well as any other problem is the lack of an obvious rounding algorithm. The paper [3] gives such a rounding algorithm for the unique games problem for graphs of low *threshold rank*. Our first result is a similar algorithm for graph coloring. We state the theorem here and will prove it in Section 4. The hypothesis uses a somewhat different notion of threshold rank than [3].

Theorem 1. *There is a constant $c > 1$ and a randomized rounding algorithm such that the following is true. If a regular 3-colorable graph G has threshold rank $\text{Rank}_{-1/16}(G)$ (i.e., the number of eigenvalues less than $-1/16$, where eigenvalues are scaled to lie in $[-1, 1]$) at most D , then the algorithm can find an independent set of size at least $n/12$ in time $n^{O(D)}$ with constant probability.*

Moreover, if the graph is vertex-transitive, there is a randomized algorithm that finds a coloring with $O(\log n)$ colors in $n^{O(D)}$ time.

As a corollary of the above result we can show that existing “counterexamples” for graph coloring algorithms (eg integrality gap examples [12]) are easy for high level Lasserre liftings since they all have low threshold rank.

When we try to apply similar ideas to general graphs, we quickly realize that the problematic cases (if they exist at all) must be such that different neighborhoods look “different.” Of course, this flies against the usual intuition about SDP relaxations: the usual reason for high integrality gaps (at least in explicit examples) is precisely that all neighborhoods look the same and the SDP gives no meaningful clues.

To quantify the notion of all neighborhoods “looking the same,” we focus on a specific kind of symmetric graph, the *distance transitive graphs*, which have been well-studied in graph theory (see the book [7]). In fact we restrict attention to such graphs that in addition have low diameter. The reason is that using simple combinatorial arguments one can show that in order to color the graph with n^ϵ colors, it suffices to restrict attention to graphs of diameter $O(1/\epsilon)$. If a 3-colorable distance transitive graph has diameter Δ we show how to find a $O(\log n)$ coloring in $O(n^{2^{O(\Delta)}})$ time. See Section 5.

How can our ideas be generalized to all graphs? In Section 6 we formulate a conjecture which if true would yield subexponential time coloring algorithms that find an n^ϵ -coloring.

2 The SDP and Lasserre Hierarchy

The standard SDP relaxation for graph 3-coloring uses *vector chromatic number*, but it is not amenable to Lasserre lifting. So we start with an equivalent (see [8]) relaxation based upon 0/1 variables. For each vertex p of the graph $G = (V, E)$, we have three variables $x_{p,R}, x_{p,Y}, x_{p,B}$ where $x_{p,C} = 1$ for $C \in \{R, Y, B\}$ “means” the vertex p is colored with color C . Thus exactly one of the three variables will be 1. The integer program makes sure $x_{p,R} + x_{p,B} + x_{p,Y} = 1$, and $x_{p,C}x_{q,C} = 0$ if p and q are adjacent.

Now we relax the integer program by replacing each $x_{p,C}$ with a vector $v_{p,C}$. The result is an SDP relaxation. Then we lift this SDP using k levels of Lasserre lifting. (For Lasserre lifting see the surveys [9,15]). The lifted SDP contains vector variables v_S , where S is a subset of the set $V \times \{R, Y, B\}$ (later denoted by Ω) and has size at most k . The resulting SDP is

$$\forall p \in V \quad v_{p,R} + v_{p,B} + v_{p,Y} = v_0 \quad (1)$$

$$\forall p \in V, C_1 \neq C_2 \quad \langle v_{p,C_1}, v_{p,C_2} \rangle = 0 \quad (2)$$

$$\forall (p, q) \in E, C \in \{R, Y, B\} \quad \langle v_{p,C}, v_{q,C} \rangle = 0 \quad (3)$$

$$\forall P, Q, S, T \subseteq \Omega, P \cup Q = S \cup T, |P \cup Q| \leq k \quad \langle v_P, v_Q \rangle = \langle v_S, v_T \rangle \quad (4)$$

In this SDP, Equations (1) to (3) are constraints obtained from the integer program; Equations (4) are the consistency constraints imposed by Lasserre

lifting; we also require $\langle v_\emptyset, v_\emptyset \rangle = 1$ for normalization. Notice that here we are abusing notation a bit: if the set S contains only one event (p, C) , we use both $v_{p,C}$ and v_S for the same vector. We call this SDP Las^k and its solution SDP^k .

2.1 Understanding the SDP Solution

Here we discuss how we should interpret the solution of coloring SDP. Throughout the discussion below, an “atomic event” (abbreviated to just “event” when this causes no confusion) consists of assigning a vertex p some color C . We denote by $\Omega = V \times \{R, Y, B\}$ the set of all atomic events. Our rounding algorithm will iteratively assign colors to vertices. Each step may assign a color C to p , or declare that color C will *never* be assigned to p . In the former case the atomic event (p, C) has happened; in the latter case the complement event happened. It is common to interpret the SDP solution as giving a distribution over these events whose probabilities are equal to the innerproducts of the vectors. We formulate this by the following theorem:

Theorem 2 ([14,8]). *A solution to k -level Lasserre lifting SDP (Las^k) encodes a locally consistent coloring for any set of k vertices. Locally consistent means all colorings with positive probability are valid colorings. If W is a set of atomic events then the probability that they happen is equal to the inner-product of v_S and v_T , where $S \cup T = W$. In particular, each vector can be decomposed as $v_W = r_W v_\emptyset + u_W$ where r_W is the probability that all events in W happen and u_W is perpendicular to v_\emptyset .*

If $w = (p, C)$ is an atomic event, properties of Lasserre lifting allow us to construct a subsolution in which event w happens (ie vertex p is assigned colored C), and a subsolution in which event w does not happen (ie color C is forbidden for p from now on). We randomly choose one of the subsolutions to preserve the probability of w . That is, if r_w is the probability of event w , we pick the subsolution in which w happens with probability r_w , and pick the subsolution in which w does not happen with probability $1 - r_w$. We call this “conditioning the solution on event w ”. The result of such an operation will be a solution for $k - 1$ level of Lasserre lifting, which we call SDP^{k-1} .

The computation needed to compute the new vectors in SDP^{k-1} is simple and follows from the above theorem: the probabilities of the new $k - 1$ -level solution must be the appropriate conditional probabilities in the locally consistent distributions in the k -level solution. For details see the full version or [3,9,8].

Note that we must use Lasserre instead of weaker relaxations: Sherali-Adams [17] and Lovász-Schrijver [16], because we consider solutions as locally consistent solutions (which rules out Lovász-Schrijver) and we use critically that probabilities correspond to inner-products of vectors (which rules out Sherali-Adams). Detailed comparison between the hierarchies are given in the surveys [9,15].

3 Global Correlation, Local Correlation and Rounding

Given a solution to the k -level lifting Las^k of a graph G , we shall define the global correlation of this solution and show how global correlation of order $\Omega(1/k)$ can help to round the solution. Intuitively global correlation measures the average correlation between the colors of two vertices chosen uniformly at random. In general, this correlation may be close to 0: knowing the color of one doesn't give much information about the color of the other. If the global correlation is bounded away from 0 however, then intuitively speaking, fixing the color for a randomly chosen vertex should bias the average remaining vertex a bit towards a particular color. Thus after fixing the colors for a sufficiently large set of vertices, the colors for most of the remaining vertices must get more or less fixed. This is the main idea of Barak et.al. [3] in the context of unique games, and Lemma 1 is adapted from there. The amount of variability in the color of the average vertex is quantified using *variance*.

We first examine how conditioning on one atomic event reduces the variance of another. Let w_1, w_2 be two atomic events, r_1, r_2 be their probabilities respectively, and r_{12} be the probability that both of them happen. The variance of the conditional random variable $w_2|w_1$ is given by:

$$\text{Var}[w_2|w_1] = \text{Var}[w_2] - \frac{(r_1 r_2 - r_{12})^2}{\text{Var}[w_1]}. \quad (5)$$

By the equation we see that the expected variance always drops, and the drop is proportional to $(r_1 r_2 - r_{12})^2$. Below we call this quantity the *correlation* between the two events.

Correlation has a geometric meaning in Lasserre solutions. Notice that $r_1 = \langle v_{w_1}, v_\emptyset \rangle$, $r_2 = \langle v_{w_2}, v_\emptyset \rangle$, and $r_{12} = \langle v_{w_1} v_{w_2} \rangle$ (by Theorem 2). As in Theorem 2 we express $v_{w_i} = r_i v_\emptyset + u_{w_i}$, then $\langle u_{w_1}, u_{w_2} \rangle = \langle v_{w_1}, v_{w_2} \rangle - r_1 r_2 = r_{12} - r_1 r_2$. Therefore we have $(r_1 r_2 - r_{12})^2 = \langle u_{w_1}, u_{w_2} \rangle^2$.

Definition 1 (Correlation, Global Correlation, Variance). *Given a solution SDP^k and two events w_1, w_2 , The correlation between w_1 and w_2 is defined as (where probabilities r and vectors u are as in Theorem 2):*

$$\text{Cor}[w_1, w_2] = (r_{w_1} r_{w_2} - r_{\{w_1, w_2\}})^2 = \langle u_{w_1}, u_{w_2} \rangle^2.$$

The global correlation of a set of vectors $\{z_p\}$ ($p \in U$) is just the expected correlation between two randomly picked vectors: $GC(\{z_p\}) = \mathbb{E}_{p, q \in U} \langle z_p, z_q \rangle^2$.

The global correlation of the SDP solution is the global correlation of all the vectors for the set of atomic events (Ω). Intuitively it is the average correlation between a pair of atomic events.

$$GC^k = \mathbb{E}_{w_1, w_2 \in \Omega} \langle u_{w_1}, u_{w_2} \rangle^2. \quad (6)$$

The variance of the solution is $VAR^k = \mathbb{E}_{w \in \Omega} r_w(1 - r_w)$.

Now we are ready to state the following Lemma for one step of rounding.

Lemma 1. *Suppose SDP solution SDP^k has global correlation GC^k and variance VAR^k . Upon picking a random event $w \in \Omega$ and conditioning on that event, the new solution SDP^{k-1} has expected variance at most $VAR^k - 4GC^k$.*

Proof. Due to space limit please see the full version.

Lemma [1](#) corresponds to a single step in our iterative rounding. So long as the global correlation is substantial —say, at least $10/k$ — we can repeat this step up to k times and drive the variance of the solution towards zero. Intuitively, once the variance is small enough, the solution should be almost integral and thus easy to round. Indeed we show the following:

Lemma 2. *Given a vector solution SDP^k ($k \geq 2$) for k -level Lasserre lifting Las^k , we say a vertex p is determined if there's a color C such that event $\{p, C\}$ that happens with probability more than $1/2$. Otherwise the vertex is undetermined. If SDP^k has variance $VAR^k < 1/8$ then at least $1/4$ of the vertices are determined. Moreover, if we color the determined vertices with the color that makes them determined, then this is a valid partial coloring (i.e., no two adjacent vertices will have the same color).*

Proof. First rewrite the definition of variance as $VAR^k = \mathbb{E}_{w \in \Omega} r_w(1 - r_w) = \mathbb{E}_{p \in V} \mathbb{E}_{C \in \{R, Y, B\}} r_{(p, C)}(1 - r_{(p, C)})$.

From this formula we know for any vertex p and its 3 events w_1, w_2, w_3 , their contribution to VAR^k is proportional to $(Var[w_1] + Var[w_2] + Var[w_3])/3$ (the second expectation in the right hand side). For undetermined vertices, the probabilities for w_1, w_2, w_3 can be more than $1/2$ and they sum up to 1, thus the minimum possible value of the contribution of this vertex p is $(1/4 + 1/4 + 0)/3 = 1/6$. If more than $3/4$ of the vertices are undetermined, we would have $VAR^k > 3/4 \cdot 1/6 = 1/8$, which contradicts our assumption.

For the moreover part, notice that the solution SDP^k is valid for the second level of Lasserre, which means it induces locally consistent distributions for any two vertices. For any edge (p, q) in the graph, if both p and q have events $\{p, C_1\}, \{q, C_2\}$ that happen with probability more than $1/2$, then we show $C_1 \neq C_2$. Suppose for contradiction that $C_1 = C_2 = C$. If we look at the distribution that the Lasserre solution induces on these two vertices, with positive probability both of them will be colored with color C . This contradicts with the validity of the Lasserre solution. Therefore we must have $C_1 \neq C_2$.

Local correlation. For an SDP solution, we would want to argue either Lemma [2](#) can be applied or the solution has large global correlation. To show this, we introduce local correlation as an intermediate step. We first show that if we cannot apply Lemma [2](#), the solution SDP^k always has *local correlation*, then we analyze the relationship between local correlation and global correlation in the next section and show high local correlation implies high global correlation.

For a vertex p , we construct a new vector $z_p = (u_{p,R}, u_{p,B}, u_{p,Y})$ (which means z_p is the concatenation of the 3 vectors, the vector u comes from Theorem [2](#)). It's easy to see that $\langle z_p, z_q \rangle = \sum_{C \in \{R, Y, B\}} \langle u_{p,C}, u_{q,C} \rangle$. Since $\langle z_p, z_q \rangle^2 \leq$

$27 \mathbb{E}_{C_1, C_2 \in \{R, Y, B\}} \langle u_{p, C_1}, u_{q, C_2} \rangle^2$, we know $GC \geq 1/27 \cdot \mathbb{E}_{p, q \in V} \langle z_p, z_q \rangle^2$. Hence the global correlation of the $\{z_p\}$ vectors $\mathbb{E}_{p, q \in V} \langle z_p, z_q \rangle^2$ can be used to lowerbound the global correlation of the solution SDP^k .

Local correlation is the expected correlation between endpoints of edges.

Definition 2 (Local Correlation). *Given a graph G and an SDP solution SDP^k , first construct vectors $z_p = (u_{p,R}, u_{p,B}, u_{p,Y})$. Then local correlation for this solution is defined to be $LC = \mathbb{E}_{(p,q) \in E} \langle z_p, z_q \rangle$.*

Local correlation depends on both the solution (SDP^k) and the graph G , unlike global correlation which only depends on the solution. Also, local correlation can be negative because we are not taking the squares of inner-products.

We shall prove the following Lemma which ensures high local correlation until we can find a large independent set.

Lemma 3. *If G is a regular 3-colorable graph, and in an SDP solution SDP^k at most $n/4$ vertices are determined in the sense of Lemma 2 then the local correlation $\mathbb{E}_{(p,q) \in E} \langle z_p, z_q \rangle \leq -1/8$.*

Proof. If (p, q) is an edge, both p and q are undetermined (as in Lemma 2), we shall prove $\langle z_p, z_q \rangle \leq -1/4$. Indeed, since (p, q) is an edge by (2) we have $\langle v_{p,R}, v_{q,R} \rangle = r_{p,R} r_{q,R} + \langle u_{p,R}, u_{q,R} \rangle = 0$. Which means $\langle z_p, z_q \rangle = -r_{p,R} r_{q,R} - r_{p,Y} r_{q,Y} - r_{p,B} r_{q,B} \leq -1/4$. The inequality holds because the r values are all in $[0, 1/2]$, the worst case for the r values are $(1/2, 1/2, 0)$ and $(0, 1/2, 1/2)$.

Since only $1/4$ of the vertices are determined (in the sense of Lemma 2), we consider the set S of undetermined vertices. At least $1/2$ of edges of G have both endpoints in S . Therefore $\mathbb{E}_{(p,q) \in E} \langle z_p, z_q \rangle \leq -1/4 * 1/2 = -1/8$.

4 Threshold Rank and Global Correlation

In this section we show how local correlation and global correlation are connected through *threshold rank*. Threshold rank of a graph $Rank_C(G)$ is defined by Arora et.al. in [1] as the number of eigenvalues larger than C . As they observed in [1], many problems have subexponential time algorithms when the underlying graph has low (i.e. sublinear) threshold rank. We show that 3-Coloring also lies in this category. If the underlying graph has low threshold rank, then an SDP solution will have high global correlation as long as it has local correlation.

Our definition for threshold rank is different from [1]. We are interested in eigenvalues that are smaller than a certain negative constant $-C$. For a graph G , we use $Rank_{-C}(G)$ to denote the number of eigenvalues of G 's normalized adjacency matrix whose value is at most $-C$. In all discussions C should be viewed as a positive constant, and we use negative sign to indicate that we are interested in eigenvalues smaller than $-C$.

Consider a convex relaxation of threshold rank given by Barak et.al. [3]. In this relaxation each vertex in the graph has a vector z_p (later we will see that they are indeed related to the vectors $\{z_p\}$ in Lemma 3). We try to maximize D , the reciprocal of global correlation subject to the following constraints

$$\mathbb{E}_{p \in V} \|z_p\|_2^2 = 1 \quad (7)$$

$$\mathbb{E}_{(p,q) \in E} \langle z_p, z_q \rangle \leq -C \quad (8)$$

$$\mathbb{E}_{p,q \in V} (\langle z_p, z_q \rangle)^2 \leq 1/D. \quad (9)$$

Barak et.al. [3] proved the following Lemma explaining why this is a relaxation to threshold rank. Due to space limit the proofs are omitted here.

Lemma 4. *If the $\text{Rank}_{-C/2}(G) = D$, then the optimal value D^* of the convex relaxation is at most $4D/C^2 = \text{Rank}_{-C/2}(G)/(C/2)^2$.*

Lemma [4] is important for our analysis because it implies if the local correlation (left-hand-side of Equation (8)) is smaller than a negative constant, and threshold rank is low, the global correlation (left-hand-side of Equation (9)) must be of order $\Omega(1/D)$. Now we are ready to prove Theorem [1].

Proof. Write the SDP in Section [2] with $c \cdot D$ levels of Lasserre lifting ($\text{Las}^{c \cdot D}$), and solve it in time $n^{O(D)}$. We apply the following rounding algorithm inspired by Lemma [1].

1. Initialize SOL to be $SDP^{c \cdot D}$
2. Repeat
3. If at least $n/4$ of the vertices are “determined” in SOL
4. Then apply Lemma [2] to get a partial coloring .
5. Pick a random event w , condition the solution SOL on this event
6. Until SOL is only valid for the first Level of Lasserre

Clearly, if the condition in Step 3 is satisfied and we proceed to Step 4, by Lemma [2] we get a partial coloring for $n/4$ vertices. In particular, one of the colors will have more than $n/12$ vertices, and they form an independent set. Therefore we only need to prove the probability that we reach Step 4 is large.

Let r_i be the probability that Step 4 is reached before iteration i . We would like to prove $r_{c \cdot D} \geq 1/2$. Assume we continue to run the algorithm even if Step 4 is reached (and we have already found an independent set). Let SOL_i be the solution at step i , GC_i be its global correlation and VAR_i be its variance.

We first prove the following Claim:

CLAIM: If the number of undetermined vertices in SOL_i is smaller than $n/4$, the global correlation GC_i is at least $\Omega(1/D)$.

Proof. Given the assumption, we can apply Lemma [3]. From the solution SOL_i , Lemma [3] constructs vectors $\{z_p\} (p \in V)$, and $\mathbb{E}_{(p,q) \in E} \langle z_p, z_q \rangle \leq -1/8$.

We shall normalize these vectors so that they satisfy Equations (7) and (8). The norm of z_p is $\|z_p\|_2^2 = \|u_{p,R}\|_2^2 + \|u_{p,Y}\|_2^2 + \|u_{p,B}\|_2^2 = 1 - r_{p,R}^2 - r_{p,Y}^2 - r_{p,B}^2$. Here $r_{p,X}$ is the probability that p is colored with color X , and the equation follows from Theorem [2]. If for vertex p no event has probability more than $1/2$,

then $\|z_p\|_2^2$ is a value between $1/4$ and 1 . As assumed the number of such vertices is at least $3n/4$ (otherwise Step 4 has already been performed), thus $E_{p \in V} \|z_p\|_2^2$ is between $3/16$ and 1 . We can normalize these vectors by multiplying with $c' = \sqrt{1/E_{p \in V} \|z_p\|_2^2}$. For the normalized vectors $\{\bar{z}_p\}$, we have $E_{p \in V} \|\bar{z}_p\|_2^2 = 1$. And since $c' \geq 1$ we still have $E_{(p,q) \in E} \langle \bar{z}_p, \bar{z}_q \rangle \leq -1/8$.

The vectors $\{\bar{z}_p\}$ satisfy Equation (7) and (8) for $C = -1/8$. Since we know $\text{Rank}_{-1/16}(G) = D$, Lemma 4 shows that the left-hand-side of Equation (9) must be at least $(1/16)^2/D = \Omega(1/D)$. That is, the global correlation between vectors $\{\bar{z}_p\}$ is at least $\Omega(1/D)$.

By analysis in Section 3, we know GC_i is within a constant factor of $E_{p,q} \langle z_p, z_q \rangle^2$. Since the normalization factor c' between z_p and \bar{z}_p is also bounded by a constant, $E_{p,q} \langle z_p, z_q \rangle^2$ and $E_{p,q} \langle \bar{z}_p, \bar{z}_q \rangle^2$ are also within a constant factor. Thus $GC_i \geq \Omega(1/D)$.

The proof proceeds as follows: when r_i , the probability that the solution has more than $3/4$ “determined” vertices, is large we can already get a good solution by applying the moreover part of Lemma 2. Otherwise we can apply the claim and Lemma 4 to conclude that the expected global correlation must be high at step i ; then Lemma 1 reduces r_i significantly.

In step i , with probability $1 - r_i$ the number of “determined” vertices (in the sense of Lemma 2) is smaller than $n/4$. When this happens (number of determined vertices small), Lemma 4 shows the global correlation is at least $\Omega(1/D)$. Therefore the expected global correlation at step i is at least $E[GC_i] \geq \Omega(1/D) * 1/2 = \Omega(1/D)$ (the expectation is over random choices of the algorithm) just by considering the situations when number of determined vertices is small. By Lemma 1 we know every time Step 5 is applied, the variance is expected to reduce by GC_i . That is, $E[VAR_{i+1}] \leq E[VAR_i] - 4E[GC_i] \leq E[VAR_i] - \Omega(1/D)$. If r_i remains smaller than $1/2$ for all the $c \cdot D$ rounds (where c is a large enough constant), we must have $E[VAR_{c \cdot D}] < 1/16$. By Markov’s Inequality with probability at least $1/2$ the variance is at most $1/8$, in which case Lemma 2 can be applied. That is, $r_{c \cdot D} \geq 1/2$. This is a contradiction and we must have $r_i \geq 1/2$ for some $i \leq c \cdot D$.

Therefore with probability at least $1/2$ the rounding algorithm will reach Step 4 and find a large independent set.

For the moreover part, we apply a random permutation π over the vertices before running the whole algorithm. In the permuted graph $n/12$ of the vertices are in the independent set S found by the algorithm above. If we apply the inverse permutation π^{-1} to the independent set found, we claim that any vertex q of the original graph is inside the independent set $\pi^{-1}(S)$ with probability at least $1/12$. This is because the graph is vertex transitive and essentially the algorithm cannot distinguish between vertices. More rigorous argument can be found in full version.

Repeat this procedure $100 \log n$ times, each vertex is in one of the $100 \log n$ independent sets with probability at least $1 - n^{-2}$. Union bound shows with high probability the union of these independent sets is the vertex set. We use

one color for each independent set (if a vertex belongs to multiple sets then choose an arbitrary one among them), which gives a valid $O(\log n)$ coloring.

5 Threshold Rank Bound for Distance Transitive Graphs

As we explained in the Introduction, symmetric graphs are a natural class of hard instances for graph coloring problem. Also, by Blum Coloring Tools [5], it is enough to consider graphs with low diameter for 3-Coloring.

In this section we focus on a class of symmetric graphs: distance transitive graphs, and we prove for a distance transitive graph with diameter Δ , the threshold rank $\text{Rank}_{-C}(G)$ is at most $(O(1/C^2))^\Delta$. We begin by defining distance transitive graphs:

Definition 3 (Distance Transitive Graph). *A graph $G = (V, E)$ is distance-transitive if for any pairs of vertices (p, q) and (s, t) , where the shortest-path distance between p, q and s, t are the same, there is always an automorphism that maps p to s and q to t .*

Distance transitive graphs have many nice properties, especially when we look at the neighborhoods of vertices. Define $\Gamma^k(p)$ to be the k -th neighborhood of p (which is the set of vertices at distance k of p), by the distance transitive condition, we know if a pair of vertices p, q have distance k , then $|\Gamma^{k-1}(p) \cap \Gamma(q)|$, $|\Gamma^k(p) \cap \Gamma(q)|$, $|\Gamma^{k+1}(p) \cap \Gamma(q)|$ are three numbers that depend only on k . As a convention, we call these numbers c_k , a_k and b_k respectively. The size of k -th neighborhood ($|\Gamma^k(p)|$) is represented by n_k . The following is known about spectral properties of distance transitive graphs[4]:

Lemma 5. *A distance transitive graph G has $\Delta + 1$ distinct eigenvalues, which are the eigenvalues of the matrix*

$$B = \begin{pmatrix} a_0 & c_1 & & & & \\ b_0 & a_1 & c_2 & & & \\ & & \cdot & \cdot & \cdot & \\ & & & b_{\Delta-2} & a_{\Delta-1} & c_\Delta \\ & & & & b_{\Delta-1} & a_\Delta \end{pmatrix}.$$

Moreover, suppose the i -th eigenvalue is λ_i ($\lambda_0 \geq \lambda_1 \geq \dots \geq \lambda_\Delta$), with left eigenvector u_i and right eigenvector v_i ($u_i^T B = \lambda_i u_i^T$, $B v_i = \lambda_i v_i$), we have $v_i(j) = n_j u_i(j)$. When v_i is normalized so that $v_i(0) = 1$, the multiplicity of λ_i in the original graph is $n / \langle u_i, v_i \rangle$.

Note that the eigenvalues in the above Lemma are for the adjacency matrix of G . There is a normalization factor d (the degree) between these eigenvalues and eigenvalues we were using for threshold rank. Now we are ready to prove the following theorem.

Theorem 3. *If a distance transitive graph is 3-colorable, then there is an algorithm that colors it with $O(\log n)$ colors in time $n^{2^{O(\Delta)}}$.*

Proof. Due to space limit please see the full version of this paper for the proof. The main idea is to prove that for each eigenvalue smaller than $-C$, its multiplicity must be smaller than $(10/C^2)^{\Delta}$ (the multiplicity can be computed by Lemma 5), then Theorem 1 gives the algorithm.

6 Conclusion

In this paper we explored the relationship between threshold rank and graph coloring. Unlike other problems such as Unique Games and MAX-CUT considered by Arora et.al. [1], we show that 3-Coloring is actually related to the negative side of the spectrum. We give an algorithm that can find linear size independent set when the graph is 3-colorable and has threshold rank D . The efficiency of our algorithm depends on the threshold rank of a graph. Known integrality gap examples [11,12] all have threshold rank that is polylog in the number of vertices. Thus our algorithm can detect in quasipolynomial time that they are not 3-Colorable. The relationship between global correlation and rounding and the convex relaxation for threshold rank are inspired by Barak et.al. [3] and we believe these techniques can be useful in other problems.

If our approach is combined with combinatorial tools, it could possibly lead to good subexponential (or even quasipolynomial-time) coloring algorithms. In particular, if the following conjecture is true for any constant C and $D = n^{\delta}$, we get a $\exp(n^{\delta})$ time algorithm for coloring 3-Colorable graph with n^{ϵ} colors (see full version). We have no counterexamples for the Conjecture when C is a constant and D is more than n^{ϵ} .

Conjecture 1. There exists an algorithm such that for any graph G , can either

- Find a subset S of vertices. The vertex expansion is at most $\Phi_V(S) \leq (n/|S|)^{1/C}$.
- Certify the existence of doubly stochastic matrix M with same support as G such that $\text{Rank}_{-1/16}(M) \leq D$.

We also give efficient algorithm to color 3-colorable distance transitive graphs with low diameter. These graphs have properties that seem to make it hard for previously known algorithms.

Acknowledgement. We thank David Steurer, Grant Schoenebeck and Aravindan Vijayaraghavan for valuable discussions. We especially thank David Steurer for sharing the result [3] with us.

References

1. Arora, S., Barak, B., Steurer, D.: Subexponential algorithms for unique games and related problems. In: Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science, FOCS 2010, pp. 563–572. IEEE Computer Society, Washington, DC, USA (2010)

2. Arora, S., Chlamtac, E., Charikar, M.: New approximation guarantee for chromatic number. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, STOC 2006, pp. 215–224. ACM, New York (2006)
3. Barak, B., Raghavendra, P., Steurer, D.: Rounding semidefinite programming hierarchies via global correlation (2011)
4. Biggs, N.L.: Intersection matrices for linear graphs. In: Welsh, D.J.A. (ed.) Combinatorial Mathematics and its Applications, pp. 15–23. Academic Press, London (1971)
5. Blum, A.: New approximation algorithms for graph coloring. *J. ACM* 41, 470–516 (1994)
6. Blum, A., Karger, D.: An $\tilde{O}(n^{3/14})$ -coloring algorithm for 3-colorable graphs. *Information Processing Letters* 61, 49–53 (1996)
7. Brouwer, A.E., Cohen, A.M., Neumaier, A.: Distance Regular Graphs. Springer, Heidelberg (1989)
8. Chlamtac, E.: Non-Local Analysis of SDP-Based Approximation Algorithms. PhD thesis, Princeton University (2009)
9. Chlamtac, E., Tulsiani, M.: Convex relaxations and integrality gaps. Unpublished manuscript
10. Dinur, I., Mossel, E., Regev, O.: Conditional hardness for approximate coloring. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, STOC 2006, pp. 344–353. ACM, New York (2006)
11. Feige, U., Langberg, M., Schechtman, G.: Graphs with tiny vector chromatic numbers and huge chromatic numbers. In: Proceedings of the 43rd Symposium on Foundations of Computer Science, FOCS 2002, pp. 283–292. IEEE Computer Society, Washington, DC, USA (2002)
12. Frankl, P., Rodl, V.: Forbidden intersections. *Transactions of the American Mathematical Society* 300(1), 259–286 (1987)
13. Karger, D., Motwani, R., Sudan, M.: Approximate graph coloring by semidefinite programming. *J. ACM* 45, 246–265 (1998)
14. Lasserre, J.B.: An explicit exact SDP relaxation for nonlinear 0-1 programs. In: Aardal, K., Gerards, B. (eds.) IPCO 2001. LNCS, vol. 2081, pp. 293–303. Springer, Heidelberg (2001)
15. Laurent, M.: A comparison of the sherali-adams, lovasz-schrijver and lasserre relaxations for 0-1 programming. *Mathematics of Operations Research* 28, 470–496 (2001)
16. Lovász, L., Schrijver, A.: Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization* 1, 166–190 (1991)
17. Sherali, H.D., Adams, W.P.: A hierarchy of relaxation between the continuous and convex hull representations. *SIAM J. Discret. Math.* 3, 411–430 (1990)
18. Szegedy, M.: A note on the θ number of lovasz and the generalized delarte bound. In: Proceedings of the 35th Annual Symposium on Foundations of Computer Science, pp. 36–39. IEEE Computer Society, Washington, DC, USA (1994)
19. Wigderson, A.: Improving the performance guarantee for approximate graph coloring. *J. ACM* 30, 729–735 (1983)
20. Zuckerman, D.: Linear degree extractors and the inapproximability of max clique and chromatic number. In: Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing, STOC 2006, pp. 681–690. ACM, New York (2006)

Inapproximability of NP-Complete Variants of Nash Equilibrium*

Per Austrin¹, Mark Braverman¹, and Eden Chlamtác²

¹ University of Toronto, Toronto, Canada
{`austrin,mbraverm`}@`cs.toronto.edu`

² Tel Aviv University, Tel Aviv, Israel
`chlamtac@cs.tau.ac.il`

Abstract. In recent work of Hazan and Krauthgamer (SICOMP 2011), it was shown that finding an ε -approximate Nash equilibrium with near-optimal value in a two-player game is as hard as finding a hidden clique of size $O(\log n)$ in the random graph $G(n, \frac{1}{2})$. This raises the question of whether a similar intractability holds for approximate Nash equilibrium without such constraints. We give evidence that the constraint of near-optimal value makes the problem distinctly harder: a simple algorithm finds an optimal $\frac{1}{2}$ -approximate equilibrium, while finding strictly better than $\frac{1}{2}$ -approximate equilibria is as hard as the Hidden Clique problem. This is in contrast to the unconstrained problem where more sophisticated algorithms, achieving better approximations, are known.

Unlike general Nash equilibrium, which is in PPAD, optimal (maximum value) Nash equilibrium is NP-hard. We proceed to show that optimal Nash equilibrium is just one of several known NP-hard problems related to Nash equilibrium, all of which have approximate variants which are as hard as finding a planted clique. In particular, we show this for approximate variants of the following problems: finding a Nash equilibrium with value greater than η (for any $\eta > 0$, even when the best Nash equilibrium has value $1 - \eta$), finding a second Nash equilibrium, and finding a Nash equilibrium with small support.

Finally, we consider the complexity of approximate pure Bayes Nash equilibria in two-player games. Here we show that for general Bayesian games the problem is NP-hard. For the special case where the distribution over types is uniform, we give a quasi-polynomial time algorithm matched by a hardness result based on the Hidden Clique problem.

1 Introduction

The classical notion of Nash equilibrium is the most fundamental concept in the theory of non-cooperative games. In recent years, there has been much work on the complexity of finding a Nash equilibrium in a given game. In particular, a series of hardness results led to the work of Chen et. al [CDT09], who showed

* Full version available as arXiv:1104.3760. Research supported by NSERC. This work was done while the third author was at the University of Toronto.

that even for two-player (bimatrix) games, the problem of computing a Nash equilibrium is PPAD-complete, thus unlikely to be solvable in polynomial time.

Therefore, it makes sense to consider the complexity of approximate equilibria. In particular, a notion which has emerged as the focus of several works is that of an ε -approximate Nash equilibrium, or ε -equilibrium for short, where neither player can gain more than ε (additively) by defecting to a different strategy (without loss of generality, all payoffs are scaled to lie in the interval $[0, 1]$). A straightforward sampling argument of Lipton et al. [LMM03] shows that in every game, there exist ε -equilibria with support $O(\log n/\varepsilon^2)$, and so they can be found in quasi-polynomial time $n^{O(\log n/\varepsilon^2)}$ by exhaustive search.

On the other hand, finding good polynomial time approximations has proved more challenging. While finding a $\frac{1}{2}$ -equilibrium turns out to be quite simple [DMP09], more complicated algorithms have given a series of improvements [DMP07, BBM10, TS08], where the current best known is the 0.3393-equilibrium shown by Tsaknakis and Spirakis [TS08]. A major open question in this area is whether or not there exists a PTAS for Nash Equilibrium (note that the algorithm of Lipton et al. [LMM03] gives a quasi-polynomial time approximation scheme for the problem).

Recently, Hazan and Krauthgamer [HK11] have attempted to provide evidence for the optimality of the QPTAS of Lipton et al. [LMM03], by showing a reduction from a well-studied and seemingly intractable problem (which can also be solved in quasi-polynomial time) to the related problem of finding an ε -equilibrium with near maximum value (the value of an equilibrium is the average of the payoffs of the two players).

The problem they reduce from is the *Hidden Clique Problem*: Given a graph sampled from $G(n, \frac{1}{2})$ with a planted (but hidden) clique of size k , find the planted clique. Since with high probability the maximum clique in $G(n, \frac{1}{2})$ is of size $(2 - o(1)) \log n$, it is easy to see that for constant $\delta > 0$, one can distinguish between $G(n, \frac{1}{2})$ and $G(n, \frac{1}{2})$ with a planted clique of size $k > (2 + \delta) \log n$ in quasi-polynomial time by exhaustive search over all subsets of $(2 + \delta) \log n$ vertices. It is also not hard to extend this to an algorithm which finds the hidden clique in quasi-polynomial time.

On the other hand, the best known polynomial time algorithm, due to Alon et al. [AKS98] only finds cliques of size $\Omega(\sqrt{n})$. In fact, Feige and Krauthgamer [FK03] show that even extending this approach by using the Lovász-Schrijver SDP hierarchy, one still requires $\Omega(\log n)$ levels of the hierarchy (corresponding to $n^{\Omega(\log n)}$ running time to solve the SDP) just to find a hidden clique of size $n^{1/2-\varepsilon}$. The only possible approach we are aware of for breaking the $\Omega(\sqrt{n})$ barrier would still (assuming certain conjectures) only discover cliques of size $\Omega(n^c)$ for some constant $c > 0$ [FK08, BV09].

Hazan and Krauthgamer show that finding a near-optimal ε -equilibrium is as hard as finding hidden cliques of size $C \log n$, for some universal constant C . Here, by near-optimal we mean having value close to maximum possible value obtained in an actual Nash equilibrium. Subsequently, Minder and Vilenchik [MV09] improved this hardness to planted cliques of size $(2 + \delta) \log n$ for arbitrarily small

$\delta > 0$.¹ Here, we will rely on the hardness assumption for hidden cliques of size $C \log n$ for any constant C , and will not attempt to optimize the value of C .

1.1 A Sharp Result for Near-Optimal Approximate Nash

It is important to note that the problem considered in [HK11] is not equivalent to finding an unconstrained ε -equilibrium. In light of the results of [HK11, MV09] it is natural to ask to what extent the hardness for near-optimal approximate equilibrium gives an indication of hardness for unconstrained approximate equilibrium. Indeed, [HK11], in their concluding remarks, ask whether their methods can be used to rule out a PTAS for unconstrained Nash equilibrium. One of the messages of this paper is that these two problems are quite different in terms of approximability and that one should not yet be overly pessimistic about the possibility for a PTAS for unconstrained Nash equilibrium. Indeed, while there is a polynomial time algorithm to find a 0.3393-equilibrium, we show that finding a near-optimal $(\frac{1}{2} - \eta)$ -equilibrium is hard.

Theorem 1.1 (Informal). *For every constant $\eta > 0$, finding a near-optimal $(\frac{1}{2} - \eta)$ -approximate equilibrium is as hard as finding a hidden clique of size $C \log n$ in $G(n, \frac{1}{2})$.*

As mentioned above, there is a simple polynomial time algorithm to find a $\frac{1}{2}$ -equilibrium, and we show that this algorithm can be extended to find a $\frac{1}{2}$ -equilibrium with value at least that of the best exact equilibrium:

Theorem 1.2 (Informal). *There exists a polynomial time algorithm to find a $\frac{1}{2}$ -approximate equilibrium with value at least that of the optimal true equilibrium.*

Thus, Theorem 1.1 is tight and unlike unconstrained Nash equilibrium, where stronger techniques yield approximations better than $\frac{1}{2}$, near-optimal Nash equilibrium does not admit efficient “non-trivial” approximations (assuming the Hidden Clique problem is hard).

1.2 The Bigger Picture: Hardness for NP-Hard Variants of Nash

Just like with unconstrained ε -equilibrium, finding a near-optimal ε -equilibrium can be done in quasi-polynomial time using the algorithm of [LMM03]. However, the exact version – finding a maximum value Nash equilibrium – is NP-hard [GZ89] and therefore harder than its unconstrained counterpart which is in PPAD [Pap94]. In fact, maximum value Nash is one of several optimization variants of Nash equilibrium which are NP-complete. Other variants include: determining whether a bimatrix game has more than one Nash equilibrium [GZ89], finding a Nash Equilibrium with minimum support [GZ89], and determining whether the maximum value equilibrium has value at least $1 - \frac{1}{n}$ or at most ε/n

¹ There is a small caveat: the reduction of [MV09] only certifies the presence of a hidden clique, but does not identify the vertices of the clique.

(even for arbitrarily small $\varepsilon = \varepsilon(n) > 0$) [CS03]. We show that approximate-equilibrium variants of these problems are also as hard as Hidden Clique.

For the problem of obtaining any non-trivial approximation to the optimal value of a Nash equilibrium, we prove the following theorem.

Theorem 1.3 (Informal). *For every constant $\eta > 0$, finding an ε -equilibrium with value at least η is as hard as finding a hidden clique of size $C \log n$ in $G(n, \frac{1}{2})$, even in a game having an equilibrium of value $\geq 1 - \eta$.*

For the case of determining whether a game has more than one equilibrium, note that by continuity considerations, every two-player game has an infinite number of ε -equilibria. Thus, the appropriate approximate analog is to consider the problem of finding two ε -equilibria with (at least) a certain total variation distance between them. We show that this is also as hard as Hidden Clique.

Theorem 1.4 (Informal). *For all sufficiently small constant $\epsilon > 0$, determining whether a game has two different ε -approximate equilibria, having statistical distance at least 3ϵ , is as hard as finding a hidden clique of size $C \log n$ in $G(n, \frac{1}{2})$.*

We then move to the problem of finding an equilibrium with small support. Recall that by [LMM03], there exist ε -Nash equilibria with support $O(\log n/\varepsilon^2)$. It is also known that for any $\eta > 0$, in certain two-player games all $(\frac{1}{2} - \eta)$ -equilibria must have support at least $\log n/(1 + \log(1/\eta))$ [FNS07] (the threshold of $\frac{1}{2}$ is tight, since the simple $\frac{1}{2}$ -equilibrium of [DMP09] has support 3). As an approximate-equilibrium variant of the Minimum Support Equilibrium problem, we consider the problem of finding an ε -equilibrium with support at most some threshold t , and prove the following hardness result.

Theorem 1.5 (Informal). *For every constant $\eta > 0$, finding a $(\frac{1}{2} - \eta)$ -equilibrium with support size $C' \log n$ is as hard as finding a hidden clique of size $C \log n$ in $G(n, \frac{1}{2})$.*

This can be seen as a complexity-theoretic analogue of the lower bound of [FNS07] mentioned above. Again, this contrasts with the case of unconstrained equilibrium, which is guaranteed to exist, and admits stronger approximations.

While these are all negative results, we again would like to stress that there is a positive message to this story: these problems are hard because they are approximate versions of NP-complete problems, not because they are approximate variants of Nash equilibrium. Therefore, these results should *not* be viewed as indications that Nash equilibrium does not have a PTAS.

1.3 The Complexity of Approximate Pure Bayes Nash Equilibria

Finally, we consider the problem of approximating pure *Bayes Nash Equilibria* (BNE) in two-player games. *Bayesian games* model the situation where the players' knowledge of the world is incomplete. In a Bayesian game, both players may be in one of a number of different states, known as *types*, representing what each player knows about the state of the world, and the payoff of each

player depends on the type of both players in addition to their strategies. The types are distributed according to some joint distribution and are not necessarily independent. A pure strategy for a Bayesian game assigns to each type a strategy that the player plays when she is in that type. In a pure BNE, conditioning on any given type for a given player, the player cannot gain by changing his strategy for that type. See Section 6 for precise definitions.

Conitzer and Sandholm [CS03] have shown that determining whether a given two-player game has a pure BNE is NP-complete. We show that this holds also for approximate pure BNE.

Theorem 1.6 (Informal). *Let $\varepsilon = 0.004$. Then given a Bayesian game that admits a pure BNE, it is NP-hard to find a pure ε -BNE for the game.*

This hardness result relies heavily on the joint distribution of the players' types being non-uniform (not even a product distribution). We show that when the distribution over type pairs is uniform, there is in fact a quasi-polynomial time algorithm for ε -approximate pure BNE (when a pure BNE exists).

Theorem 1.7 (Informal). *For every constant $\varepsilon > 0$ there is a quasipolynomial time algorithm to find a pure ε -BNE in two-player Bayesian games with uniformly distributed types and in which a pure BNE exists.*

We remark that this algorithm extends easily to arbitrary product distributions over types but in order to keep the notation simple we restrict our attention to the uniform case. The quasi-polynomial running time of the algorithm is possibly optimal: it follows immediately from our hardness for Small Support Equilibrium that this problem is also as hard as Hidden Clique.

Theorem 1.8 (Informal). *For every constant $\eta > 0$, finding a $(\frac{1}{4} - \eta)$ -approximate pure BNE in a two-player Bayesian game with uniformly distributed types and in which a pure BNE exists is as hard as finding a hidden clique of size $C \log n$.*

1.4 Organization

In Section 3 we prove the results relating to approximate equilibria with large value: Theorem 1.1 (Section 3.2), Theorem 1.2 (Section 3.4), and Theorem 1.3 (Section 3.3). In Section 4 we prove Theorem 1.4 by a black-box application of Theorem 1.3. In Section 5 we prove Theorem 1.5 using similar techniques as for the hardness results of Section 3. In Section 6 we prove our results for Bayesian games, Theorems 1.6, 1.7, and 1.8. Due to space restrictions, most of the actual proofs are deferred to the full version of the paper [ABC11].

2 Preliminaries

A *bimatrix game* $\mathcal{G} = (M_{\text{row}}, M_{\text{col}})$ is a game defined by two finite matrices, M_{row} and M_{col} , and two players: the *row player* and the *column player*. We

assume throughout that the game is normalized, i.e. both matrices have values in the interval $[0, 1]$. The row and column players choose *strategies* x and y respectively, where x, y are nonnegative vectors satisfying $\sum_i x_i = \sum_j y_j = 1$. A *pure strategy* has support 1 (i.e. a vector with 1 in one entry and 0 in the rest). The row (resp. column) player's *payoff* is given by $x^\top M_{\text{row}} y$ (resp. $x^\top M_{\text{col}} y$).

A *Nash equilibrium* is a pair of strategies (x, y) such that neither player has any incentive to deviate to a different strategy, assuming the other player does not deviate. Formally, in an equilibrium, for all i, j we have $e_i^\top M_{\text{row}} y \leq x^\top M_{\text{row}} y$ and $x^\top M_{\text{col}} e_j \leq x^\top M_{\text{col}} y$. An ε -approximate Nash equilibrium, or ε -*equilibrium* for short, is a pair of strategies x, y where each player has incentive at most ε to deviate. That is, for all i, j ,

$$e_i^\top M_{\text{row}} y \leq x^\top M_{\text{row}} y + \varepsilon \quad \text{and} \quad x^\top M_{\text{col}} e_j \leq x^\top M_{\text{col}} y + \varepsilon.$$

The *value* of a pair of strategies, denoted $v_G(x, y)$, is the average payoff of the two players, i.e.,

$$v_G(x, y) = \frac{1}{2}(x^\top M_{\text{row}} y + x^\top M_{\text{col}} y) = \frac{1}{2} \sum_{i,j} x_i y_j (M_{\text{row}}(i, j) + M_{\text{col}}(i, j)).$$

For a vector $x \in \mathbb{R}^n$ and $S \subseteq [n]$, we write x_S for the projection of x to the coordinates S . We write $\|x\| = \sum_{i=1}^n |x_i|$ for the ℓ_1 norm of x . Thus, for a strategy (in other words, a probability distribution) $x \in [0, 1]^n$ we write $\|x_S\|$ for the probability that the player plays an element of S .

Further, for a set $S \subseteq [n]$ of strategies, we use $v_{G|S}(x, y)$ to denote the value of (x, y) *conditioned* on both players playing in S . Formally,

$$v_{G|S}(x, y) = \mathbb{E}_{i \sim x, j \sim y} \left[\frac{1}{2}(M_{\text{row}}(i, j) + M_{\text{col}}(i, j)) \mid i, j \in S \right] = v_G(x_S, y_S) / (\|x_S\| \|y_S\|).$$

(If $\|x_S\| = 0$ or $\|y_S\| = 0$, $v_{G|S}(x, y)$ is undefined.)

Given an undirected graph $G = (V, E)$, and (not necessarily disjoint) vertex sets $S_1, S_2 \subseteq V$, we will denote by $E(S_1, S_2)$ the set of ordered pairs $\{(i, j) \in S_1 \times S_2 \mid \{i, j\} \in E \text{ or } i = j\}$. We will refer to $d(S_1, S_2) = |E(S_1, S_2)| / (|S_1| |S_2|)$ as the *density* of the pair (S_1, S_2) .

3 Approximate Equilibria with Good Value

3.1 The Reduction

In this section we describe the general reduction that we use to prove Theorems [1.1](#) and [1.3](#) and describe its properties. This reduction also forms the basis for the reductions we use to prove Theorems [1.4](#), [1.5](#) and [1.8](#). It is based on the reduction of [\[HK11\]](#).

As in [\[HK11\]](#) our soundness analysis proceeds by using the approximate equilibrium to find a dense bipartite subgraph of G . The following lemma shows that this is sufficient to recover the hidden clique.

Lemma 3.1 ([HK11, Lemma 5.3]). *There exist universal constants c_1 and c_2 such that the following holds. Let G be a sample from $G(n, \frac{1}{2})$ with a hidden clique of size $C \log n$ for some $C \geq c_1$. Then, given a pair of vertex sets $S_1, S_2 \subseteq [n]$ of size $c_2 \log n$ and density $d(S_1, S_2) \geq 5/9$ we can in polynomial time reconstruct the hidden clique (with high probability over G).*

The lemma is slightly different from Lemma 5.3 of [HK11]: there we start with a bipartite subgraph of density $3/5$ instead of $5/9$ but this minor difference only changes the value of the constant c_2 – the lemma holds for any constant density strictly larger than $\frac{1}{2}$.

We now describe the reduction. It is controlled by three parameters $\alpha, \beta, \gamma \in (0, 1)$. Setting these parameters appropriately gives the various hardness results.

Reduction 3.2 *Let $G = (V, E)$ be an n vertex graph and A its adjacency matrix (with 1s on the diagonal). Then, for parameters $\alpha, \beta, \gamma \in (0, 1)$, we define a (random) bimatrix game $\mathcal{G} := \mathcal{G}(G, \alpha, \beta, \gamma)$ as follows.*

Let $N = n^c$ where $c = (c_2 + 1) \log 1/\beta$ for the universal constant c_2 of Lemma 3.1. Pick a random $N \times n$ matrix B whose entries are i.i.d. $\{0, 1\}$ variables with expectation β . Then $\mathcal{G} = (M_{\text{row}}, M_{\text{col}})$, where the payoff matrices are:

$$M_{\text{row}} = \begin{pmatrix} \alpha A & 0 \\ B & \gamma J \end{pmatrix} \quad M_{\text{col}} = \begin{pmatrix} \alpha A & B^\top \\ 0 & \gamma J \end{pmatrix}, \quad (1)$$

where J is the all-ones $N \times N$ matrix.

We conclude this section with an additional lemma which shows how to obtain a dense bipartite subgraph given an approximate equilibrium of \mathcal{G} with certain properties. This lemma (and its proof) is analogous to [HK11, Lemma 5.2], but as we need it in larger generality we also give the proof (deferred to the full version [ABCT1]).

Lemma 3.3. *Let \mathcal{G} be as in Reduction 3.2. Fix any $s \in [0, 1]$, $t \in [0, 1]$ and $\varepsilon \in [0, 1]$ such that $1 - t - 3\sqrt{s}/2 \geq \alpha + \varepsilon$, and let (x, y) be an ε -approximate equilibrium of \mathcal{G} with the following two properties:*

- Both $\|x_{[n]}\| \geq 1 - t$ and $\|y_{[n]}\| \geq 1 - t$.
- The conditional value $v_{\mathcal{G}|[n]}(x, y) \geq (1 - s)\alpha$.

Then, given (x, y) as above, we can efficiently find vertex sets $S_1, S_2 \subseteq [n]$ each of size $c_2 \log n$ and density $d(S_1, S_2) \geq 5/9$.

3.2 Hardness for ε Close to $\frac{1}{2}$

To obtain Theorem 1.1 the main requirement is to set $\alpha = \frac{1}{2} + O(\eta)$. The values of β and γ are essentially irrelevant in this case – the only thing needed is that β is bounded away from both 0 and α and that $\gamma \leq \frac{1}{2}$.

Lemma 3.4. *Let $\alpha = \frac{1}{2} + t$, $\gamma \leq \frac{1}{2}$ and \mathcal{G} be the game of Reduction 3.2. Then for any pair of strategies (x, y) with value at least $v_{\mathcal{G}}(x, y) \geq \alpha - t^2$ it holds that $\|x_{[n]}\|$ and $\|y_{[n]}\|$ are both at least $1 - t$.*

Observation 3.5 *Let (x, y) be any pair of strategies with value $v_{\mathcal{G}}(x, y) \geq \frac{1}{2}$ and $\|x_{[n]}\| > 0$, $\|y_{[n]}\| > 0$. Then $v_{\mathcal{G}|_{[n]}}(x, y) \geq v_{\mathcal{G}}(x, y)$, provided that $\gamma \leq \frac{1}{2}$.*

Plugging this into Lemma 3.3, we can now easily complete the proof of hardness for ε close to $\frac{1}{2}$.

Theorem 3.6 (Detailed Statement of Theorem 1.1). *For every $\eta > 0$ there exist $\delta = \Omega(\eta^2)$, $\alpha \geq \frac{1}{2}$ and universal constant C not depending on η such that the following holds. Given a graph $G = (V, E)$ we can in randomized polynomial time construct a bimatrix game \mathcal{G} with maximum value α (over all strategy pairs) such that, if $G = G(n, \frac{1}{2})$ with a hidden clique of size $C \log n$, the following holds (w.h.p. over G and \mathcal{G}):*

Completeness. *There is a Nash equilibrium (x, y) with value α .*

Soundness. *Given any $(\frac{1}{2} - \eta)$ -equilibrium with value $\geq \alpha - \delta$, we can efficiently recover the hidden clique.*

3.3 Distinguishing between Low and High Value

For Theorem 1.3 the choices of all three parameters α, β, γ of Reduction 3.2 are important. We are going to set γ close to 0, and $\alpha > \beta$ both close to 1.

On a high level, the proof has the same structure as that of Theorem 1.1. However, in the current setting Lemma 3.4 and Observation 3.5 do not apply. To arrive at similar conclusions we use a different argument, exploiting the fact that (x, y) is a ε -equilibrium. Essentially, the argument is as follows: the off-diagonal blocks (B and B^\top) are not stable, since there is too much incentive for at least one player to deviate. Therefore, most of the probability mass in an equilibrium is concentrated either in the αA block, or in the γJ block. However, in the γJ block, the value is too small. So, if the equilibrium has even slightly larger value, its mass must be concentrated in the αA block. There it has to actually have very large value, since otherwise, there is incentive for both players to deviate to B and B^\top to get reward β . The rest of the proof follows as before.

Formally, we show that (under certain conditions) any ε -equilibrium with non-negligible value must satisfy the conditions of Lemma 3.3:

Lemma 3.7. *Fix a parameter $\varepsilon \in (0, 1)$, let $\alpha - \beta \leq \varepsilon$, and $\gamma = 4\sqrt{\varepsilon}$ and consider the game \mathcal{G} as in Reduction 3.2.*

Then, w.h.p. over \mathcal{G} , any ε -equilibrium (x, y) with value $> 5\sqrt{\varepsilon}$ satisfies:

- Both $\|x_{[n]}\|$ and $\|y_{[n]}\|$ are at least $1 - \sqrt{\varepsilon}$.
- $v_{\mathcal{G}|_{[n]}}(x, y) \geq \alpha - 3\varepsilon$.

Equipped with Lemma 3.7, it is easy to finish the proof of Theorem 1.3.

Theorem 3.8 (Detailed statement of Theorem 1.3). *For every constant $\eta > 0$ there exist $\varepsilon = \Omega(\eta^2)$ and $C = O(1/\eta^3)$ such that the following holds. Given a graph G , we can in randomized polynomial time construct a bimatrix game \mathcal{G} such that, if $G = G(n, \frac{1}{2})$ with a hidden clique of size $C \log n$, the following holds (w.h.p. over G and \mathcal{G}):*

Completeness. *There is a Nash equilibrium (x, y) with both payoffs $\geq 1 - \eta$.*

Soundness. *Given any ε -equilibrium with value $\geq \eta$, we can efficiently recover the hidden clique.*

3.4 An Algorithm for Good $\frac{1}{2}$ -Approximate Equilibria

In this section we prove Theorem 1.2 by describing a simple algorithm to find a $\frac{1}{2}$ -approximate Nash equilibrium with at least as good value as the best exact Nash equilibrium. This shows that the bound on ε in Theorem 1.1 is tight.

For general $\frac{1}{2}$ -approximate equilibria (without any constraint on the value), the following simple algorithm was suggested by Daskalakis, Mehta and Papadimitiou [DMP09]. Start by choosing an arbitrary pure strategy e_i for the row player, let e_j be the column player's best response to e_i , and let e_k be the row player's best response to e_j . Then the following is a $\frac{1}{2}$ -equilibrium: let the column player play e_j , and let the row player play e_i with probability $\frac{1}{2}$ and e_k with probability $\frac{1}{2}$ (neither player can gain more than $\frac{1}{2}$ by deviating, since each player is playing a best response strategy with probability $\frac{1}{2}$). Thus, every bimatrix game has a $\frac{1}{2}$ -approximate equilibrium in which one of the players plays a pure strategy. We show that this is also the case for optimal value $\frac{1}{2}$ -equilibria (the difference being that in this case, the other player may play a mixed strategy with arbitrarily large support – not just two strategies). Theorem 1.2 follows easily from the following lemma:

Lemma 3.9. *For every bimatrix game which has a Nash equilibrium of value v , there exists a $\frac{1}{2}$ -approximate equilibrium with value at least v in which one of the players plays a pure strategy.*

4 Finding a Second Equilibrium

In the following Theorem, d_{TV} refers to the total variation distance between two vectors, i.e., $d_{TV}(x, y) = \frac{1}{2} \sum |x_i - y_i|$.

Theorem 4.1 (Detailed Statement of Theorem 1.4). *There is a $C > 0$ such that the following holds for all sufficiently small $\varepsilon > 0$. Given a graph G we can in randomized polynomial time construct a bimatrix game \mathcal{G}' which admits a pure Nash equilibrium (e_i, e_j) such that, if $G = G(n, \frac{1}{2})$ with a hidden clique of size $C \log n$, the following holds (w.h.p. over G and \mathcal{G}'):*

Completeness. *There is an equilibrium (x, y) with $d_{TV}(e_i, x) = d_{TV}(e_j, y) = 1$.*

Soundness. *Given any ε -equilibrium (x, y) of \mathcal{G}' with $d_{TV}(e_i, x) \geq \varepsilon + O(\varepsilon^2)$ or $d_{TV}(e_j, y) \geq \varepsilon + O(\varepsilon^2)$, we can efficiently recover the hidden clique.*

Remark 4.2. Note that the bound $\varepsilon + O(\varepsilon^2)$ on the statistical distance is almost tight: given any true equilibrium (x, y) there are ε -approximate equilibria (x', y') with $d_{TV}(x, x') \geq \varepsilon$ and $d_{TV}(y, y') \geq \varepsilon$.

5 Small Support Equilibria

In this section, we show hardness of finding an ε -approximate Nash equilibrium with small (logarithmic) support when one exists, even for ε close to $\frac{1}{2}$. Note that an ε -approximate Nash equilibrium for two-player n' -strategy games with support at most $O(\log n'/\varepsilon)$ is guaranteed to exist by the algorithm of Lipton et al. [LMM03]. Here we consider approximate equilibria with smaller (but still logarithmic) support. Also, note that this is tight, since for $\varepsilon = \frac{1}{2}$, we have the simple algorithm of [DMP09], which gives a $\frac{1}{2}$ equilibrium of support 3.

Theorem 5.1. *For every $\eta > 0$ there exists $C > 0$ such that finding a $(\frac{1}{2} - \eta)$ -equilibrium with support at most $(\log n)/2$ is as hard as finding a hidden clique of size $C \log n$ in $G(n, \frac{1}{2})$.*

Note that we have a much smaller gap between the completeness and hardness above than in the other problems we have considered. In particular, we do not claim that finding a $(\frac{1}{2} - \eta)$ -equilibrium with small support is hard even when an exact equilibrium with small support exists. However, such hardness can be shown for a smaller additive approximation:

Theorem 5.2. *For every $\eta > 0$ there exists $C > 0$ such that finding a $(\frac{1}{4} - \eta)$ -equilibrium with support at most $O(\log n)$ in a two-player game which admits an exact Nash equilibrium with support $O(\log n)$ is as hard as finding a hidden clique of size $C \log n$ in $G(n, \frac{1}{2})$.*

6 Computing Approximate Pure Bayes-Nash Equilibrium

We focus on Bayesian games with two players, but the results generalize to an arbitrary number of players. More details on Bayesian games can be found in most Game Theory textbooks, for example in [FT91].

In a *Bayesian game* the payoff of the players depends on the state of the world in addition to the players' strategies. In a situation with two players, the row player and the column player, each player is presented with a signal, called type, about the state of the world $\theta_{\text{row}} \in \Theta_{\text{row}}$ and $\theta_{\text{col}} \in \Theta_{\text{col}}$, respectively. The types are distributed according to some joint distribution \mathcal{P} and are not necessarily independent. The types determine the payoff matrices $M_{\text{row}}(\theta_{\text{row}}, \theta_{\text{col}})$ and $M_{\text{col}}(\theta_{\text{row}}, \theta_{\text{col}})$. Denote the set of rows and columns in this matrix by S_{row} and S_{col} , respectively. Each player chooses an action $s_{\text{row}} \in S_{\text{row}}$ and $s_{\text{col}} \in S_{\text{col}}$

from their respective set of actions. The payoff function of the first player is thus $u_{\text{row}}(s_{\text{row}}, s_{\text{col}}, \theta_{\text{row}}, \theta_{\text{col}}) = M_{\text{row}}(\theta_{\text{row}}, \theta_{\text{col}})_{s_{\text{row}}, s_{\text{col}}} \in [0, 1]$. The payoff function u_{col} is defined similarly. The payoff matrices, that depend on the players' types, as well as the distribution on types is known to the players ahead of the game.

A *pure strategy* for the row player in a Bayesian game is a function (that by a slight abuse of notation) we denote by $s_{\text{row}} : \Theta_{\text{row}} \rightarrow S_{\text{row}}$ that for each type θ_{row} as observed by row player associates a strategy $s_{\text{row}}(\theta_{\text{row}})$ that the player chooses to execute. A pure strategy $s_{\text{col}} : \Theta_{\text{col}} \rightarrow S_{\text{col}}$ is defined similarly.

Denote by $\mathcal{P}_{\theta_{\text{row}}}$ the distribution on player column player's types θ_{col} conditioned on the type θ_{row} being observed. For a pair of pure strategies $(s_{\text{row}}, s_{\text{col}})$ the payoff function of the row player is given by

$$p_{\text{row}}(\theta_{\text{row}}) = \mathbb{E}_{\theta_{\text{col}} \sim \mathcal{P}_{\theta_{\text{row}}}} [u_{\text{row}}(s_{\text{row}}(\theta_{\text{row}}), s_{\text{col}}(\theta_{\text{col}}), \theta_{\text{row}}, \theta_{\text{col}})].$$

A *pure strategy Nash equilibrium* in a Bayesian game, is a pair of functions $s_{\text{row}}, s_{\text{col}}$ s.t. for all types observed, neither player has an incentive to deviate from his current strategy. In other words, for each θ_{row} , and for each $s'_{\text{row}} \in S_{\text{row}}$,

$$p_{\text{row}}(\theta_{\text{row}}) \geq \mathbb{E}_{\theta_{\text{col}} \sim \mathcal{P}_{\theta_{\text{row}}}} [u_{\text{row}}(s'_{\text{row}}, s_{\text{col}}(\theta_{\text{col}}), \theta_{\text{row}}, \theta_{\text{col}})],$$

and a similar condition holds for p_{col} .

Since a pure Nash equilibrium need not exist in non-Bayesian games, it need not exist in Bayesian games either. Moreover, while verifying whether a non-Bayesian two player game has a pure Nash equilibrium is trivial, verifying whether a pure Bayesian Nash equilibrium exists is NP-hard [CS03]. Furthermore, this problem remains hard even when the distribution on types is uniform and the payoff does not depend on the players' types.

A *pure ε -Bayesian Nash equilibrium (ε -BNE)* is similar to an ε -Nash equilibrium. For each observed type θ_{row} , the incentive to deviate is bounded:

$$p_{\text{row}}(\theta_{\text{row}}) > \mathbb{E}_{\theta_{\text{col}} \sim \mathcal{P}_{\theta_{\text{row}}}} [u_{\text{row}}(s'_{\text{row}}, s_{\text{col}}(\theta_{\text{col}}), \theta_{\text{row}}, \theta_{\text{col}})] - \varepsilon.$$

A similar requirement should hold for the column player.

We show that for some constant ε , determining whether a pure strategy ε -Bayes Nash equilibrium exists is NP-hard. Specifically, for $\varepsilon = 0.004$. We prove:

Theorem 6.1. *Let $\varepsilon = 0.004$. Then given a Bayesian game that admits a pure BNE, it is NP-hard to find a pure ε -BNE for the game. Moreover, it is NP-hard to solve the promise problem of distinguishing games that admit a pure BNE from games that do not admit a pure ε -BNE.*

Next, we show that when the distribution on types is uniform, a pure ε -BNE can be computed in quasi-polynomial time. As noted earlier, computing a pure BNE is NP-hard even in this special case [CS03].

Theorem 6.2. *In a two-player Bayesian game, suppose that the types are distributed uniformly on the space $\Theta_{\text{row}} \times \Theta_{\text{col}}$, and that $|\Theta_{\text{row}}| = |\Theta_{\text{col}}| = k$, and $|S_{\text{row}}| = |S_{\text{col}}| = n$. Assuming that a pure BNE exists, we can find a pure ε -BNE in time $n^{O((\log n + \log k)/\varepsilon^2)}$.*

Remark 6.3. The assumption in Theorem [6.2](#) can be relaxed to a pure $(\varepsilon/2)$ -BNE equilibrium existing (instead of an actual equilibrium).

As for other quasi-polynomial time computable approximate equilibria with NP-hard exact variants we've seen, this problem is also as hard as Hidden Clique:

Theorem 6.4. *For every $\eta > 0$, finding a $(\frac{1}{4} - \eta)$ -approximate pure BNE in a two-player Bayesian game with uniformly distributed types and in which a pure BNE exists is as hard as finding a hidden clique of size $C \log n$.*

References

- [ABC11] Austrin, P., Braverman, M., Chlamtác, E.: Inapproximability of NP-Complete Variants of Nash Equilibrium. arXiv:1104.3760 (2011)
- [AKS98] Alon, N., Krivelevich, M., Sudakov, B.: Finding a large hidden clique in a random graph. *Rand. Struct. Algos.* 13(3-4), 457–466 (1998)
- [BBM10] Bosse, H., Byrka, J., Markakis, E.: New algorithms for approximate nash equilibria in bimatrix games. *Theor. Comput. Sci.* 411(1), 164–173 (2010)
- [BV09] Brubaker, S.C., Vempala, S.S.: Random tensors and planted cliques. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX 2009. LNCS, vol. 5687, pp. 406–419. Springer, Heidelberg (2009)
- [CDT09] Chen, X., Deng, X., Teng, S.-H.: Settling the complexity of computing two-player nash equilibria. *J. ACM* 56(3) (2009)
- [CS03] Conitzer, V., Sandholm, T.: New complexity results about Nash equilibria. *Games and Economic Behavior* 63(2), 621–641 (2008)
- [DMP07] Daskalakis, C., Mehta, A., Papadimitriou, C.H.: Progress in approximate nash equilibria. In: ACM Conference on Electronic Commerce, pp. 355–358 (2007)
- [DMP09] Daskalakis, C., Mehta, A., Papadimitriou, C.H.: A note on approximate Nash equilibria. *Theor. Comput. Sci.* 410(17), 1581–1588 (2009)
- [FK03] Feige, U., Krauthgamer, R.: The probable value of the lovász–schrijver relaxations for maximum independent set. *SIAM J. Comput.* 32(2), 345–370 (2003)
- [FK08] Frieze, A.M., Kannan, R.: A new approach to the planted clique problem. In: FSTTCS, pp. 187–198 (2008)
- [FNS07] Feder, T., Nazerzadeh, H., Saberi, A.: Approximating Nash equilibria using small-support strategies. In: ACM Conference on Electronic Commerce, pp. 352–354 (2007)
- [FT91] Fudenberg, D., Tirole, J.: *Game Theory*. MIT Press, Cambridge (1991)
- [GZ89] Gilboa, I., Zemel, E.: Nash and correlated equilibria: Some complexity considerations. *Games and Econ. Behavior* 1(1), 80–93 (1989)

- [HK11] Hazan, E., Krauthgamer, R.: How Hard Is It to Approximate the Best Nash Equilibrium?. *SIAM J. Comput.* 40(1), 79–91 (2011)
- [LMM03] Lipton, R.J., Markakis, E., Mehta, A.: Playing large games using simple strategies. In: *ACM Conference on Electronic Commerce*, pp. 36–41 (2003)
- [MV09] Minder, L., Vilenchik, D.: Small clique detection and approximate nash equilibria. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) *APPROX 2009*. LNCS, vol. 5687, pp. 673–685. Springer, Heidelberg (2009)
- [Pap94] Papadimitriou, C.H.: On the complexity of the parity argument and other inefficient proofs of existence. *J. Comp. Sys. Sci.* 48(3), 498–532 (1994)
- [TS08] Tsaknakis, H., Spirakis, P.G.: An optimization approach for approximate nash equilibria. *Internet Mathematics* 5(4), 365–382 (2008)

Sparse Recovery with Partial Support Knowledge^{*}

Khanh Do Ba and Piotr Indyk

Massachusetts Institute of Technology, Cambridge MA 02139, USA

Abstract. The goal of sparse recovery is to recover the (approximately) best k -sparse approximation \hat{x} of an n -dimensional vector x from linear measurements Ax of x . We consider a variant of the problem which takes into account *partial knowledge* about the signal. In particular, we focus on the scenario where, after the measurements are taken, we are given a set S of size s that is supposed to contain most of the “large” coefficients of x . The goal is then to find \hat{x} such that

$$\|x - \hat{x}\|_p \leq C \min_{\substack{k\text{-sparse } x' \\ \text{supp}(x') \subseteq S}} \|x - x'\|_q . \quad (1)$$

We refer to this formulation as the *sparse recovery with partial support knowledge problem (SRPSK)*. We show that SRPSK can be solved, up to an approximation factor of $C = 1 + \epsilon$, using $O((k/\epsilon) \log(s/k))$ measurements, for $p = q = 2$. Moreover, this bound is tight as long as $s = O(\epsilon n / \log(n/\epsilon))$. This completely resolves the asymptotic measurement complexity of the problem except for a very small range of the parameter s .

To the best of our knowledge, this is the first variant of $(1 + \epsilon)$ -approximate sparse recovery for which the asymptotic measurement complexity has been determined.

1 Introduction

In recent years, a new “linear” approach for obtaining a succinct approximate representation of n -dimensional vectors (or signals) has been discovered. For any signal x , the representation is equal to Ax , where A is an $m \times n$ matrix, or possibly a random variable chosen from some distribution over such matrices. The vector Ax is often referred to as the *measurement vector* or *linear sketch* of x . Although m is typically much smaller than n , the sketch Ax often contains plenty of useful information about the signal x .

A particularly useful and well-studied problem is that of *stable sparse recovery*. We say that a vector x' is k -sparse if it has at most k non-zero coordinates.

^{*} This material is based upon work supported by the Space and Naval Warfare Systems Center Pacific under Contract No. N66001-11-C-4092, David and Lucille Packard Fellowship, MADALGO (Center for Massive Data Algorithmics, funded by the Danish National Research Association) and NSF grants CCF-0728645 and CCF-1065125.

The sparse recovery problem is typically defined as follows: for some norm parameters p and q and an approximation factor $C > 0$, given Ax , recover an “approximation” vector \hat{x} such that

$$\|x - \hat{x}\|_p \leq C \min_{k\text{-sparse } x'} \|x - x'\|_q \tag{2}$$

(this inequality is often referred to as ℓ_p/ℓ_q *guarantee*). If the matrix A is random, then (2) should hold for each x with some probability (say, 3/4). Sparse recovery has a tremendous number of applications in areas such as compressive sensing of signals [4,11], genetic data acquisition and analysis [23,3] and data stream algorithms [20,17].

It is known [4] that there exist matrices A and associated recovery algorithms that produce approximations \hat{x} satisfying (2) with $p = q = 1$, constant approximation factor C , and sketch length

$$m = O(k \log(n/k)) . \tag{3}$$

A similar bound, albeit using random matrices A , was later obtained for $p = q = 2$ [15] (building on [6,7,8]). Specifically, for $C = 1 + \epsilon$, they provide a distribution over matrices A with

$$m = O((k/\epsilon) \log(n/k)) \tag{4}$$

rows, together with an associated recovery algorithm.

It is also known that the bound in (3) is asymptotically optimal for some constant C and $p = q = 1$ (see [10] and [13], building on [14,16,18]). The bound of [10] also extends to the randomized case and $p = q = 2$. For $C = 1 + \epsilon$, a lower bound of (roughly) $m = \Omega(k/\epsilon^{p/2})$ was recently shown [22].

The necessity of the “extra” logarithmic factor multiplying k is quite unfortunate: the sketch length determines the “compression rate”, and for large n any logarithmic factor can worsen that rate tenfold.

In this paper we show that this extra factor can be reduced if we allow the recovery process to take into account some *partial knowledge* about the signal. In particular, we focus on the scenario where, after the measurements are taken, we are given a set S of size s (s is known beforehand) that is supposed to contain most of the “large” coefficients of x . The goal is then to find \hat{x} such that

$$\|x - \hat{x}\|_p \leq C \min_{\substack{k\text{-sparse } x' \\ \text{supp}(x') \subseteq S}} \|x - x'\|_q . \tag{5}$$

We refer to this formulation as the *sparse recovery with partial support knowledge problem (SRPSK)*.

Results. We show that SRPSK can be solved, up to an approximation factor of $C = 1 + \epsilon$, using $O((k/\epsilon) \log(s/k))$ measurements, for $p = q = 2$. Moreover, we show that this bound is tight as long as $s = O(en/\log(n/\epsilon))$. This completely resolves the asymptotic measurement complexity of the problem except for a very small range of the parameter s .

To the best of our knowledge, this is the first variant of $(1 + \epsilon)$ -approximate sparse recovery for which the asymptotic measurement complexity has been determined.

Motivation. The challenge of incorporating external knowledge into the sparse recovery process has received a fair amount of attention in recent years [9]. Approaches include *model-based compressive sensing* [20,12] (where the sets of large coefficients are known to exhibit some patterns), *Bayesian compressive sensing* [5] (where the signals are generated from a known distribution) and support restriction.

There are several scenarios where our formulation (SRPSK) could be applicable. For example, for tracking tasks, the object position typically does not change much between frames, so one can limit the search for current position to a small set. The framework can also be useful for exploratory tasks, where there is a collection \mathcal{S} of sets, one of which is assumed to contain the support. In that case, setting the probability of failure to $\frac{1}{|\mathcal{S}|}$ enables exploring all sets in the family and finding the one which yields the best approximation.

From a theoretical perspective, our results provide a smooth tradeoff between the $\Theta(k \log(n/k))$ bound for “standard” sparse recovery and the $\Theta(k)$ bound known for the *set query problem* [21]. In the latter problem we have the full knowledge of the signal support, i.e., $s = k$.

Our techniques. Consider the *upper bound* first. The general approach of our algorithm is to reduce SRPSK to the *noisy sparse recovery problem* (NSR). The latter is a generalization of sparse recovery where the recovery algorithm is given $Ax + \nu$, where ν is the *measurement noise*. The reduction proceeds by representing Ax as $Ax_S + Ax_{\bar{S}}$, and interpreting the second term as noise. Since the vector x_S has dimension s , not n , we can use A with only $O(k \log(s/k))$ rows. This yields the desired measurement bound.

To make this work, however, we need to ensure that for any fixed S , the sub-matrix A_S of A (containing the columns with indices in S) is a valid sparse recovery matrix for s -dimensional vectors. This would be immediate if (as often happens, e.g. [4]) each column of A was an i.i.d. random variable chosen from some distribution: we could simply sample the n columns of A from the distribution parametrized by k and s . Unfortunately, the algorithm of [15] (which has the best known dependence on ϵ) does not have this independence property; in fact, the columns are highly dependent on each other. However, we show that it is possible to modify it so that the independence property holds.

Our *lower bound* argument mimics the approach of [10]. Specifically, consider fixing $s = \Theta(\epsilon n / \log(n/\epsilon))$; we show how to encode $\alpha = \Theta(\log(n/\epsilon)/\epsilon)$ code words x_1, \dots, x_α , from some code C containing $2^{\Theta(k \log(s/k))}$ code words, into a vector x , such that a $(1 + \epsilon)$ -approximate algorithm for SRPSK can iteratively decode all x_i 's, starting from x_α and ending with x_1 . This shows that one can “pack” $\Theta(\log(n/\epsilon)/\epsilon \cdot k \log(s/k))$ bits into Ax . Since one can show that each coordinate of Ax yields only $O(\log(n/\epsilon))$ bits of information, it follows that Ax has to have $\Theta((k/\epsilon) \log(s/k))$ coordinates.

Unfortunately, the argument of [10] applied only to the case of when ϵ is a constant strictly greater than 0 (i.e., $\epsilon = \Omega(1)$). For $\epsilon = o(1)$, the recovery algorithm could return a convex combination of several x_i 's, which might not be decodable. Perhaps surprisingly, we show that the formulation of SRPSK avoids this problem. Intuitively, this is because different x_i 's have different supports, and SRPSK enables us to restrict sparse approximation to a particular subset of coordinates.

2 Preliminaries

For positive integer n , let $[n] = \{1, 2, \dots, n\}$. For positive integer $s \leq n$, let $\binom{[n]}{s}$ denote the set of subsets of cardinality s in $[n]$.

Let $v \in \mathbb{R}^n$. For any positive integer $k \leq s$ and set $S \in \binom{[n]}{s}$, denote by $v_k \in \mathbb{R}^n$ the vector comprising the k largest components of v , breaking ties by some canonical ordering (say, leftmost-first), and 0 everywhere else. Denote by $v_S \in \mathbb{R}^n$ the vector comprising of components of v indexed by S , with 0 everywhere else, and denote by $v_{S,k} \in \mathbb{R}^n$ the vector comprising the k largest components of v among those indexed by S , with 0 everywhere else.

Let $\Pi_S \in \mathbb{R}^{s \times n}$ denote the projection matrix that keeps only components indexed by S (the dimension n will be clear from context). In particular, $\Pi_S v \in \mathbb{R}^s$ consists of components of v indexed by S , and for any matrix $A \in \mathbb{R}^{m \times n}$, $A\Pi_S^T \in \mathbb{R}^{m \times s}$ consists of the columns of A indexed by S .

Define the ℓ_p/ℓ_p *sparse recovery with partial support knowledge problem* (denoted SRPSK _{p}) to be the following:

Given parameters (n, s, k, ϵ) , where $1 \leq k \leq s \leq n$ and $0 < \epsilon < 1$, design an algorithm and a distribution over matrices $A \in \mathbb{R}^{m \times n}$, where $m = m(n, s, k, \epsilon)$, such that for any $x \in \mathbb{R}^n$, the algorithm, given Ax and a specified set $S \in \binom{[n]}{s}$, recovers (with knowledge of A) a vector $\hat{x} \in \mathbb{R}^n$ such that, with probability $3/4$,

$$\|x - \hat{x}\|_p^p \leq (1 + \epsilon)\|x - x_{S,k}\|_p^p. \quad (6)$$

Define the ℓ_p/ℓ_p *noisy sparse recovery problem* (NSR _{p}) to be the following:

Given parameters (n, k, ϵ) , where $1 \leq k \leq n$ and $0 < \epsilon < 1$, design an algorithm and a distribution over matrices $A \in \mathbb{R}^{m \times n}$, where $m = m(n, k, \epsilon)$, such that for any $x \in \mathbb{R}^n$ and $\nu \in \mathbb{R}^m$, the algorithm recovers from $Ax + \nu$ (with knowledge of A) a vector $\hat{x} \in \mathbb{R}^n$ such that, with probability $3/4$,

$$\|x - \hat{x}\|_p^p \leq (1 + \epsilon)\|x - x_k\|_p^p + \epsilon\|\nu\|_p^p. \quad (7)$$

The distribution of A must be “normalized” so that for any $v \in \mathbb{R}^n$, $\mathbb{E}[\|Av\|_p] \leq \|v\|_p$.

For all four problems, we will denote a solution by a pair (A, \mathcal{R}) , where A is the measurement matrix and \mathcal{R} is the recovery algorithm. For SRPSK₁ and SRPSK₂, we will also often denote the recovery algorithm with the parameter S as a subscript, e.g., \mathcal{R}_S .

3 Lower Bounds

We will need a result from communication complexity. Consider the following two-party communication game involving Alice and Bob: Alice is given a string $y \in \{0, 1\}^d$. Bob is given an index $i \in [d]$, together with $y_{i+1}, y_{i+2}, \dots, y_d$. They also share an arbitrarily long common random string r . Alice sends a single message $M(y, r)$ to Bob, who must output y_i correctly with probability at least $3/4$, where the probability is taken over r . We refer to this problem as the *augmented indexing problem* (AIP). The communication cost of AIP is the minimum, over all correct protocols, of the length of the message $M(y, r)$ on the worst-case choice of r and y . The following lemma is well-known (see, e.g., [19] or [11]):

Lemma 1. *The communication cost of AIP is $\Omega(d)$.*

We will also make use of Lemma 5.1 of [10], which we reproduce below:

Lemma 2. *Consider any $m \times n$ matrix A with orthonormal rows. Let A' be the result of rounding A to b bits per entry. Then for any $v \in \mathbb{R}^n$ there exists a $\sigma \in \mathbb{R}^n$ with $A'v = A(v - \sigma)$ and $\|\sigma\|_1 < n^2 2^{-b} \|v\|_1$.*

Now we can prove our lower bounds for SRPSK₁ and SRPSK₂:

Theorem 3. *Any solution to SRPSK₁ requires, for $s = O(\epsilon n / \log(n/\epsilon))$, at least $\Omega((k/\epsilon) \log(s/k))$ measurements.*

Proof. For $\alpha = n/s$, divide $[n]$ into α equal-sized disjoint blocks, S_i for $i = 1, \dots, \alpha$. For each block S_i , we will choose a binary error-correcting code $C_i \subseteq \{0, 1\}^n$ with minimum Hamming distance k , where all the code words have Hamming weight exactly k and support contained in S_i . Since $|S_i| = s = n/\alpha$, we know each C_i can be chosen big enough that

$$\log |C_i| = \Theta(k \log(n/(\alpha k))) . \quad (8)$$

Now, we will use any solution to SRPSK₁ to design a protocol for AIP with instance size

$$d = \Theta(\alpha k \log(n/(\alpha k))) . \quad (9)$$

The protocol is as follows:

Alice divides her input y into α equal-sized blocks each of size

$$d/\alpha = \Theta(k \log(n/(\alpha k))) . \quad (10)$$

Interpreting each block y_i as a binary number, she uses it to index into C_i (notice that C_i has sufficiently many code words for each y_i to index a different one), specifying a code word $x_i \in C_i$. She then computes

$$x = D^1 x_1 + D^2 x_2 + \dots + D^\alpha x_\alpha \quad (11)$$

for some fixed D dependent on ϵ . Then, using shared randomness, and following the hypothetical protocol, Alice and Bob agree on a matrix A (wlog, and for

technical reasons, with orthonormal rows), which they both round to A' so that each entry has b bits. Alice computes $A'x$ and sends it to Bob.

Bob, knowing his input i , can compute the $j = j(i)$ for which block y_j of Alice's input contains i , and hence knows the set S_j . Moreover, he knows $y_{j'}$, and thereby $x_{j'}$, for every $j' > j$, so he can compute

$$z = D^{j+1}x_{j+1} + \cdots + D^\alpha x_\alpha . \quad (12)$$

From Alice's message, using linearity, he can then compute $A'(x - z)$. Now, by Lem. 2, there must exist some $\sigma \in \mathbb{R}^n$ with $A'(x - z) = A(x - z - \sigma)$ and

$$\|\sigma\|_1 < n^2 2^{-b} \|x - z\|_1 = n^2 2^{-b} \sum_{i'=1}^j k D^{i'} < n^2 2^{-b} k \frac{D^{j+1}}{D-1} . \quad (13)$$

Now, let $w = x - z - \sigma$, so that Bob has $Aw = A'(x - z)$. He then runs \mathcal{R}_{S_j} on Aw to recover \hat{w} with the properties that $\text{supp}(\hat{w}) \subseteq S_j$ and

$$\begin{aligned} \|w - \hat{w}\|_1 &\leq (1 + \epsilon) \|w - w_{S_j, k}\|_1 \leq (1 + \epsilon) \|w - D^j x_j\|_1 \\ &\leq (1 + \epsilon) (\|D^1 x_1 + \cdots + D^{j-1} x_{j-1}\|_1 + \|\sigma\|_1) \\ &= (1 + \epsilon) \left(k \frac{D^j - D}{D-1} + \|\sigma\|_1 \right) . \end{aligned} \quad (14)$$

Bob then finds the code word in C_j that is closest in ℓ_1 -distance to \hat{w}/D^j (which he hopes is x_j) and, looking at the index of that code word within C_j (which he hopes is y_j), he returns the bit corresponding to his index i .

Now, suppose that Bob was wrong. This means he obtained a \hat{w} that, appropriately scaled, was closer or equidistant to another code word in C_j than x_j , implying that $\|x_j - \hat{w}/D^j\|_1 \geq k/2$. Since $\text{supp}(\hat{w}) \subseteq S_j$, we can write

$$\begin{aligned} \|w - \hat{w}\|_1 &\geq \|x - z - \hat{w}\|_1 - \|\sigma\|_1 \\ &= \|D^1 x_1 + \cdots + D^{j-1} x_{j-1}\|_1 + D^j \|x_j - \hat{w}/D^j\|_1 - \|\sigma\|_1 \\ &\geq k \left(\frac{D^j - D}{D-1} + D^j/2 \right) - \|\sigma\|_1 . \end{aligned} \quad (15)$$

We will show that for appropriate choices of D and b , (14) and (15) contradict each other, implying that Bob must have correctly extracted his bit and solved AIP. To this end, it suffices to prove the following inequality:

$$\|\sigma\|_1 < \frac{k}{3} \left(D^j/2 - \epsilon \frac{D^j}{D-1} \right) , \quad (16)$$

where we used the fact that $\epsilon < 1$. Now, let us fix $D = 1 + 4\epsilon$. The above inequality becomes

$$\|\sigma\|_1 < \frac{k}{3} \left((1 + 4\epsilon)^j/2 - (1 + 4\epsilon)^j/4 \right) = k(1 + 4\epsilon)^j/12 . \quad (17)$$

Now, from (13) we know that

$$\|\sigma\|_1 < n^2 2^{-b} k \frac{D^{j+1}}{D-1} = n^2 2^{-b} k (1 + 4\epsilon)^{j+1}/(4\epsilon) , \quad (18)$$

so we need only choose b large enough that $2^b \geq 15n^2/\epsilon$, i.e., $b = O(\log(n/\epsilon))$ suffices. Recall that b is the number of bits per component of A' , and each component of $x-z$ can require up to $\alpha \log D = O(\epsilon\alpha)$ bits, so the message $A'(x-z)$ which Alice sends to Bob contains at most $O(m(b+\epsilon\alpha)) = O(m(\log(n/\epsilon)+\epsilon\alpha))$ bits, with which they solve AIP with $d = \Theta(\alpha k \log(n/(\alpha k)))$. It follows from Lem. [6](#) that

$$m = \Omega\left(\frac{\alpha k \log(n/(\alpha k))}{\log(n/\epsilon) + \epsilon\alpha}\right). \quad (19)$$

Finally, as long as $\epsilon\alpha = \Omega(\log(n/\epsilon))$, or equivalently, $s = n/\alpha = O(\epsilon n/\log(n/\epsilon))$, this simplifies to

$$m = \Omega((k/\epsilon) \log(s/k)). \quad (20)$$

□

Theorem 4. *Any solution to SRPSK₂ requires, for $s = O(\epsilon n/\log(n/\epsilon))$ and $\epsilon \leq 1/6$ [1](#), requires at least $\Omega((k/\epsilon) \log(s/k))$ measurements.*

We omit the proof, which involves only algebraic modifications from the proof of Thm. [3](#), due to space constraints.

4 Upper Bounds

First we prove a general black box reduction from SRPSK₁ to NSR₁ that works if the solution to NSR₁ has certain additional properties:

Lemma 5. *Suppose we have a solution to NSR₁ with parameters (n, k, ϵ) , where the $m \times n$ measurement matrix A' has $m = m(n, k, \epsilon)$ rows. Suppose in addition that the columns of A' are generated i.i.d. from some distribution. Then there exists a solution (A, \mathcal{R}) to SRPSK₁ with parameters (n, s, k, ϵ) that uses $O(m(s, k, \Theta(\epsilon)))$ measurements. Moreover, if A' has, in expectation, $h(n, k, \epsilon)$ non-zeros per column, and the NSR₁ recovery time is $t(n, k, \epsilon)$, then A has, in expectation, $O(h(s, k, \Theta(\epsilon)))$ non-zeros, and \mathcal{R} runs in $O(t(s, k, \Theta(\epsilon)))$ time[2](#).*

Proof. We construct our solution (A, \mathcal{R}) to SRPSK₁ as follows:

1. Let $\delta > 0$ be a constant to be specified later. Consider an instantiation of the solution to NSR₁ with parameters $(s, k, \delta\epsilon)$, so that its measurement matrix A' is $m \times s$, where $m = m(s, k, \delta\epsilon)$. Generate the n columns of our $m \times n$ measurement matrix A i.i.d. from the same distribution used to generate the i.i.d. columns of A' (note that the number of rows m is the same for both A and A').

¹ The assumption that $\epsilon \leq 1/6$ is not necessary, but makes the proof simpler, so we leave it in the theorem statement.

² Note that this recovery time is based on the assumption that the solution to NSR generated the columns of its measurement matrix i.i.d. In our application of this reduction (Lems. [7](#) and [8](#)), we will need to modify the NSR solution to enforce this requirement, which will increase its recovery time.

2. Given $S \subseteq [n]$, $|S| = s$, let \mathcal{R}'_S denote the recovery algorithm for NSR₁ corresponding to the parameters $(s, k, \delta\epsilon)$ and given the matrix $A\Pi_S^T$ (recall that a recovery algorithm for NSR₁ is allowed to behave differently given different instances of the measurement matrix). Define our recovery procedure \mathcal{R}_S by $\mathcal{R}_S(y) = \Pi_S^T(\mathcal{R}'_S(y))$; in words, we run \mathcal{R}'_S on our m -dimensional measurement vector y to obtain an s -dimensional vector, which we embed into an n -dimensional vector at positions corresponding to S , filling the rest with zeros.

Note that the number of non-zeros per column of A and the running time of \mathcal{R} follow immediately.

Observe that, thanks to the independence of the columns of A , the submatrix comprised of any s of them (namely, $A\Pi_S^T$) is a valid $m \times s$ measurement matrix. Thus we have the guarantee that for any signal $x' \in \mathbb{R}^s$ and noise vector $\nu \in \mathbb{R}^m$, \mathcal{R}'_S recovers from $A\Pi_S^T x' + \nu$ a vector $\hat{x}' \in \mathbb{R}^s$ satisfying, with probability $3/4$,

$$\|x' - \hat{x}'\|_1 \leq (1 + \epsilon)\|x' - x'_k\|_1 + \delta\epsilon\|\nu\|_1 . \quad (21)$$

Now, let $x \in \mathbb{R}^n$ be our signal for SRPSK₁. We interpret $\Pi_S x \in \mathbb{R}^s$ to be the sparse signal and $Ax_{\bar{S}} \in \mathbb{R}^m$ to be the noise, so that running \mathcal{R}'_S on $A\Pi_S^T(\Pi_S x) + Ax_{\bar{S}}$ returns $\hat{x}' \in \mathbb{R}^s$ satisfying, with probability $3/4$,

$$\begin{aligned} \|\Pi_S x - \hat{x}'\|_1 &\leq (1 + \epsilon)\|\Pi_S x - (\Pi_S x)_k\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 \\ &= (1 + \epsilon)\|x_S - x_{S,k}\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 . \end{aligned} \quad (22)$$

Finally, consider the $\hat{x} \in \mathbb{R}^n$ recovered by \mathcal{R}_S in our procedure for SRPSK₁ when run on

$$Ax = Ax_S + Ax_{\bar{S}} = A\Pi_S^T(\Pi_S x) + Ax_{\bar{S}} . \quad (23)$$

We have $\hat{x} = \Pi_S^T \hat{x}'$, or, equivalently, $\Pi_S \hat{x} = \hat{x}'$, so

$$\begin{aligned} \|x - \hat{x}\|_1 &= \|x_{\bar{S}}\|_1 + \|x_S - \hat{x}\|_1 = \|x_{\bar{S}}\|_1 + \|\Pi_S x - \hat{x}'\|_1 \\ &\leq \|x_{\bar{S}}\|_1 + (1 + \epsilon)\|x_S - x_{S,k}\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 \\ &= \|x_{\bar{S}}\|_1 + (1 + \epsilon)(\|x - x_{S,k}\|_1 - \|x_{\bar{S}}\|_1) + \delta\epsilon\|Ax_{\bar{S}}\|_1 \\ &= (1 + \epsilon)\|x - x_{S,k}\|_1 - \epsilon\|x_{\bar{S}}\|_1 + \delta\epsilon\|Ax_{\bar{S}}\|_1 . \end{aligned}$$

Thus, if we can ensure that $\|Ax_{\bar{S}}\|_1 \leq (1/\delta)\|x_{\bar{S}}\|_1$, we would obtain the desired guarantee for SRPSK₁ of

$$\|x - \hat{x}\|_1 \leq (1 + \epsilon)\|x - x_{S,k}\|_1 . \quad (24)$$

But we know that $\mathbb{E}[\|Ax_{\bar{S}}\|_1] \leq \|x_{\bar{S}}\|_1$, so by the Markov bound

$$\Pr[\|Ax_{\bar{S}}\|_1 > (1/\delta)\|x_{\bar{S}}\|_1] \leq \delta . \quad (25)$$

Choosing, say, $\delta = 1/12$ would give us an overall success probability of at least $2/3$, which can be amplified by independent repetitions and taking a component-wise median in the standard way. \square

Straightforward modification of the above proof yields the ℓ_2/ℓ_2 version:

Lemma 6. *Suppose we have a solution to NSR_2 with parameters (n, k, ϵ) , where the $m \times n$ measurement matrix A' has $m = m(n, k, \epsilon)$ rows. Suppose in addition that the columns of A' are generated i.i.d. from some distribution. Then there exists a solution (A, \mathcal{R}) to $SRPSK_2$ with parameters (n, s, k, ϵ) that uses $O(m(s, k, \Theta(\epsilon)))$ measurements. Moreover, if A' has, in expectation, $h(n, k, \epsilon)$ non-zeros per column, and the NSR_2 recovery time is $t(n, k, \epsilon)$, then A has, in expectation, $O(h(s, k, \Theta(\epsilon)))$ non-zeros, and \mathcal{R} runs in $O(t(s, k, \Theta(\epsilon)))$ time³.*

By a modification of the algorithm of [15], we prove the following result:

Lemma 7. *There exist a distribution on $m \times n$ matrices A and a collection of algorithms $\{\mathcal{R}_S \mid S \subseteq [n], |S| = s\}$ such that for any $x \in \mathbb{R}^n$ and set $S \subseteq [n]$, $|\mathcal{R}_S(Ax)$ recovers \hat{x} with the guarantee that*

$$\|x - \hat{x}\|_2 \leq (1 + \epsilon)\|x - x_{S,k}\|_2 \quad (26)$$

with probability $3/4$. The matrix A has $m = O((k/\epsilon) \log(s/k))$ rows.

Proof. To apply a NSR_2 solution to $SRPSK_2$ using Lem. 6, we need the columns of the measurement matrix to be generated independently. However, this requirement does not hold with the algorithm in [15] as is. Therefore, we show how to modify it to satisfy this requirement without changing its recovery properties and asymptotic number of measurements. For simplicity, we will ignore pseudo-randomness considerations, and replace all k -wise independence by full independence in the construction of [15].

We begin by describing the measurement matrix A of [15] (denoted by Φ in that paper). At the highest level, A is formed by vertically stacking matrices $A^{(j)}$, for $j = 1, \dots, \log k$. Each $A^{(j)}$ is formed by vertically stacking two matrices, $E^{(j)}$ and $D^{(j)}$. It will suffice for our purposes if the columns of each $E^{(j)}$ and each $D^{(j)}$ are independent.

Consider, first, $E^{(j)}$, which consists of several i.i.d. submatrices, again stacked vertically, in each of which every entry is set i.i.d. (to 1, -1 or 0). Thus, every entry, and therefore every column, of $E^{(j)}$ is already independent without modification.

Next, consider $D^{(j)}$, which consists of several similarly stacked i.i.d. submatrices. For some constant $c < 1$, each one of these submatrices consists of kc^j i.i.d. ‘‘blocks’’ $B_1^{(j)}, B_2^{(j)}, \dots, B_{kc^j}^{(j)}$, which will be the smallest unit of vertically stacked submatrices we need to consider (see Fig. 1). Within each block $B_i^{(j)}$, each column is independently chosen to be non-zero with some probability, and the i^{th} non-zero column is equal to the i^{th} code word w_i from some error-correcting code C . The code C has a constant rate and constant fractional distance. Therefore, each block has $O(\log h)$ rows (and C needs to have $O(h)$ code words), where h is the expected number of non-zero columns per block.

³ See footnote to Lem. 5

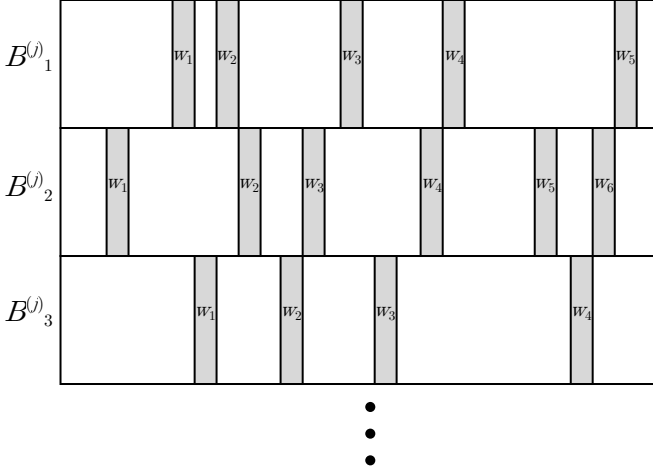


Fig. 1. Example of an i.i.d. submatrix in $D^{(j)}$ consisting of $k c^j$ blocks. Each grey rectangle represents a code word, and white space represents zeros.

The problem with the construction of $D^{(j)}$ (from our perspective) is that each column chosen to be non-zero is not independently chosen, but instead is determined by a code word that depends on how many non-zero columns are to its left. In order to overcome this obstacle, we observe that the algorithm of [15] only requires that the codewords of the consecutive non-zero columns are *distinct*, not *consecutive*. Thus, we use as ECC C' with the same rate and error-correction, but with $O(h^3)$ code words instead of $O(h)$; for each column chosen to be non-zero, we set it to a code word chosen uniformly at random from C' . In terms of Fig. 1, each grey rectangle, instead of being the code word from C specified in the figure, is instead a random code word from a larger code C' . Note that each block has still $O(\log h)$ rows as before.

A block is *good* if all codewords corresponding to it are distinct. Observe that for any given block, the probability it is not good is at most $O(1/h)$. If there are fewer than $O(h)$ blocks in all of $D^{(j)}$, we could take a union bound over all of them to show that all blocks are good constant probability. Unfortunately, for $j = 1$, we have $h = O(n/k)$ while the number of blocks is $\Omega(k)$. The latter value could be much larger than h .

Instead, we will simply double the number of blocks. Even though we cannot guarantee that all blocks are good, we know that most of them will be, since each one is with probability $1 - O(1/h)$. Specifically, by the Chernoff bound, at least half of them will be with high probability (namely, $1 - e^{-\Omega(k)}$). We can use only those *good* blocks during recovery and still have sufficiently many of them to work with.

The result is a solution to NSR_2 still with $O((k/\epsilon) \log(n/k))$ rows (roughly 6 times the solution of [15]), but where each column of the measurement matrix

is independent, as required by Lem. 6. A direct application of the lemma gives us the theorem. \square

Lemma 8. *The matrix A of Lem. 7 has, in expectation, $O(\log^2 k \log(s/k))$ non-zeros per column, and each algorithm \mathcal{R}_S runs in $O(s \log^2 k + (k/\epsilon) \log^{O(1)} s)$ time.*

Proof. It suffices to show that the modifications we made to [15] do not change the asymptotic expected number of non-zeros in each column and does not increase the recovery time by more than an additive term of $O(n \log^2 k)$. Lem. 6 then gives us this lemma (by replacing n with s in both quantities).

Consider, first, the number of non-zeros. In both the unmodified and the modified matrices, this is dominated by the number of non-zeros in the (mostly dense) code words in the D^j 's. But in the modified D^j , we do not change the asymptotic length of each code word, while only doubling, in expectation, the number of code words (in each column as well as overall). Thus the expected number of non-zeros per column of A remains $O(\log^2 k \log(n/k))$ as claimed.

Next, consider the running time. The first of our modifications, namely, increasing the number of code words from $O(h)$ to $O(h^3)$, and hence their lengths by a constant factor, does not change the asymptotic running time since we can use the same encoding and decoding functions (it suffices that these be polynomial time, while they are in fact poly-logarithmic time). The second of our modifications, namely, doubling the number of blocks, involves a little additional work to identify the *good* blocks at recovery time. Observe that, for each block, we can detect any collision in time linear in the number of code words. In $D^{(j)}$ there are $O(jkc^j)$ blocks each containing $O(n/(kc^j))$ code words, so the time to process $D^{(j)}$ is $O(jn)$. Thus, overall, for $j = 1, \dots, \log k$, it takes $O(n \log^2 k)$ time to identify all *good* blocks. After that, we need only work with the same number of blocks as there had been in the unmodified matrix, so the overall running time is $O(n \log^2 k + (k/\epsilon) \log^{O(1)} n)$ as required. \square

For completeness, we state the section's main result:

Theorem 9. *There exist a distribution on $m \times n$ matrices A and a collection of algorithms $\{\mathcal{R}_S \mid S \in \binom{[n]}{s}\}$ such that for any $x \in \mathbb{R}^n$ and set $S \subseteq [n]$, $|S| = s$, $\mathcal{R}_S(Ax)$ recovers \hat{x} with the guarantee that*

$$\|x - \hat{x}\|_2 \leq (1 + \epsilon) \|x - x_{S,k}\|_2 \quad (27)$$

with probability $3/4$. The matrix A has $m = O((k/\epsilon) \log(s/k))$ rows and, in expectation, $O(\log^2 k \log(s/k))$ non-zeros per column. Each algorithm \mathcal{R}_S runs in $O(s \log^2 k + (k/\epsilon) \log^{O(1)} s)$ time.

References

1. Bar-Yossef, Z., Jayram, T.S., Krauthgamer, R., Kumar, R.: The sketching complexity of pattern matching. In: Jansen, K., Khanna, S., Rolim, J.D.P., Ron, D. (eds.) RANDOM 2004 and APPROX 2004. LNCS, vol. 3122, pp. 261–272. Springer, Heidelberg (2004)

2. Baraniuk, R.G., Cevher, V., Duarte, M.F., Hegde, C.: Model-based compressive sensing. *IEEE Transactions on Information Theory* 56(4), 1982–2001 (2010)
3. Bruex, A., Gilbert, A., Kainkaryam, R., Schiefelbein, J., Woolf, P.: Poolmc: Smart pooling of mRNA samples in microarray experiments. *BMC Bioinformatics* (2010)
4. Candès, E.J., Romberg, J., Tao, T.: Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.* 59(8), 1208–1223 (2006)
5. Cevher, V., Indyk, P., Carin, L., Baraniuk, R.G.: Sparse signal recovery and acquisition with graphical models. *Signal Processing Magazine*, 92 – 103 (2010)
6. Charikar, M., Chen, K., Farach-Colton, M.: Finding frequent items in data streams. In: Widmayer, P., Triguero, F., Morales, R., Hennessy, M., Eidenbenz, S., Conejo, R. (eds.) *ICALP 2002*. LNCS, vol. 2380, p. 693. Springer, Heidelberg (2002)
7. Cormode, G., Muthukrishnan, S.M.: An improved data stream summary: The count-min sketch and its applications. In: Farach-Colton, M. (ed.) *LATIN 2004*. LNCS, vol. 2976, pp. 29–38. Springer, Heidelberg (2004)
8. Cormode, G., Muthukrishnan, S.: Combinatorial algorithms for Compressed Sensing. In: *Proc. 40th Ann. Conf. Information Sciences and Systems*, Princeton (March 2006)
9. Defense Sciences Office. Knowledge enhanced compressive measurement. Broad Agency Announcement, DARPA-BAA-10-38 (2010)
10. Do Ba, K., Indyk, P., Price, E., Woodruff, D.: Lower bounds for sparse recovery. In: *SODA* (2010)
11. Donoho, D.L.: Compressed Sensing. *IEEE Trans. Info. Theory* 52(4), 1289–1306 (2006)
12. Eldar, Y.C., Bolcskei, H.: Block-sparsity: Coherence and efficient recovery. In: *IEEE Int. Conf. Acoustics, Speech and Signal Processing* (2009)
13. Foucart, S., Pajor, A., Rauhut, H., Ullrich, T.: The gelfand widths of l_p -balls for $0 < p \leq 1$. *J. Complexity* (2010)
14. Garnaev, A.Y., Gluskin, E.D.: On widths of the euclidean ball. *Sov. Math., Dokl.*, 200–204 (1984)
15. Gilbert, A.C., Li, Y., Porat, E., Strauss, M.J.: Approximate sparse recovery: optimizing time and measurements. In: *STOC*, pp. 475–484 (2010)
16. Gluskin, E.D.: Norms of random matrices and widths of finite-dimensional sets. *Math. USSR-Sb.* 48, 173–182 (1984)
17. Indyk, P.: Sketching, streaming and sublinear-space algorithms. Graduate course notes (2007), <http://stellar.mit.edu/S/course/6/fa07/6.895/>
18. Kashin, B.S.: Diameters of some finite-dimensional sets and classes of smooth functions. *Math. USSR, Izv.* 11, 317–333 (1977)
19. Milterson, P.B., Nisan, N., Safra, S., Wigderson, A.: On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.* 57(1), 37–49 (1998)
20. Muthukrishnan, S.: *Data streams: Algorithms and applications*. Foundations and Trends in Theoretical Computer Science (2005)
21. Price, E.: Efficient sketches for the set query problem. In: *SODA* (2011)
22. Price, E., Woodruff, D. $(1 + \epsilon)$ -approximate sparse recovery. Preprint (2011)
23. Shental, N., Amir, A.: Or Zuk. Identification of rare alleles and their carriers using compressed se(que)nsing. *Nucleic Acids Research* 38(19), 1–22 (2010)

On Capacitated Set Cover Problems

Nikhil Bansal¹, Ravishankar Krishnaswamy^{2,*}, and Barna Saha^{3,*}

¹ IBM T. J. Watson Research Center, Yorktown Heights, NY 10598, USA

² Department of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA

³ Computer Science Department, University of Maryland College Park,
College Park, MD 20740, USA

Abstract. Recently, Chakrabarty et al. [5] initiated a systematic study of capacitated set cover problems, and considered the question of how their approximability relates to that of the uncapacitated problem on the same underlying set system. Here, we investigate this connection further and give several results, both positive and negative. In particular, we show that if the underlying set system satisfies a certain *hereditary property*, then the approximability of the capacitated problem is closely related to that of the uncapacitated version. We also give related lower bounds, and show that the hereditary property is necessary to obtain non-trivial results. Finally, we give some results for capacitated covering problems on set systems with low hereditary discrepancy and low VC dimension.

1 Introduction

In this paper, we consider the approximability of *capacitated* set cover problems (CSC). In a typical (uncapacitated) set cover instance, we are given a universe X of n elements and a collection \mathcal{S} of m subsets of X , each subset with an associated cost; the goal is to pick the collection of sets $\mathcal{S}' \subseteq \mathcal{S}$ of least total cost, such that each element $e \in X$ is contained in at least one set $S \in \mathcal{S}'$. It is well known that the greedy algorithm for set cover achieves an approximation ratio of $\ln n$, and that in general this approximation factor cannot be improved up to lower order terms [9]. However, in several cases of interest, improved approximation guarantees or even exact algorithms can be obtained. Typical examples are problems arising in network design where the underlying set system may be totally unimodular or have other interesting structural properties [12], or in geometric settings where the set system may have low structural complexity, often measured in terms of VC dimension [3] or union complexity [13]. In general, the study of covering problems is an extensive area of research in both combinatorial optimization and algorithms.

In the capacitated version of the set cover problem, the elements additionally have *demands* $d : X \rightarrow \mathbb{R}^+$, sets have *supplies* $s : \mathcal{S} \rightarrow \mathbb{R}^+$, and the goal is to find a minimum cost collection of sets \mathcal{S}' such that for each element e , the total supply of sets in \mathcal{S}' that cover e is at least $d(e)$. A general CSC is defined by the following integer linear program:

* Work done while visiting IBM T.J. Watson Research Yorktown Heights.

$$\begin{aligned} \text{CSC}(A, d, s, c) \quad \min \quad & \sum_{i=1}^m c_i x_i & (1) \\ \text{s.t.} \quad & \sum_{i=1}^m A_{ij} s_i x_i \geq d_j \quad \forall 1 \leq j \leq n & (2) \\ & x_i \in \{0, 1\} \quad \forall S_i \in \mathcal{S} & (3) \end{aligned}$$

Here, s_i denotes the supply of set S_i , d_j is the demand of element $j \in [n]$, and A is the $\{0, 1\}$ incidence matrix of the set system. The variable x_i indicates whether S_i is chosen or not, and hence the constraints ensure that for each element $j \in [n]$, the total supply of sets containing it is at least d_j .

Capacitated covering problems arise naturally in a variety of scenarios. For example, consider the minimum Steiner tree problem where the goal is to find the minimum-cost subgraph connecting terminals to a root. This can be cast as a set cover problem, viewing each graph cut separating some terminal from the root as an element, and each edge in the graph as a set (that covers every cut that it crosses). Now, if the terminals have a bandwidth requirement, and the edges have different bandwidth capacities, this corresponds to a capacitated covering problem. Similar generalizations naturally arise for most uncapacitated covering problems. Capacitated covering problems also arise indirectly as subroutines in other problems. For example, Bansal and Pruhs [11] showed that the scheduling problem of minimizing arbitrary functions of flow time on a single machine is equivalent (up to $O(1)$ factors) to the capacitated version of the geometric set cover problem of covering points in \mathbb{R}^2 using axis-aligned rectangles all of which touch the x -axis.

While capacitated covering problems have been studied previously, Chakrabarty et al. [5] recently initiated a more systematic study of these problems. Motivated by the extensive existing works on the uncapacitated set cover problem, they considered the following natural question. *Is there a relationship between the approximability of a capacitated set cover problem and the uncapacitated problem on the same underlying set system?* In particular, is it possible to exploit the combinatorial structure of the underlying incidence matrix in the set cover problem to design good algorithms for the capacitated case?

To understand this question better, it is instructive to even consider the case of the simplest possible set system: that with a *single* element. In this case, the problem reduces to precisely the so-called Knapsack Cover problem, where given a knapsack (element) of demand B and items (sets) with supplies s_1, \dots, s_m and costs c_1, \dots, c_m , the goal is to find a minimum cost collection of items that covers the knapsack. Already here, it turns out that the natural LP relaxation¹ of the integer program (1)-(3) has arbitrarily large integrality gap². In a celebrated result, Carr et al. [4] showed that this natural LP can be strengthened by adding exponentially many so-called Knapsack Cover (KC) inequalities. These inequalities can be separated in polynomial time and hence the LP can be solved efficiently to within any accuracy using the Ellipsoid method. The integrality gap of this strengthened LP reduces to 2, and this is also tight. We remark that there is also a local ratio based interpretation of these KC inequalities [2].

¹ Where we replace the $x \in \{0, 1\}$ in the IP by $x \in [0, 1]$.

² Consider an instance with two items of size $B - 1$ each and costs 0 and 1 respectively. Clearly any integral solution must choose both items, incurring a cost of 1. The LP can however choose the 0 cost item completely, and cost 1 item to extent $1/(B - 1)$, incurring a cost of $1/(B - 1)$.

Interestingly, Chakrabarty et al. [5] showed (see Theorem 1 for a formal statement) that given any CSC problem, the natural LP relaxation strengthened by adding KC inequalities for each element has an integrality gap that is no worse (up to $O(1)$ factors) than the integrality gap for two related *uncapacitated* problems. The first of these problems is simply the multi-cover problem on the same set system A , and the second one is the so-called *priority* set cover version of A , that we next define. Thus, roughly speaking their result shows that KC inequalities allow us to forget about capacities, at the expense of somewhat complicating the underlying set system.

Priority Covering Problems: Given a set cover instance specified by the incidence matrix A (the representation could be implicit as in network design problems), a *priority* version of the covering problem (PSC) is defined as follows. The elements and sets have priorities $\pi : X \cup \mathcal{S} \rightarrow \mathbb{Z}^+$. The goal is to pick a minimum cost collection of sets \mathcal{S}' such that for each element j , there is at least one set $S_i \in \mathcal{S}'$ containing j and with priority at least that of j , i.e., $\pi(S_i) \geq \pi(e_j)$.

The natural integer programming formulation for PSC is:

$$\begin{aligned} \text{PSC}(A, \pi, c) \text{ minimize } & \sum_{i=1}^m c_i x_i \\ \text{subject to } & (1) \sum_{i=1}^m A_{ij} \mathbf{1}_{(\pi(S_i) \geq \pi(e_j))} x_i \geq 1 \quad \forall 1 \leq j \leq n \\ & (2) \quad x_i \in \{0, 1\} \quad \forall S_i \in \mathcal{S} \end{aligned}$$

Here, $\mathbf{1}_{(a \geq b)}$ is the indicator variable for the condition inside (i.e., 1 if $a \geq b$ and 0 otherwise). Thus a priority cover problem is an (uncapacitated) set cover problem, with the incidence matrix $B_{ij} = A_{ij} \cdot \mathbf{1}_{(\pi_i \geq \pi_j)}$ instead of A . The structure of B has an interesting geometric connection to that of A . In particular, permute the columns of A in non-decreasing order of supply priorities and rows of A in non-decreasing order of demand priorities. Then, the priority matrix Π defined as $\Pi_{ij} = \mathbf{1}_{(\pi(S_i) \geq \pi(e_j))}$ has a “stair-case” structure of 1’s (see Figure 1 for an illustration), and $B = A \wedge \Pi$ is the element-wise product of A and Π . The number of stairs in Π is equal to number of distinct priorities k (which plays an important role in our results later).

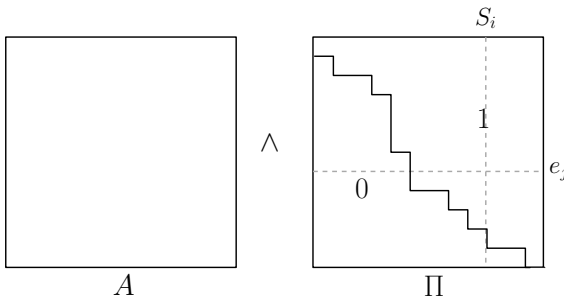


Fig. 1. A is an arbitrary $\{0, 1\}$ matrix, $\Pi_{ij} = 1$ if $\pi(S_i) \geq \pi(e_j)$, and $B = A \wedge \Pi$

Formally, Chakrabarty et al. [5] showed the following result.

Theorem 1 ([5]). *Let $\text{CSC}(A, d, s, c)$ be a capacitated set cover problem instance. Let $\text{MSC}(A, d', \mathbf{1}, c)$ denote the uncapacitated multi-cover³ problem with incidence matrix A and covering requirements d' , and let $\text{PSC}(A, \pi, c)$ denote the priority covering problem with incidence matrix A and priorities π . If*

1. *The integrality gap of the natural LP relaxation of $\text{MSC}(A, d', \mathbf{1}, c)$ is at most α for all possible covering requirements d' , and*
2. *The integrality gap of the priority problem $\text{PSC}(A, \pi, c)$ is at most β for all priority functions π ,*

Then the integrality gap of the LP relaxation of the capacitated problem $\text{CSC}(A, d, s, c)$ strengthened by KC inequalities is $O(\alpha + \beta)$. Moreover, the number of distinct priorities in the instance $\text{PSC}(A, \pi, c)$ is at most $\log s_{\max}$ where $s_{\max} = \max_{i \in [m]} s_i$ denotes the maximum supply of a set.

Here, as usual, we say that a problem has integrality gap α if for every feasible fractional solution x , there is a feasible integer solution \tilde{x} with cost at most α times the fractional cost. Also note that in the priority cover problem, only the relative values of priorities matter, hence we can assume that the priorities are always integers $1, \dots, k$.

In light of Theorem 1 it suffices to bound the integrality gap of the multi-cover version and priority cover version of the underlying set cover problem. Typically, the multi-cover version is not much harder than the set cover problem itself (e.g. if the matrix is totally unimodular, for various geometric systems, and so on), and the hard work lies in analyzing the priority problem.

We note here that a converse of Theorem 1 also holds in the sense that a capacitated problem is at least as hard as the priority problem. In particular, given any priority cover instance $\text{PSC}(A, \pi, c)$ with k priorities, consider the capacitated instance $\text{CSC}(A, d, s, c)$ where each element j with priority p has demand $d_j = m^{2p}$ and a set i with priority p has supply m^{2p} , where m is the number of sets in A . It can be easily verified that a collection of sets is feasible for CSC if and only if it is feasible for PSC .

1.1 Our Results

Given a set system (X, \mathcal{S}) with incidence matrix A , we will relate the integrality gap of a priority cover problem on A to the integrality gap of the set cover problem on A . We need the following additional definition.

Definition 1 (Hereditary Integrality Gap). *A set system (X, \mathcal{S}) with incidence matrix A has hereditary integrality gap α if the integrality gap for the natural LP relaxation of the set cover instance (A, c) restricted to any sub-system (X', \mathcal{S}) , where $X' \subseteq X$, is at most α .*

That is, the integrality gap is at most α if we restrict the system to any subset of elements⁴. Clearly, solving separately for demands in each of the k priority classes, the

³ By multi-cover we mean the usual generalization of standard set cover where an element j may wish to be covered by d_j distinct sets, instead of just one.

⁴ We note that this definition also allows the restriction of the system to $\mathcal{S}' \subseteq \mathcal{S}$, by considering the integrality gap on fractional solutions with support \mathcal{S}' .

integrality gap for any PSC instance is at most k times the hereditary integrality gap. We show that this can be improved substantially.

Theorem 2. *The integrality gap of any instance $\text{PSC}(A, \pi, c)$ with k priorities is $O(\alpha \log^2 k)$, where α is the hereditary integrality gap of the corresponding set cover instance (A, c) .*

According to Theorem 1, given any capacitated instance, the number of priorities in the associated priority instance is $k = O(\log s_{\max})$, and hence Theorem 2 implies that having capacities increases the hereditary integrality gap by at most $O((\log \log s_{\max})^2)$ (provided the multi-cover problem is also well-behaved w.r.t to the integrality gap).

Theorem 2 is proved in section 2 and its proof is surprisingly simple. However, this general result already achieves guarantees close to those known for very special systems. For example, for the previously mentioned CSC problem of covering points with rectangles touching the x -axis [1], a $O(\log k)$ guarantee for k priorities was obtained only recently using breakthrough geometric techniques of [13]. Using theorem 2 instead of the results of [13] already yields major improvements over previous results for the problem studied in [1].

Another corollary of Theorem 2 is that if A is *totally unimodular* (TU), then there this is an $O((\log \log s_{\max})^2)$ approximation for any capacitated problem on A . This follows as a TU matrix has a hereditary integrality gap of 1 for the multi-cover problem. This motivates our next result for set systems with low *hereditary discrepancy* (see Section 3 for a definition). Recall that TU matrices have a hereditary discrepancy of 1 (see e.g., [12]). Low hereditary discrepancy set systems arise naturally when the underlying system is a union of TU or other simpler systems. Recently, [8] also gave a surprising connection between low discrepancy and bin packing.

Theorem 3. *For any set system A where the dual set system A^T has hereditary discrepancy α , the integrality gap of the multi-cover instance $\text{MSC}(A, d, \mathbf{1}, c)$ for any demands d is α .*

As stated earlier, this implies an $O(\alpha(\log \log s_{\max})^2)$ integrality gap for any instance $\text{CSC}(A, d, s, c)$. Note that the integrality gap we show for the multi-cover problem is exactly α (and not just $O(\alpha)$), and hence this strictly generalizes the TU property, which results in an integral polytope.

The priority covering framework is particularly useful in geometric settings⁵. Appealing to this connection, our next result relates the VC dimension of the priority version of a problem to the original system. This is useful as low VC dimension can be exploited to obtain good set cover guarantees [3].⁶

⁵ Given a geometric set cover problem, its priority version can be encoded as another geometric problem (this increases the underlying dimension by 1). By adding a new dimension to encode priority, replace sets S_i with priority p by the geometric object $S_i \times [0, p]$ and points p_j with priority q by point $p_j \times [0, q]$. It is easily checked that the set cover problem on this instance is equivalent to the priority cover problem on the original instance. This observation was used in [15], and we do not elaborate more on it here.

⁶ Note that we need to bound the VC dimension of the dual set system A^T to obtain guarantees for the set cover instance A .

Theorem 4. *For any set system with VC dimension d , the VC dimension of its priority version (for any setting of priorities) is at most $d + 1$.*

Lower Bounds. In light of the above results, two natural questions arise. First, can similar guarantees for capacitated version be obtained without any hereditary assumption, that is w.r.t to the integrality gap alone? Second, is the loss of factor $O(\log^2 k)$ in the guarantees necessary?

For the first question, we note that there are natural problems such as priority Steiner tree [6,7], where the underlying set system is not hereditary, and the LP for the priority version has an integrality gap of $\Omega(k)$.

For the second question, in Section 5 we show that even for hereditary set systems, there are instances where the priority version has an integrality gap of $\Omega(\alpha \log k)$ when the original set cover problem has hereditary integrality gap of at most α .

Theorem 5. *There exist hereditary set cover instances with $O(1)$ -integrality gap for which the priority version has an integrality gap of $\Omega(\log k)$.*

These gap instances rely on the recent breakthrough constructions of Pach and Tardos [11] for geometric set systems with large ϵ -nets. In particular, we show that the gap already holds for the rectangle cover problem considered in [11] which we mentioned earlier. This shows that Theorem 2 is tight up to a $O(\log k)$ factor. Closing this gap would be an interesting question to study.

1.2 Other Related Work

Besides the work of Chakrabarty et al. [5] mentioned above, a work in spirit similar of ours is that of Kolliopoulos [10]. They studied the relationship between the approximability of a CSC and its corresponding set cover problem under the *no-bottleneck* assumption: this states that “the supply of every set/column is smaller than the demand of every element/row” (i.e. maximum supply is no more than minimum demand). Under this assumption, they show if the $x \leq 1$ constraint (or $x \leq d$ in general) can be violated by a constant multiplicative factor, then the integrality gap of any CSC is within an $O(1)$ factor of the corresponding $\{0, 1\}$ -CIP. However, nothing better than the standard set cover guarantee was known even with the no-bottleneck assumption. We refer the reader to [5] for further discussions on related work.

2 Bounding the Integrality Gap of PSC’s

In this section, we prove Theorem 2. We show that the integrality gap of PSC instances that are characterized by hereditary set systems is $O(\alpha \log^2 k)$, where k is the number of priorities. Recall the stair-case structure of Π and the definition of $B = A \wedge \Pi$.

The idea is rather simple. We decompose the incidence matrix B of the PSC instance into a collection of submatrices $\{D_0, \dots, D_\ell\}$ with the following properties.

⁷ N is submatrix of M if N is obtained by restricting M to a subset of rows and columns.

1. Each such submatrix D_q is also a submatrix of A (and not just that of B).
2. Each element j appears as a row in at most $O(\log k)$ of the submatrices D_q and,
3. Each set i appears as a column in at most $O(\log k)$ of the submatrices D_q .

As A has a hereditary integrality gap of α , by the first property above, any fractional set cover solution restricted to the sub-system D_q has an integrality gap of α . Now, any fractional set cover solution x on A induces a fractional solution on D_q (after appropriate scaling). We use the second and the third properties stated above to show that the rounded solution for these D_q 's can be combined to obtain a feasible integral solution for A while increasing the cost by an $O(\log^2 k)$ factor. We begin by describing the decomposition procedure.

Decomposition Procedure: By adding dummy priorities if necessary, let us assume without loss of generality that k is an integral power of 2. Let the priorities be indexed by $1, \dots, k$ (with 1 being the lowest priority and k the highest). For priorities p, p', q, q' such that $p \leq p'$ and $q \leq q'$, let $B([p, p'][q, q'])$ denote the submatrix of B consisting of columns (resp. rows) with priorities in the range $[p, p']$ (resp. $[q, q']$).

A crucial observation is that if $p \geq q'$, then for any $p' \geq p$ and $q \leq q'$, the submatrix $B' = B([p, p'][q, q'])$ is also a submatrix of A . This simply follows as p is the lowest priority of any set in B' , which is at least as large as the priority of any row.

We define the decomposition of B inductively as follows: In the base case when $k = 1$, the decomposition consists of the single matrix $\{B\}$ itself. For general k , we define the decomposition as consisting of the matrix $D_0 = B([k/2 + 1, k][1, k/2])$, together with the (inductive) decompositions of

$$B_1 = B([1, k/2][1, k/2]) \quad \text{and} \quad B_2 = B([k/2 + 1, k][k/2 + 1, k]).$$

Note that both B_1 and B_2 involve only $k/2$ priorities. See Figure 2 for an illustration of the decomposition scheme.

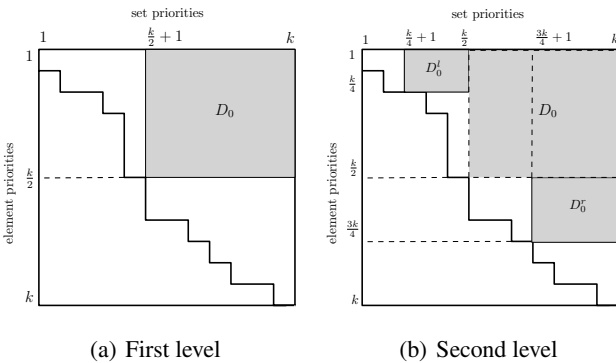


Fig. 2. Recursive partitioning of B

Lemma 1. *The decomposition procedure satisfies the three properties claimed above.*

Proof. It is easily checked that this procedure gives a decomposition of B . Moreover, as $B([k/2 + 1, k][1, k/2])$ is a valid submatrix of A , it follows that all the submatrices obtained in the decomposition are submatrices of A .

Next we show by a simple induction that each element and set can lie in at most $1 + \log k$ submatrices D_q . This is clearly true if $k = 1$. Now, suppose $k > 1$ and consider some fixed element j . It can lie in the submatrix D_0 and exactly one of B_1 or B_2 . Since B_1 and B_2 are $k/2 \times k/2$ matrices, the claim follows by induction. An identical argument works for sets.

Rounding Algorithm:

1. Let $x^* = \{x_1^*, x_2^*, \dots, x_m^*\}$ be some optimal fractional solution for the set system B .
2. For each submatrix D in the decomposition of B , do the following:
 - (a) Let \mathcal{S}_D denote the collection of sets that lie in D , and let x_D denote the solution $\min(x^*(1 + \log k), 1)$ restricted to sets in \mathcal{S}_D .
 - (b) Let E_D be the set of elements in D that are covered fractionally to an extent of at least 1 by x_D .
 - (c) Consider the set system (E_D, \mathcal{S}_D) . Now, x_D is a feasible fraction set cover solution for this set system. As the hereditary integrality gap of A is α , apply the rounding algorithm to (E_D, \mathcal{S}_D) with x_D as the fractional solution. Let \mathcal{S}'_D denote the collection of sets chosen by this rounding.
3. Our final solution is simply the union of \mathcal{S}'_D , over all D in the decomposition of B .

Analysis: We first show that the algorithm produces a valid set cover and then bound the total cost, which will complete the proof of Theorem 2.

Lemma 2. *Each element in B is covered by some set in the solution.*

Proof. Consider some fixed element j . As j lies in at most $1 + \log k$ sets in the decomposition of B , and as x^* is a feasible fractional solution for B , there is some submatrix D that contains j and such that $\sum_{i \in \mathcal{S}_D} x_j^* A_{ij} \geq 1/(1 + \log k)$. Hence, the solution x_D covers j to an extent of at least 1 (i.e. $j \in E_D$), and the rounding algorithm applied to (E_D, \mathcal{S}_D) will ensure that j is covered by some set in \mathcal{S}'_D .

Lemma 3. *The total cost of the solution produced is $O(\log^2 k)\alpha$ times the LP cost.*

Proof. As A has hereditary integrality gap α , the cost of the collection \mathcal{S}'_D is at most α times the cost of the fraction solution x_D , which itself is at most $O(\log k)$ times the cost of solution x^* restricted to the variables (sets) in \mathcal{S}_D . As each set i lies in $O(\log k)$ submatrices D in the decomposition of B , summing up over all D , this implies that the total cost of the solution is $O(\alpha \log^2 k)$ times the cost of x^* .

3 Set Systems with Small Hereditary Discrepancy

In this section, we consider set systems with low hereditary discrepancy and prove Theorem 3. Recall that, for a set system (X, \mathcal{S}) the discrepancy is defined as $\text{disc}(X, \mathcal{S}) = \min_f \max_{S' \in \mathcal{S}} |\sum_{e \in S'} f(e)|$ where $f : X \rightarrow \{-1, 1\}$ is a two coloring of the universe X , and the hereditary discrepancy is defined as $\text{herdisc}(X, \mathcal{S}) = \max_{X' \subseteq X} \text{disc}(X', \mathcal{S}|_{X'})$ where $\mathcal{S}|_{X'}$ is the collection of sets restricted to the elements X' .

In the setting of Theorem 3 where the (dual) set system A^T has *hereditary discrepancy* at most α , this means that given any sub-collection \mathcal{S}' of sets, there is a $\{-1, +1\}$ coloring χ of \mathcal{S}' that satisfies $|\sum_{i \in \mathcal{S}'} A_{ij} \chi(i)| \leq \alpha$ for each row j .

3.1 Rounding Procedure

Let x^* be an optimal solution to the following natural LP relaxation of $\text{MSC}(A, d, \mathbf{1}, c)$.

$$\begin{aligned} & \text{minimize} && \sum_{i=1}^m c_i x_i \\ & \text{subject to} && \sum_{i=1}^m A_{ij} x_i \geq d_j && \forall 1 \leq j \leq n \\ & && x_i \in [0, 1] && \forall S_i \in \mathcal{S} \end{aligned}$$

Scaling. First we scale x^* by a factor of α , i.e. $x'_i = \min(\alpha x_i^*, 1)$. Let \mathcal{H} be the set of variables for which $x' = 1$ and let $\mathcal{L} = \mathcal{S} \setminus \mathcal{H}$. Clearly, the solution $\{x'_i : i \in \mathcal{L}\}$ is feasible to the following (residual) set of constraints (for all elements j):

$$\sum_{i \in \mathcal{L}} A_{ij} x'_i \geq \alpha \left(d_j - \sum_{i \in \mathcal{H}} A_{ij} \right)$$

Iterative Rounding. In this step, we iteratively round the solution x' , without increasing its total cost, while also ensuring that the constraints remain satisfied. Consider the binary representation of variables in solution x' and let t denote the least significant bit in the representation. We index the rounds ℓ from t down to 1. Let us initialize the solution in the initial round $\ell = t$ as $x^t = x'$ and repeat the following step.

Round ℓ : Let S_ℓ denote the set of columns that have a 1 in their least significant bit (i.e. at position ℓ) in this round, and let $f_\ell : S_\ell \rightarrow \{-1, 1\}$ be a ± 1 coloring of the columns that minimizes discrepancy (w.r.t S_ℓ) for all the rows. Clearly, there exists one with discrepancy at most α .

Now, consider the following two solutions: For all $i \in S_\ell$, set $x_i^+ = x_i^\ell + \frac{f_\ell(i)}{2^\ell}$ and $x_i^- = x_i^\ell - \frac{f_\ell(i)}{2^\ell}$. As $x_i^+ + x_i^- = 2x_i^\ell$, it is easy to see that at least one of the solutions x^+ or x^- has cost no more than that of x^ℓ , and we set $x^{\ell-1}$ to that solution. Furthermore, because we have either added or subtracted $1/2^\ell$ from all the variables in S_ℓ , the least significant bit of the solution $x^{\ell-1}$ is now $\ell - 1$.

Having ensured that the cost does not increase, it remains to bound the change in the coverage of any element, for which we use the bounded discrepancy of the coloring f_ℓ . Indeed, since f_ℓ has discrepancy at most α , we have that $\sum_{i \in S_\ell} A_{ij} f_\ell(i) \in [-\alpha, \alpha]$ for all j , and hence

$$\sum_{i \in S_\ell} A_{ij} (x^{\ell-1} - x^\ell) = \sum_{i \in S_\ell} A_{ij} \frac{1}{2^\ell} f_\ell(i) \geq -\frac{\alpha}{2^\ell}.$$

Thus the coverage for any element drops by at most $\alpha/2^\ell$ in round ℓ , and this will be crucial for the analysis.

Output. By the invariant about the least significant bit after each round, at the conclusion of the rounding phase, all variables are either 0 or 1. Our final solution is then $\mathcal{H} \cup \mathcal{X}$, where $\mathcal{X} := \{i \in \mathcal{L} : x_i^0 = 1\}$.

3.2 Analysis

Final Cost. As the cost of the solution can only go down in each round ℓ , the cost of the final solution is at most that of x' , which is at most α times the LP optimum.

Feasibility. Consider any element j . In round ℓ , the coverage of j can drop by at most $\alpha/2^\ell$. Hence, over all the rounds, the total drop in coverage is $\sum_{i=1}^t \alpha/2^\ell$ which is *strictly smaller* than α . Therefore,

$$\sum_{i \in \mathcal{L}} A_{ij} x_i^0 > \alpha \left(d_j - \sum_{i \in \mathcal{H}} A_{ij} \right) - \alpha \geq \left(d_j - \sum_{i \in \mathcal{H}} A_{ij} - 1 \right). \quad (4)$$

As $\sum_{i \in \mathcal{L}} A_{ij} x_i^0$ is integral, the strict inequality in (4) implies that $\sum_{i \in \mathcal{L}} A_{ij} x_i^0 \geq (d_j - \sum_{i \in \mathcal{H}} A_{ij})$, and hence the solution is feasible.

4 Set Systems with Small VC Dimension

We consider set systems with small VC dimension and prove theorem 4. We first recall the definition of VC dimension. Given a set system $(\mathcal{X}, \mathcal{S})$, for $X' \subseteq X$ let $\mathcal{S}|_{X'}$ denote the set system restricted to X' . We say that X' is shattered by \mathcal{S} if there are $2^{|X'|}$ distinct sets in $\mathcal{S}|_{X'}$. A set system (X, \mathcal{S}) is said to have VC dimension d , if $d > 0$ is the smallest integer such that no $d+1$ point subset $X' \subseteq X$ can be shattered. Also recall that the incidence matrix B of a PSC instance is obtained as $B_{ij} = A_{ij} \mathbf{1}_{\pi(S_i) \geq \pi(e_j)}$.

Theorem 4. *The VC dimension of the set system B is at most one more than that of A .*

Proof. Consider the matrix B and order the demand and the supply-priorities in non-decreasing order (as shown in Figure 1). Let d denote the VC dimension of A , and for the sake of contraction, suppose B has VC dimension at least $d+2$. Then there exists a subset of rows Y in B , $|Y| = d+2$, such that there are 2^{d+2} distinct columns in the submatrix induced by Y . Consider all the 2^{d+1} columns in this submatrix that have a 1 in their bottom-most coordinate. As $B_{ij} = A_{ij} \mathbf{1}_{\pi(S_i) \geq \pi(e_j)}$, every coordinate starting from the bottom-most coordinate with a 1 in B has the same value in both A and B . But then the rows of Y except the bottom-most one (there are $d+1$ of them) are shattered by A , contradicting that it has VC dimension d .

5 Lower Bounds

In this section, we establish a $\Omega(\log k)$ lower bound on the integrality gap of the PSC LP for hereditary instances for which the underlying set cover instance has $O(1)$ hereditary integrality gap. This shows that Theorem 2 is tight up to an $O(\log k)$ factor.

The Hinged Axis-Aligned Rectangle Cover Problem. The underlying problem we start with is the *Hinged Axis-Aligned Rectangle Cover* problem (HARC): we are given a set of points $X = \{(x_j, y_j) : 1 \leq j \leq n\}$ in the 2-dimensional plane and a collection of axis-aligned rectangles $\mathcal{S} = \{[a_i, b_i] \times [0, d_i] : 1 \leq i \leq m\}$ all of which have one side on the X-axis. The goal is to pick a minimum number of rectangles to cover X where the notion of coverage is simply containment of the point inside the rectangle.

It is known that the natural LP relaxation for this problem has an integrality gap of 2 [1]. Moreover the gap is clearly hereditary as any sub-collection of sets and elements is also a problem of the same type. We will now show that the natural LP relaxation for *Priority HARC* has an integrality gap of $\Omega(\log k)$ when there are k priorities. We achieve this by relating the priority version of the HARC problem to the 2D rectangle covering problem (2DRC).

The 2D Rectangle Cover Problem. In the 2DRC problem, we are given a set of points $X = \{(x_j, y_j) : 1 \leq j \leq n\}$ in the 2-dimensional plane and a collection of axis-aligned rectangles $\mathcal{S} = \{[a_i, b_i] \times [c_i, d_i] : 1 \leq i \leq m\}$. The goal is to pick a minimum number of rectangles to cover each of the given points where the notion of coverage is simply containment of the point inside the rectangle.

Step 1: Reducing 2DRC to Priority HARC. Consider an instance of 2DRC $\mathcal{I} = (X, \mathcal{S})$. Without loss of generality, we assume that no two points share any coordinate (which we can ensure by moving the points by infinitesimal amounts). We now create the Priority HARC instance \mathcal{I}' as follows: for each point $(x_j, y_j) \in X$, create the point $e_j = (x_j, y_j)$ with priority $\pi(e_j) = 1/y_j$. For each rectangle $[a_i, b_i] \times [c_i, d_i]$, create an axis-aligned rectangle $S_i = [a_i, b_i] \times [0, d_i]$ with priority $\pi(S_i) = 1/c_i$. By construction, it is clear to see that $(x_j, y_j) \in [a_i, b_i] \times [c_i, d_i]$ iff e_j is covered by S_i and $\pi(S_j) \geq \pi(e_i)$.

Note. Since each set could (in the worst case) be associated with its own priority, the number of priorities k created in the above reduction is $O(m)$, where m is the number of rectangles.

Step 2: Lower Bound for 2DRC. Therefore it suffices to obtain an integrality gap for the 2DRC in order to get the same gap for priority HARC. The idea is to use recent super-linear lower bounds on ϵ -nets for 2DRC, and the strong connection between ϵ -nets and the LP relaxation for set cover. In particular, we use the following theorem on ϵ -net lower bounds due to Pach and Tardos [11].

Theorem 6 ([11]). *For any $\epsilon > 0$ and for any sufficiently large integer $m \geq m_0(\epsilon) = \text{poly}(\frac{1}{\epsilon})$, there exists a range space (X, R) , where X is a set of points in \mathbb{R}^2 and R consists of m axis-aligned rectangles, such that the size of the smallest ϵ -net (w.r.t the points) is at least $\Omega(\frac{1}{\epsilon} \log \frac{1}{\epsilon})$. That is, if $S \subseteq R$ is such that any point $p \in X$ that is contained in at least ϵm rectangles is covered by a rectangle in S , then $|S| = \Omega(\frac{1}{\epsilon} \log \frac{1}{\epsilon})$.*

To get our integrality gap, consider the following set cover instance: the sets are all the rectangles, and elements are only those points which are contained in at least ϵm points. Then clearly from the above theorem, and the fact that any feasible integer solution is a valid ϵ -net, we get $\text{Opt} \geq \Omega(\frac{1}{\epsilon} \log \frac{1}{\epsilon})$. To complete the proof, we need to upper bound the cost of an optimal LP solution: if we set each x_S to $1/(\epsilon m)$, we see that such a solution is feasible, i.e., all elements are fractionally covered to extent 1; furthermore, the total cost of this fractional cover is $1/\epsilon$. Now this immediately gives us an integrality gap of $\Omega(\log(1/\epsilon))$. Now notice that the number of rectangles in the instances created can be set to $m = m_0(\epsilon) = \text{poly}(\frac{1}{\epsilon})$. Therefore, the integrality gap is also $\Omega(\log m) = \Omega(\log k)$, since m is linearly related to the number of priorities k as noted above. This proves Theorem [5](#).

Acknowledgments. We would like to thank Anupam Gupta for several useful discussions.

References

1. Bansal, N., Pruhs, K.: The geometry of scheduling. In: FOCS 2010, pp. 407–414 (2010)
2. Bar-Noy, A., Bar-Yehuda, R., Freund, A., Naor, J.(S.), Schieber, B.: A unified approach to approximating resource allocation and scheduling. *J. ACM* 48, 1069–1090 (2001)
3. Bronnimann, H., Goodrich, M.: Almost optimal set covers in finite VC-dimension. *Discrete and Computational Geometry* 14, 463–479 (1995)
4. Carr, R.D., Fleischer, L.K., Leung, V.J., Phillips, C.A.: Strengthening integrality gaps for capacitated network design and covering problems. In: SODA 2000, pp. 106–115 (2000)
5. Chakrabarty, D., Grant, E., Könemann, J.: On column-restricted and priority covering integer programs. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 355–368. Springer, Heidelberg (2010)
6. Charikar, M., Naor, J.(S.), Schieber, B.: Resource optimization in qos multicast routing of real-time multimedia. *IEEE/ACM Trans. Netw.* 12, 340–348 (2004)
7. Chuzhoy, J., Gupta, A., Naor, J.(S.), Sinha, A.: On the approximability of some network design problems. In: SODA 2005, pp. 943–951 (2005)
8. Eisenbrand, F., Pálvölgyi, D., Rothvoß, T.: Bin packing via discrepancy of permutations. In: SODA, pp. 476–481 (2011)
9. Feige, U.: A threshold of $\ln n$ for approximating set cover. *J. ACM* 45, 634–652 (1998)
10. Kolliopoulos, S.G.: Approximating covering integer programs with multiplicity constraints. *Discrete Appl. Math.* 129, 461–473 (2003)
11. Pach, J., Tardos, G.: Tight lower bounds for the size of epsilon-nets. *CoRR*, abs/1012.1240 (2010)
12. Schrijver, A.: *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, Heidelberg (2003)
13. Varadarajan, K.: Weighted geometric set cover via quasi-uniform sampling. In: STOC 2010, pp. 641–648 (2010)

Bandwidth and Low Dimensional Embedding

Yair Bartal^{1,*}, Douglas E. Carroll^{2,**}, Adam Meyerson^{3,***},
and Ofer Neiman^{4,†}

¹ Institute of Computer Science, Hebrew University of Jerusalem, Israel
yair@cs.huji.ac.il

² NetSeer.com

dougecarroll@yahoo.com

³ Department of Computer Science, University of California, Los Angeles
awm@cs.ucla.edu

⁴ Princeton University and Center for Computational Intractability
oneiman@princeton.edu

Abstract. We design an algorithm to embed graph metrics into ℓ_p with dimension and distortion both dependent only upon the bandwidth of the graph. In particular we show that any graph of bandwidth k embeds with distortion polynomial in k into $O(\log k)$ dimensional ℓ_p , $1 \leq p \leq \infty$. Prior to our result the only known embedding with distortion independent of n was into high dimensional ℓ_1 and had distortion exponential in k . Our low dimensional embedding is based on a general method for reducing dimension in an ℓ_p embedding, satisfying certain conditions, to the intrinsic dimension of the point set, without substantially increasing the distortion. As we observe that the family of graphs with bounded bandwidth are doubling, our result can be viewed as a positive answer to a conjecture of Assouad [2], limited to this family. We also study an extension to graphs of bounded tree-bandwidth.

1 Introduction

The problem of embedding graph metrics into normed spaces with low dimension and distortion has attracted much research attention (cf. [14]). In this paper we study the family of graphs with bounded bandwidth. The bandwidth of an unweighted graph $G = (V, E)$ is the minimal k such that there exists an ordering of the vertices in which the end points of every edge are at most k apart. Let d_G

* The work was done in part while the author was at the Center for the Mathematics of Information, Caltech, and the Institute for Pure and Applied Mathematics, UCLA. Supported in part by a grant from the Israeli Science Foundation (195/02) and in part by a grant from the National Science Foundation (NSF CCF-065253).

** Research done while a student at UCLA.

*** Research supported by the National Science Foundation under Grant No. CCF-106540.

† Part of this work was done while the author was a student at the Hebrew university, and was visiting Caltech and UCLA. Supported by a grant from the Israeli Science Foundation (195/02).

be the shortest path metric on the graph G . Let (Y, ρ) be a metric space, we say that an embedding $f : V \rightarrow Y$ has distortion $D \geq 1$ if there exists a constant $c > 0$ such that for all $x, y \in V$,

$$d_G(x, y) \leq c\rho(f(x), f(y)) \leq Dd_G(x, y) .$$

Our main result is the following.

Theorem 1. *For any integer $k \geq 1$ there exist $d = d(k)$ and $D = D(k)$ with the following property. For every $p \geq 1$ and graph $G = (V, E)$ with bandwidth at most k , there exists an embedding of (V, d_G) into ℓ_p space of dimension d with distortion D . In particular we have: $D(k) = O(k^2)$ and $d(k) = O(\log^2 k)$. Alternatively we also get: $D(k) = O(k^{2.001})$ and $d(k) = O(\log k)$.*

Our work is related to a conjecture of Assouad [2]. The doubling constant of a metric space is the minimal α such that any ball of radius r can be covered by α balls of half the radius, then the doubling dimension of V is defined as $\log_2 \alpha$. Assouad proved that for any metric (V, d) , the "snow-flake" metric $(V, d^{1-\epsilon})$ embeds into Euclidean space with distortion and dimension depending only on the doubling constant of (V, d) and on ϵ . Assouad conjectured that this is possible even when $\epsilon = 0$, but this was disproved by [19] (a quantitative bound was given by [12]). It is also shown in [12] that Assouad's conjecture holds for the family of doubling tree metrics. As the doubling constant of graphs with bandwidth k can be bounded by $O(k)$, one can view our result as providing a different family of doubling metrics for which Assouad's conjecture holds.

Graphs with low bandwidth play an important role in fast manipulation of matrices, in particular computing Gauss elimination and multiplication [10]. In his seminal paper Feige [11] showed an approximation algorithm for computing the bandwidth with poly-logarithmic guarantee. The bandwidth of a graph also plays a role in certain biological settings, such as gene clustering problems [20].

There has been a great deal of previous work on embedding families of graphs into ℓ_p with bounded distortion (for example [9,12,13,18,8]). The problem of embedding graphs of bounded bandwidth has been first tackled by [7]. They show that this family of graphs includes interesting instances which do not fall within any of the cases for which constant distortion embeddings are known. In their paper they show that bounded bandwidth graphs can be embedded into ℓ_1 [7] with distortion independent of the number of vertices n . However, the distortion of their embedding was exponential in the bandwidth k . Also, the dimension of that embedding was dependent on the number of vertices (in fact polynomial in n). We improve the result of [7] for graphs of bandwidth k in several ways: First, our embedding works for any ℓ_p space ($1 \leq p \leq \infty$) as a target space, not just ℓ_1 . Second, the distortion obtained is polynomial in k ; specifically: $O(k^{2+\theta})$. Finally, we show that the dimension can be independent of n as well, and as low as $O((\log k)/\theta)$ (for any $0 < \theta < 1$).

Note that the fact that a graph has bandwidth k can be viewed as providing an embedding into 1 dimension with expansion bounded by k , but without any

control on the contraction. Our result means that by increasing the dimension to $O(\log k)$, one can get a bound not only on the expansion but also on the contraction of the embedding.

The low dimensionality of our embedding follows from a generalization we give for a result of [1], who study embedding metric spaces in their intrinsic dimension. In [1] it is shown that for any n point metric space, with doubling constant α , there exists an embedding into ℓ_p space with distortion $O(\log^{1+\theta} n)$ and dimension $O((\log \alpha)/\theta)$ (for any $0 < \theta < 1$). Here, we extend their method in a way that may be applicable for reducing the dimension of embeddings in other settings. We show sufficient conditions on an embedding of any metric space (V, d) into ℓ_p (possibly high dimensional) with distortion γ , allowing to reduce the dimension to $O((\log \alpha)/\theta)$ with distortion only $O(\gamma^{1+\theta})$.

Our embedding for graphs of bandwidth k is obtained as follows: we first provide an embedding with distortion $O(k^2)$ which satisfies the conditions of the dimension reduction theorem. Our final embedding follows from the fact that the doubling constant of graphs of bandwidth k is $O(k)$.

It is worth noting that our embedding provides bounds independent of n for all $1 \leq p \leq \infty$. This is unusual: most previous non-trivial results for embedding infinite graph classes into normed spaces with constant distortion (independent of n) have ℓ_1 as a target metric [9,13] (and require high dimension). This is because of strong lower bounds indicating that trees have a distortion of $\Omega(\sqrt{\log \log n})$ [15] and tree-width two graphs have a distortion of $\Omega(\sqrt{\log n})$ when embedded into ℓ_2 [17]. Since bandwidth k graphs do not include all trees, these lower bounds will not apply and we are able to embed into ℓ_2 with constant (independent of n) distortion. We observe that ℓ_2 is potentially a more natural and useful target metric.

We extend our study to graphs of bounded tree-bandwidth [7] (see Definition 7 for precise definition). While this family of graphs includes all trees and thus requires distortion at least $\Omega(\sqrt{\log \log n})$ when embedded into ℓ_2 , we are still able to apply our techniques with an additional overhead related to the embedability of trees. We provide an embedding of tree-bandwidth k graphs into ℓ_2 with distortion $O(\text{poly}(k)\sqrt{\log \log n})$ and into ℓ_1 with distortion polynomial in k . Moreover, when the graph has bounded doubling dimension we can apply our dimension reduction technique to achieve distortion and dimension depending solely on the doubling dimension and on k , utilizing the embedding of [12] for trees with bounded doubling dimension.

In general there has been a great deal of work on finding low-distortion embeddings. These embeddings have a wide range of applications in approximation algorithms, and in most cases low dimension is also desirable (for example improving the running time). Our work makes further progress towards achieving low-distortion results with dimension reduced to the intrinsic dimension. In particular, our embeddings imply better bounds in applications such as nearest neighbor search, distance labeling and clustering.

1.1 Summary of Techniques

The result of [1] includes the design of a specific embedding technique (locally padded probabilistic partitions), combined with the careful application of the Lovasz Local Lemma to show that it is possible to randomly merge the coordinates of this embedding in such a way that there is a non-zero probability that no distance is contracted. This can then be combined with constructive versions of the local lemma [3,16] to deterministically produce a low-dimensional embedding with no contraction.

We decouple the embedding technique of [1] from the local lemma, showing that any embedding technique which satisfies certain locality properties as well as having a single coordinate which lower bounds each particular distance can be applied in this way. Given any metric space (V, d) with doubling constant α , we give sufficient conditions to reduce the dimension of an embedding of V into ℓ_p with distortion γ to have dimension $O((\log \alpha \log \gamma) / \log(1/\epsilon))$ and distortion $O(\gamma/\epsilon)$ where $\gamma^{-1} < \epsilon < 1$. This approach allows some modularity in defining an embedding – if we are given a low distortion embedding (potentially much lower distortion than $\log n$ for some source metrics) which satisfies the locality properties then we can maintain the low distortion while obtaining low dimension as well.

In order to demonstrate the power of this approach, we apply it to the problem of embedding bounded bandwidth graphs into ℓ_p . We first need to define a low distortion embedding. The embedding of [7] is not useful for our purpose as it does not satisfy the necessary properties (in particular the single coordinate lower bound on distances fails) and because its distortion is undesirably high (exponential in bandwidth). Instead, we define a new embedding. The basis for our embedding is the standard scale based approach [18] using probabilistic partitions of [12,1] as a black box. The problem is that when using this approach we obtain an expansion factor of 1 at each scale of the embedding. The number of scales is logarithmic in the graph diameter, giving us a total expansion of $\Theta(\log n)$. The key innovation of our bandwidth embedding is showing that the number of scales can be reduced to $O(k)$.

Of course, for any scale there may be some point pair whose distance is at that scale (there are n^2 point pairs and only $\log n$ scales after all). We cannot simply remove some scales and expect our distortion to be reasonable. Instead, we compute a set of *active scales* for each graph vertex; these are the scales that represent distances to other points which are nearby in the optimum bandwidth ordering of the graph. We will reduce coordinate values to zero for vertices which do not consider the coordinate's scale to be active. Each vertex has only $O(k)$ active scales; the issue now is that different vertices have different scales and if two adjacent vertices have different active scales we might potentially introduce large expansion. In addition, we need to show that the critical coordinates which maintain the lower bound of $d(x, y)$ (thus preventing contraction) are active at one of the two points (x or y). Instead of applying active coordinates directly, we allow coordinates to decline gracefully by upper-bounding them by the distance to the nearest point where they are inactive, then use the bandwidth ordering to

prove that the critical coordinates for preventing contraction are not only active where they need to be, but have not declined by too much to be useful.

A careful analysis of this construction shows that we can obtain distortion of $O(k^2)$. We also show that our modified embedding still possesses the locality properties. Thus we can apply our dimension reduction technique to get dimension $O((\log k)/\theta)$ while maintaining the distortion bound up to a factor $O(k^\theta)$.

2 Embedding in the Doubling Dimension

2.1 Preliminaries and Definitions

Definition 1. *The **doubling constant** of a metric space (V, d) is the minimal integer α such that for any $r > 0$ and $x \in V$, the ball $B(x, 2r)$ can be covered by α balls of radius r . The **doubling dimension** or **intrinsic dimension**, denoted by $\dim(V)$, is defined as $\log \alpha$.*

Suppose we are given a metric space (V, d) along with a randomly selected mapping $\phi : V \rightarrow \mathbb{R}^D$ for some dimension D . For $1 \leq c \leq D$ we denote by $\phi_c(x)$ the c 'th coordinate of $\phi(x)$ and thus we have $\phi_c : V \rightarrow \mathbb{R}$. We may assume w.l.o.g. that all coordinates of all points in the range of this mapping are non-negative.

Definition 2. *The mapping ϕ is **single-coordinate** (ϵ, β) **lower-bounded** if for every pair of points $x, y \in V$ there is some coordinate c such that $|\phi_c(x) - \phi_c(y)| \geq \beta d(x, y)$ with probability at least $1 - \epsilon$.*

In the metric embedding literature, we often speak of an embedding having contraction β . For ℓ_1 embedding, this means there is a set of coordinates whose sum is lower-bounded by $\beta d(x, y)$. The single-coordinate (ϵ, β) lower-bounded condition is stronger than contraction β , although for ℓ_p norms with large values of p (i.e. as p tends towards infinity) it becomes equivalent.

Definition 3. *Given a mapping ϕ , the ℓ_1 **expansion** of ϕ is $\delta = \max_{x, y} \frac{\|\phi(x) - \phi(y)\|_1}{d(x, y)}$.*

We observe that the expansion of ϕ when viewed as an ℓ_p embedding for $p > 1$ will be at most the ℓ_1 expansion. On the other hand, the single-coordinate (ϵ, β) lower-bound condition will still imply that the embedding has contraction β (for any pair of points with $1 - \epsilon$ probability).

Definition 4. *A mapping ϕ has the **local property** if for every coordinate c we can assign a scale s_c which is a power of two such that the following conditions hold:*

1. *For every $x, y \in V$ with $d(x, y) > s_c$ we have either $\phi_c(x) = 0$ or $\phi_c(y) = 0$.*
2. *For every $x, y \in V$, if there is a single-coordinate lower-bound for x, y , it has scale $\Omega(d(x, y)) < s_c < d(x, y)$.*

We observe that a mapping ϕ can be viewed as an embedding of (V, d) into normed space. Provided that the mapping is single-coordinate (ϵ, β) lower-bounded, we can eliminate contraction by repeatedly (and independently) selecting such mappings many times over and weighting the results by the number of selections, then multiplying all coordinates by $\frac{1}{\beta}$. This provides an embedding into ℓ_1 with distortion upper-bounded by $\frac{\delta}{\beta}$; note that this embedding can also be viewed as into ℓ_p for any $p > 1$ and in fact will have only lower distortion (the single-coordinate lower-bound condition still guarantees non-contraction).

2.2 Low Dimensional Embedding

An embedding ϕ maps (V, d) to potentially high dimensional space, and we are interested in *reducing the dimension* of such an embedding to resemble the doubling dimension of (V, d) without increasing the distortion. While for general ϕ such a result would imply dimension reduction for ℓ_1 (which is impossible in general [6]), the additional constraints that ϕ be single-coordinate lower-bounded and local will enable us to reduce the dimension. In the full version of the paper we prove the following generalization of [11].

Theorem 2. *Suppose we are given a metric space (V, d) with doubling constant α and a mapping $\phi : (V, d) \rightarrow \mathbb{R}^D$ where ϕ is single-coordinate (ϵ, β) lower-bounded, local, and has ℓ_1 expansion at most δ for some $\beta/\delta \leq \epsilon \leq 1/8$. Then for any $1 \leq p \leq \infty$ we can produce in polynomial time an embedding $\tilde{\phi} : (V, d) \rightarrow \ell_p^m$ with distortion at most $O(\delta/(\epsilon\beta))$, where $m = O\left(\frac{\log \alpha \log(\delta/\beta)}{\log(1/\epsilon)}\right)$.*

Next we construct an embedding with the local property, which will serve as a basis embedding in Section 3. Recall that a partition $P = \{C_1, \dots, C_n\}$ of an n -point metric space (V, d) is a pairwise disjoint collection of clusters (possibly some clusters are empty) which covers V , and $P(x)$ denotes the cluster containing $x \in V$. W.l.o.g we may assume that $\min_{x \neq y \in X} \{d(x, y)\} \geq 1$. The following lemma is a generalization of a lemma of [12] and was proven in [11].

Lemma 1. *For any metric space (V, d) with doubling constant α , any $0 < \Lambda < \text{diam}(V)$ and $0 < \epsilon \leq 1/2$ there exists a distribution $\hat{\mathcal{P}}$ over a set of partitions \mathcal{P} such that the following conditions hold.*

- For any $1 \leq j \leq n$, $\text{diam}(C_j) \leq \Lambda$.
- For any $x \in V$, $\Pr_{P \sim \hat{\mathcal{P}}}[B(x, \epsilon\Lambda/(64 \log \alpha)) \not\subseteq P(x)] \leq \epsilon$.

For every scale $s \in I = \{2^i \mid -1 \leq i < \log(\text{diam}(V)), i \in \mathbb{Z}\}$ let $P_s = \{C_1(s), \dots, C_n(s)\}$ be a random partition sampled from $\hat{\mathcal{P}}$ with $\Lambda = s$, and let $c_1(s), \dots, c_n(s)$ be n coordinates that are assigned to the scale s . The random mapping is defined as

$$\phi_{c_j(s)}(x) = \begin{cases} d(x, V \setminus C_j(s)) & x \in C_j(s) \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

and

$$\phi = \bigoplus_{s \in I, 1 \leq j \leq n} \phi_{c_j(s)} \quad (2)$$

Proposition 1. *For any $0 < \epsilon \leq 1/2$ the mapping ϕ is single-coordinate $(\epsilon, \epsilon/(128 \log \alpha))$ lower-bounded, and its ℓ_1 expansion is at most $O(\log(\text{diam}(V)))$.*

Proof. For any $x, y \in V$ let s be a power of two such that $s < d(x, y) \leq 2s$, then in the coordinates assigned to scale s , the first property of [Lemma 1](#) suggests that it must be that x, y fall into different clusters of the partition associated with the coordinates. Let j be such that $x \in C_j$, it follows that with probability $1 - \epsilon$, $\phi_{c_j(s)}(x) \geq \epsilon s / (64 \log \alpha) \geq \epsilon d(x, y) / (128 \log \alpha)$ and that with probability 1, $\phi_{c_j(s)}(y) = 0$.

To see that the ℓ_1 expansion is at most $2(\log(\text{diam}(V)) + 2)$, note that the triangle inequality implies that $|\phi_{c_j(s)}(x) - \phi_{c_j(s)}(y)| \leq d(x, y)$ for any $x, y \in V$ and $j \in [n]$, and since $\phi_{c_j(s)}(x)$ is non-zero for a single $j \in [n]$ it follows that for any $s \in I$

$$\sum_{1 \leq j \leq n} |\phi_{c_j(s)}(x) - \phi_{c_j(s)}(y)| \leq 2d(x, y), \quad (3)$$

and hence

$$\sum_{s \in I, 1 \leq j \leq n} |\phi_{c_j(s)}(x) - \phi_{c_j(s)}(y)| \leq \sum_{s \in I} 2d(x, y) = 2(\log(\text{diam}(V)) + 2)d(x, y).$$

Proposition 2. *The mapping ϕ has the local property.*

Proof. The first local property is immediate by the first property of [Lemma 1](#) and by [\(II\)](#). The second local property follows from the proof of [Proposition 1](#).

3 Low Distortion ℓ_p -Embeddings of Low Bandwidth Graphs

3.1 Preliminaries and Definitions

Definition 5. *Given graph $G = (V, E)$ and linear ordering $f : V \rightarrow \{1, 2, \dots, |V|\}$ the bandwidth of f is $\max\{|f(v) - f(w)| \mid (v, w) \in E\}$. The bandwidth of G is the minimum bandwidth over all linear orderings f . Given an optimal bandwidth ordering f , the index of u is simply $f(u)$.*

Definition 6. *Define $\lambda(x, y) = |f(x) - f(y)|$ which is the distance between x, y in the bandwidth ordering f of G .*

In what follows we are given a graph G of bandwidth k , the metric space associated with G is the usual shortest-path metric, and we assume we are given the optimal ordering f obtaining this bandwidth. This ordering is computable in time exponential in k , and since our embedding only improves upon previous work (for example Bourgain [\[4\]](#)) when k is quite small, it may be reasonable to assume that the ordering is given. In general computing the best bandwidth ordering is NP-Hard, and the best approximations are poly-logarithmic in n [\[11\]](#).

Proposition 3. *Let G be a graph of bandwidth k . Then there exists an ordering where for any $x, y \in G$, $\lambda(x, y) \leq k \cdot d(x, y)$.*

Proof. Assume $d(x, y) = r$, and let $P_{xy} = (x = v_0, v_1, \dots, v_r = y)$ be a shortest path in G connecting x and y , then by the triangle inequality $\lambda(x, y) \leq \sum_{i=1}^r \lambda(v_{i-1}, v_i)$. By the definition of bandwidth for all $1 \leq i \leq r$, $\lambda(v_{i-1}, v_i) \leq k$, hence the proposition follows.

Proposition 4. *If $G = (V, E)$ has bandwidth k , then the doubling constant α of G is at most $4k + 1$.*

Proof. Consider the ball of radius $2r$ about some point $x \in V$. We must show that this ball can be covered by at most $4k + 1$ balls of radius r .

Consider any integer $0 < a \leq r$. Let Y_a be the set of points y such that $d(x, y) = a$; similarly let Y_{a+r} be the set of points y such that $d(x, y) = a + r$. We claim that the set of balls of radius r centered at points in $\{x\} \cup Y_a \cup Y_{a+r}$ covers the ball of radius $2r$ around x . In particular, consider any point z in this ball. If $d(x, z) \leq r$ then $z \in B(x, r)$. If $a \leq d(x, z) < a + r$ then there is some shortest path from x to z of length $d(x, z)$ which must include a point y with $d(x, y) = a$ and $d(y, z) = d(x, z) - a < r$. It follows that $z \in B(y, r)$ and that $y \in Y_a$. If $a + r \leq d(x, z) < 2r$ then again there is a shortest path from x to z of length $d(x, z)$ which must include a point y with $d(x, y) = a + r$ and $d(y, z) = d(x, z) - a - r < r$. It follows that $z \in B(y, r)$ and $y \in Y_{a+r}$.

Now consider the various sets $\{x\} \cup Y_a \cup Y_{a+r}$ as we allow a to range from 1 to r . With the exception of x , these sets are disjoint for distinct values of r . So every point in $B(x, 2r)$ other than x appears exactly once. It follows that there must be some choice of a such that the size of this set is only $1 + \frac{1}{r}|B(x, 2r)|$. Since the graph G has bandwidth k , it follows that any pair of adjacent nodes are within k of each other in the bandwidth ordering. So all points in $B(x, 2r)$ are within $2rk$ of x in the ordering, and thus there are at most $4rk$ such points. From this it follows that we need only $4k + 1$ balls to cover $B(x, 2r)$.

The remainder of this section will be devoted to proving our main theorem, that graphs of bounded bandwidth embed into ℓ_p with low dimension and distortion.

Theorem 3. *Let G be a graph with bandwidth k and let $0 < \theta < 1$, then for any $p \geq 1$, there exists an embedding of G into ℓ_p with distortion $O(k^{2+\theta})$ and dimension $O((\log k)/\theta)$.*

3.2 Proof of Theorem 3

Consider the mapping ϕ defined in (2). By Proposition 1 combined with Theorem 2 (noting that for unweighted graphs we get $O(\log n)$ ℓ_1 expansion) we can transform it into an embedding of a graph with bandwidth k into any ℓ_p space of dimension $O((\log k)/\theta)$ with distortion $O(\log^{1+\theta} n)$ for any $0 < \theta \leq 1$.

¹ A somewhat simpler argument could be applied to give an $O(k)$ bound on the doubling constant, which would suffice for our application. The argument presented here seems to give a better estimate on the constant.

Our main innovation is to reduce the number of scales effecting each of the points, thereby reducing the overall distortion to $O(k^2)$.

Let G be a graph with bandwidth k and f be the optimal ordering obtaining this bandwidth. Let $\alpha \leq 4k + 1$ be the doubling constant of G . For each scale s , we will say that scale s is *active* at point x if there exists a y such that $\lambda(x, y) \leq k$ and $s/8 \leq d(x, y) \leq 4s$. We define $h_s(x)$ to be the distance from x to the nearest point z for which s is not active (note that $h_s(x) = 0$ if s is not active at x). We then define a mapping $\hat{\phi}$ as follows (recall the definition of s_c in [Definition 4](#)):

$$\hat{\phi}_c(x) = \min(\phi_c(x), h_{s_c}(x))$$

We will claim that for suitable values of ϵ , this $\hat{\phi}$ is single-coordinate $(\epsilon, \frac{1}{k})$ lower-bounded for all point pairs x, y with $|f(x) - f(y)| \leq \frac{1}{4}d(x, y)$, that it is local, and that it has ℓ_1 expansion bounded by $O(k)$. The final embedding will be $\hat{\phi}$ concatenated with an extra coordinate f , which is the location of the points in the bandwidth ordering. This will allow us to apply [Theorem 2](#) without the f coordinate, then add in the f coordinate to get our final embedding.

Lemma 2. *The mapping $\hat{\phi}$ has ℓ_1 expansion at most $O(k)$.*

Proof. Consider any pair of points x, y . We observe that the total number of scales which are *active* for these two points is at most $O(k)$, this is because for x there are at most $2k - 1$ other points z satisfying $\lambda(x, z) \leq k$, and each of these points may activate at most 6 different scales. We conclude that there are at most $O(k)$ non-zero coordinates for these two points. So the ℓ_1 expansion expression has only $O(k)$ non-zero terms. Let c be a non-zero coordinate. The triangle inequality suggests that each coordinate produces expansion of at most 1 in $\hat{\phi}$, that is

$$\phi_c(x) - \phi_c(y) \leq d(x, y)$$

If $\hat{\phi}_c(y) = \phi_c(y)$ then since $\hat{\phi}_c(x) \leq \phi_c(x)$, we can write:

$$\hat{\phi}_c(x) - \hat{\phi}_c(y) \leq \phi_c(x) - \phi_c(y) \leq d(x, y)$$

On the other hand, suppose that $\hat{\phi}_c(y) = h_s(y)$ where $s = s_c$. Then there is some z where scale s is inactive, such that $h_s(y) = d(y, z)$. Now

$$\hat{\phi}_c(x) - \hat{\phi}_c(y) \leq h_s(x) - h_s(y) \leq d(x, z) - d(y, z) \leq d(x, y)$$

From this we conclude that each non-zero coordinate produces expansion at most 1, and when we total this over $O(k)$ non-zero coordinates we get total ℓ_1 expansion at most $O(k)$.

We note that adding the coordinate f does not increase the expansion by much. In particular, for any point pair x, y we have $|f(x) - f(y)| \leq kd(x, y)$ by [Proposition 3](#). So the extra coordinate increases expansion by at most an additive k .

The tricky part is proving that the mapping is single-coordinate $(\epsilon, \frac{1}{k})$ lower-bounded. Given some pair of points x, y , one might imagine that the critical coordinates were deemed *inactive* for x and y , and thus the single-coordinate lower-bound will no longer hold. We will prove that this is not the case.

Lemma 3. *For any $k^{-1/2} \leq \epsilon \leq 1/2$ the mapping $\hat{\phi}$ is single-coordinate $(\epsilon, \Omega(\frac{1}{k}))$ lower-bounded for any pair of points x, y with $|f(x) - f(y)| \leq \frac{1}{4}d(x, y)$.*

Proof. Consider any pair of points x, y with $|f(x) - f(y)| \leq \frac{1}{4}d(x, y)$. Let $x' \in B(x, \frac{d(x, y)}{8k})$ and $y' \in B(y, \frac{d(x, y)}{8k})$. Let s be the scale such that $d(x, y)/2 \leq s < d(x, y)$. We will show that scale s must be *active* at x' or at y' . But this holds for *any pair* of points x', y' from the appropriate balls around x, y . It follows that for one of these two balls it must be the case that scale s is active at *all points in the ball*. Suppose without loss of generality that this is $B(x, \frac{d(x, y)}{8k})$. Then since all points in this ball have scale s active, we conclude that $h_s(x) \geq \frac{d(x, y)}{8k}$. By [Proposition 1](#) and the local property of ϕ , there is a coordinate c assigned to scale s , which with $1 - \epsilon$ probability, has $\phi_c(x) \geq \Omega(\frac{\epsilon}{\log \alpha})d(x, y)$ and by the first local property of ϕ also $\phi_c(y) = 0$. If this event occurs, then since $\Omega\left(\frac{\epsilon}{\log \alpha}\right) \geq \Omega\left(\frac{k^{-1/2}}{2 \log k}\right) \geq \Omega(1/k)$, we get that $\hat{\phi}_c(x) \geq d(x, y) \min(\Omega(\frac{\epsilon}{\log \alpha}), \frac{1}{8k}) \geq \Omega(\frac{d(x, y)}{k})$, and of course $\hat{\phi}_c(y) = 0$. We conclude that x, y are $(\epsilon, \Omega(1/k))$ lower bounded.

In the remainder of proof we show that indeed scale s must be active at either x' or y' . Since $d(x, x') \leq d(x, y)/(8k)$ and $d(y, y') \leq d(x, y)/(8k)$ it follows that $d(x', y') \geq d(x, y)(1 - \frac{1}{4k}) \geq \frac{3}{4}d(x, y)$. On the other hand, $|f(x) - f(x')| \leq \frac{d(x, y)}{8}$ and similarly for $|f(y) - f(y')|$ from which we can conclude that $|f(x') - f(y')| \leq \frac{1}{2}d(x, y)$. Now consider a fixed shortest path from x' to y' . Assume without loss of generality that $f(x') < f(y')$. We define two special points along this path as follows:

- \tilde{x} is the first point on the path from x' to y' such that for all points z subsequent to or equal to \tilde{x} on the path, we have $f(z) \geq f(x')$.
- \tilde{y} is the first point on the path from \tilde{x} to y' with $f(\tilde{y}) \geq f(y')$

These points will be auxiliary points showing that scale s is active at either x' or y' . For instance to show that scale s is active at x' it is enough to show that $\lambda(x', \tilde{x}) \leq k$ and that $s/8 \leq d(x', \tilde{x}) \leq 4s$. We observe that because any pair of consecutive vertices on a path are at most k apart in the bandwidth ordering, it must be that $|f(x') - f(\tilde{x})| \leq k$ and $|f(y') - f(\tilde{y})| \leq k$. Note that for every point z on the path from \tilde{x} to \tilde{y} , the value of $f(z)$ is a unique point between $f(x')$ and $f(y')$. We conclude that $d(\tilde{x}, \tilde{y}) \leq |f(x') - f(y')| \leq \frac{1}{2}d(x, y)$. Since these points are on the shortest path, we know that $d(x', y') = d(x', \tilde{x}) + d(\tilde{x}, \tilde{y}) + d(\tilde{y}, y')$. It follows that either $d(x', \tilde{x}) \geq \frac{1}{8}d(x, y)$ or $d(\tilde{y}, y') \geq \frac{1}{8}d(x, y)$.

On the other hand, it is not hard to see that $d(x', \tilde{x}) \leq d(x', y') \leq d(x, y) + \frac{1}{4k}d(x, y) \leq 2d(x, y)$ and similarly for $d(y', \tilde{y})$. We conclude that indeed scale s must be active for one of x', y' .

Lemma 4. *The mapping $\hat{\phi}$ is local.*

Proof. The first condition follows immediately from the fact that ϕ is local and $\hat{\phi}_c(x) \leq \phi_c(x)$ for all c and x . The second condition follows from [Lemma 3](#).

We now combine the lemmas and apply [Theorem 2](#) to $\hat{\phi}$. This guarantees bounded contraction for point pairs with $|f(x) - f(y)| \leq \frac{1}{4}d(x, y)$. We add the single additional coordinate f , and this guarantees bounded contraction for points with $|f(x) - f(y)| \geq \frac{1}{4}d(x, y)$. Choosing for any $0 < \theta < 1$, $\epsilon = k^{-\theta}$ will give distortion $O(k^{2+\theta})$ and dimension $O((\log k)/\theta)$.

4 Tree Bandwidth

We will give an embedding of a graph of low tree-bandwidth [\[7\]](#) into ℓ_p . The distortion will be polynomial in k , with a multiplicative $O(\sqrt{\log \log n})$ term for $p > 1$ [\[5\]](#). This improves upon the result of [\[7\]](#) by reducing the distortion and extending to ℓ_p . All the proofs appear in the full version of the paper.

Definition 7. [\[7\]](#) *Given a graph $G = (V, E)$, we say that it has **tree-bandwidth** k if there is a rooted tree $T = (I, F)$ and a collection of sets $\{X_i \subset V \mid i \in I\}$ such that: $\forall i, |X_i| \leq k$, $V = \bigcup X_i$, the X_i are disjoint, $\forall (u, v) \in E$, u and v lie in the same set X_i or $u \in X_i$ and $v \in X_j$ and $(i, j) \in F$, and if i has parent $p(i)$ in T , then $\forall v \in X_i, \exists u \in X_{p(i)}$ such that $d(u, v) \leq k$. T is called the decomposition tree of G .*

Theorem 4. *There is a randomized algorithm to embed tree-bandwidth k graphs into ℓ_p with expected distortion $O(k^3 \log k + k\rho)$ where ρ is the distortion for embedding the decomposition tree into ℓ_p .*

In the case of ℓ_1 , there is a simple embedding of a tree with $\rho = 1$. For ℓ_2 , the bound of [\[5\]](#) ensures $\rho = O(\sqrt{\log \log n})$.

We can also apply [Theorem 2](#) to reduce the dimension the embedding of [Theorem 4](#). To do this we need to bound the dimension in which the tree can be embedded. We have the following lemma,

Lemma 5. *Let G be a graph with tree bandwidth k , and let α be the doubling constant of G , then the doubling dimension of the decomposition tree T for G is $\log \alpha_T = O((\log \alpha)(\log k))$.*

It follows that we can use an embedding for the decomposition tree T of G where the distortion and dimension are functions of the doubling dimension of T , as shown in [\[12\]](#), and therefore are a function of α and k alone.

Theorem 5. *Suppose that we are given a tree-bandwidth k graph along with its tree decomposition. Let the doubling constant of this graph be α . Let α_T be the doubling constant of T , given by [Lemma 5](#). Further, suppose that there exists an embedding of the tree decomposition into $d(\alpha_T)$ dimensional ℓ_p with distortion $\rho(\alpha_T)$. Then for any $0 < \theta < 1$ there is an embedding of the graph into ℓ_p with expected distortion $O(k^{3+\theta} \log k + k\rho(\alpha_T))$ and dimension $O((\log \alpha)/\theta + d(\alpha_T))$.*

References

1. Abraham, I., Bartal, Y., Neiman, O.: Embedding metric spaces in their intrinsic dimension. In: SODA 2008: Proceedings of the 18th Ann. ACM-SIAM Sym. on Discrete Algorithms (2008)
2. Assouad, P.: Plongements lipschitziens dans \mathbb{R}^n . Bull. Soc. Math. France 111(4), 429–448 (1983)
3. Beck, J.: An algorithmic approach to the lovász local lemma. Random Struct. Algorithms 2, 343–365 (1991)
4. Bourgain, J.: On Lipschitz embedding of finite metric spaces in Hilbert space. Israel J. Math. 52(1-2), 46–52 (1985)
5. Bourgain, J.: The metrical interpretation of superreflexivity in Banach spaces. Israel J. Math. 56(2), 222–230 (1986)
6. Brinkman, B., Charikar, M.: On the impossibility of dimension reduction in l_1 . In: FOCS, pp. 514–523 (2003)
7. Carroll, D.E., Goel, A., Meyerson, A.: Embedding bounded bandwidth graphs into l_1 . In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4051, pp. 27–37. Springer, Heidelberg (2006)
8. Chakrabarti, A., Jaffe, A., Lee, J.R., Vincent, J.: Embeddings of topological graphs: Lossy invariants, linearization, and 2-sums. In: Proceedings of the 49th Ann. IEEE Sym. on Foundations of Computer Science, Washington, DC, USA, pp. 761–770 (2008)
9. Chekuri, C., Gupta, A., Newman, I., Rabinovich, Y., Sinclair, A.: Embedding k -outerplanar graphs into l_1 . In: SODA 2003: Proceedings of the 14th Ann. ACM-SIAM Sym. on Discrete Algorithms, pp. 527–536 (2003)
10. Chinn, P.Z., Chvátalová, J., Dewdney, A.K., Gibbs, N.E.: The bandwidth problem for graphs and matrices - a survey. Journal of Graph Theory (6), 223–254 (1982)
11. Feige, U.: Approximating the bandwidth via volume respecting embeddings. In: Proceedings of the 30th Ann. ACM Sym. on Theory of Computing, STOC 1998, pp. 90–99. ACM, New York (1998)
12. Gupta, A., Krauthgamer, R., Lee, J.s.R.: Bounded geometries, fractals, and low-distortion embeddings. In: Proceedings of the 44th Ann. IEEE Sym. on Foundations of Computer Science, Washington, DC, USA, p. 534 (2003)
13. Gupta, A., Newman, I., Rabinovich, Y., Sinclair, A.: Cuts, trees and l_1 -embeddings of graphs. In: Proceedings of the 40th Ann. Sym. on Foundations of Computer Science, pp. 399–409 (1999)
14. Linial, N., London, E., Rabinovich, Y.: The geometry of graphs and some of its algorithmic applications. Combinatorica 15(2), 215–245 (1995)
15. Matousek, J.: On embedding trees into uniformly convex banach spaces. Israel Journal of Mathematics 114, 221–237 (1999)
16. Moser, R.A., Tardos, G.: A constructive proof of the general lovász local lemma. J. ACM 57(2), 1–15 (2010)
17. Newman, I., Rabinovich, Y.: A lower bound on the distortion of embedding planar metrics into euclidean space. In: SCG 2002: Proceedings of the 18th ann. sym. on Computational geometry, pp. 94–96. ACM Press, New York (2002)
18. Rao, S.: Small distortion and volume preserving embeddings for planar and Euclidean metrics. In: Proceedings of the 15th Ann. Sym. on Computational Geometry, pp. 300–306. ACM, New York (1999)
19. Semmes, S.: On the nonexistence of bilipschitz parameterizations and geometric problems about a_∞ weights. Revista Matemática Iberoamericana 12, 337–410 (1996)
20. Zhu, Q., Adam, Z., Choi, V., Sankoff, D.: Generalized gene adjacencies, graph bandwidth, and clusters in yeast evolution. IEEE/ACM Trans. Comput. Biol. Bioinformatics 6(2), 213–220 (2009)

$O(1)$ -Approximations for Maximum Movement Problems

Piotr Berman¹, Erik D. Demaine², and Morteza Zadimoghaddam²

¹ Department of Computer Science and Engineering, Pennsylvania State University, University Park, PA 16802, USA

² MIT Computer Science and Artificial Intelligence Laboratory, 32 Vassar St., Cambridge, MA 02139, USA

Abstract. We develop constant-factor approximation algorithms for minimizing the maximum movement made by pebbles on a graph to reach a configuration in which the pebbles form a connected subgraph (connectivity), or interconnect a constant number of stationary nodes (Steiner tree). These problems model the minimization of the total time required to reconfigure a robot swarm to achieve a proximity (e.g., radio) network with these connectivity properties. Our approximation factors are tight up to constant factors, as none of these problems admit a $(2 - \epsilon)$ -approximation assuming $P \neq NP$.

1 Introduction

A central problem in swarm robotics is to reconfigure the robots into an arrangement with a desired property. For example, in the *connectivity* goal, the proximity of the robots should form a connected graph. Two motivations for this goal are forming a connected data network with short-range radios, and forming a connected physical network for transporting materials. In the first situation, the robots initially communicate via another channel, e.g., via slow and/or power-intensive long-distance communication (such as satellite or the same radios with power turned up high), or via two traversals by aircraft to locate robots and disseminate locations. Another connectivity goal is *Steiner connectivity*, where a subset of the robots should form a connected network that contains k stationary nodes (such as buildings or sensors) which need to be interconnected. In both of these problems, we suppose that we know the initial locations of robots, and that we have a map of the environment the robots can traverse and defining proximity among the robots. Our goal is to move the robots as quickly as possible into a configuration with the desired property.

These problems fit into the broad family of *movement problems*, introduced in [5], and further explored in [6,8]. In general, we have a graph G with pebbles on some of the vertices. The goal is to move some of the pebbles along edges to obtain a desired property P , while minimizing either the maximum movement or the total movement of the pebbles, motivated by minimizing either execution time or energy usage. Specific problems considered in [5] include the *connectivity movement problem* mentioned above, where property P is that the vertices with at least one pebble induce a connected graph, and the *s-t path movement problem*, where property P is that the vertices with at

least one pebble induce a graph in which two given vertices are in a common connected component (the special case of the Steiner connectivity movement problem with two terminals).

Several approximation algorithms and inapproximability results for these movement problems were presented in [5]. Of primary relevance to this paper, the connectivity and s - t path movement problems with the maximum movement objective function have an $O(\sqrt{n})$ -approximation algorithm [1]. Furthermore, both of these problems are $(2 - \epsilon)$ -inapproximable assuming $P \neq NP$. On the other hand, with the total movement object function, the connectivity movement problem is $\Omega(n^{1-\epsilon})$ -inapproximable, and there is an $\tilde{O}(n)$ -approximation algorithm. This negative result motivates our focus here on the maximum movement objective.

In FOCS 2008, Friggstad and Salavatipour [8] considered a *facility location movement problem*: moving pebbles of two types, facilities and clients, to achieve the property that each client is at a vertex that contains some facility. This problem was first introduced in [5], which presented a simple 2-approximation algorithm for this problem with the maximum movement objective function. But the problem with the total movement objective remained an open problem until Friggstad and Salavatipour developed an LP-based constant-factor approximation algorithm.

Demaine et. al. characterize the tractable and intractable movements problems in a general setting in [6]. They consider the class of edge-deletion minimal graphs that have the desired property P , and can be the final destination of the pebbles, e.g. for connectivity this class is the set of subtrees of graph G with at most m nodes where m is the number of pebbles. They prove that if for property P the treewidth is bounded by a constant, the movement problem of this property is Fixed Parameter Tractable. They consider the number of pebbles as the parameter of their algorithms. So their results are applicable only for small number of pebbles, e.g. $m = O(\log(n))$.

As some applications of these movement problems in Wireless Networks, we can refer to the works of Basagni et. al. who consider movements of some mobile sinks in order to maximize the network lifetime [1][2][3]. There are several fixed (non-mobile) sensor nodes with limited power. The mobile nodes should move between some fixed sites to gather data from the sensor nodes. We have to find the best schedule for mobile sinks' movements to maximize the network lifetime, i.e. lifetime is equal to the time duration in which all nodes have enough energy supply to handle all their operations including sensing, transmitting, and etc.

Our Results. Connectivity is the quintessential movement problem. For the total movement objective function, the best possible approximation ratio is known to be roughly linear [5] (assuming $P \neq NP$). But for the maximum movement objective function, which models the parallel execution time or makespan required for the motion, there is a huge gap in the best possible approximation ratio, between $2 - \epsilon$ and $O(\sqrt{n})$. In this paper, we close this gap up to constant factors, by obtaining the first constant-factor approximation algorithm for the connectivity movement problem.

¹ In fact the approximation ratio is $O(\sqrt{m/OPT})$ where m is the number of pebbles and OPT is the maximum movement of the optimum solution. This ratio can be as large as $\Theta(\sqrt{n})$, when $m = \Theta(n)$ and $OPT = \Theta(1)$.

An ingredient in this result is a constant-factor approximation algorithm for the s - t path movement problem. This result is also a breakthrough, as again the best possible approximation ratio was previously only known to be between $2 - \varepsilon$ and $O(\sqrt{n})$. We use our approximation algorithm for s - t path problem as a black box in our solution to the connectivity problem.

Finally we introduce the *Steiner connectivity movement problem*, which is a natural generalization of the s - t path movement problem. Here we are given a set T of terminal vertices, and the goal is to move some of our pebbles to interconnect all terminal vertices. More precisely, property P is that the vertices with at least one pebble induce a graph that places all terminals in the same connected component. For $|T| = O(\log n)$, we present an $O(|T|)$ -approximation algorithm for the Steiner connectivity movement problem with the maximum movement objective, again using our approximation algorithm for s - t path. Note that we cannot hope to approximate the Steiner connectivity movement problem for arbitrary $|T|$, because even deciding whether there is a feasible solution with the available pebbles is the NP-hard node-weighted Steiner tree problem. Unfortunately we include this result only in the complete version because of space limits.

Techniques. Our algorithms introduce several new techniques for approximating movement problems with the maximum movement objective. In general, these problems would be easy if we allowed approximating the number of pebbles (via resource augmentation) in addition to the cost. But robots cannot (yet) replicate themselves, so resource augmentation is not very useful. We develop powerful tools to resolve multiple desires for the location of a single pebble/robot.

In the s - t path movement problem (Section 2), we define a concept of a *locally consistent paths* such that (a) a correct solution is also locally consistent, (b) the minimum length locally consistent solution can be found in polynomial time, and (c) with only limited additional pebble movement, it can be converted to a consistent solution. For the sake of simplicity, we describe an algorithm with an extremely large polynomial time bound that achieves a 7-approximation; in the full paper, we will describe the available trade-off between the running time and the approximation of the maximum movement.

In the connectivity movement problem (Section 3), we present a three stage algorithm. In all stages, we maintain a set of pebbles S (initially empty), and try to move some pebbles and insert them into S . The new locations of pebbles in S form a connected subgraph. In the first stage, we define dense vertices which are basically the vertices with a large enough number of pebbles around them. Once we find a dense vertex, we can use the s - t path algorithm, and the pebbles around the dense vertex to insert a subset of pebbles into set S . We do this process iteratively in the first stage until there exists no dense vertex.

Then we analyze the structure of the remaining pebbles in the optimal solution. The final locations of all pebbles in the optimal solution is a connected subgraph and has a spanning tree T . In Figure 1 (page 68), you can see an instance of our problem, and its optimal solution on the left. The spanning tree T is shown with bold edges on the right. After the first stage, some pebbles are inserted into S , we remove these pebbles from tree T . The remaining subgraph is a forest F with some interesting properties. We prove that there can not be a vertex in this forest with three long paths attached to it, call such a vertex “tripod vertex”, e.g. vertex v in Figure 1. We prove that after the first stage

there exists no tripod vertex in forest F . We then prove that every subtree in F either has low diameter or is a long path with some low-diameter branchlets attached to it, call it a thick path tree². If the subtree has low diameter, we prove that all its pebbles are close to set S , so we can insert them into S by moving them directly toward this set. If it is a thick path tree, we enumerate to find the head and tail of its longest path, then we use s - t path algorithm to connect its head and tail using its own pebbles. We also prove that every other pebble in this subtree is close to the path we find (we need the second stage for proving this claim), and we can move all these pebbles to connect them to set S with $O(M)$ movement where M is the maximum movement of an optimum solution. We note that one can enumerate on all possible values of M which are $1, 2, \dots, n$, or use binary search to find it. So we can assume that our algorithm known the value of M .

The important problem is that these subtrees are not easily distinguishable. Since pebbles from different subtrees can be close to each other in their starting configuration, we can not find out which pebbles are in which subtree. We know that two adjacent pebbles in the optimum solution have distance at most $M + 1 + M = 2M + 1$ from each other in their starting configuration. We make graph H with the remaining pebbles (outside set S) as its vertices. We put an edge between two pebbles if their distance is at most $2M + 1$ from each other. This way we can be sure that all pebbles in a subtree of F are in the same connected component in H . But there might be pebbles from several subtrees of F in the same connected component in H .

We define some relaxed versions of dense vertices, and try to insert more pebbles into set S by finding these vertices in the second stage. In the third stage, we find all remaining pebbles close to set S , and insert them to set S by moving them toward S . The second stage helps us prove that there can not be pebbles from more than two thick path trees in a connected component of H . We can distinguish two thick path trees in a common connected component of H with some other techniques in polynomial time. But there might be pebbles from several low diameter subtrees in this connected component as well. We prove that all pebbles in low diameter subtrees are close to set S , and in the third stage we take care of all of them. So after the third stage, we might have some connected components in H containing up to 2 thick path trees. If the connected component has no thick path tree, we can show that all its pebbles are close to set S , so we just move them toward set S . If it has one thick path tree, we can enumerate to find the head and tail of the longest paths of this thick path tree, and run our s - t path algorithm to take care of its pebbles. If it has two thick path trees, we prove that all pebbles from the two thick path trees that are close to each other are around the tails of the longest paths of these two thick path trees. So we enumerate to find the tail of one of the thick path trees, and remove all pebbles in the vicinity of the tail vertex. This way we can distinguish between the two thick path trees, and handle each of them by the s - t path algorithm.

2 s - t Path Movement Problem

In the s - t path movement problem, we are given a graph G with two vertices s and t , and a set of pebbles with positions on the vertices of our graph. We want to move

² We call it Caterpillar Shape tree as well.

some pebbles to construct a path between s and t with at least one pebble on each of its vertices. Our objective is to minimize the maximum movement. The following theorem presents a constant-factor approximation algorithm for this problem.

Theorem 1. *There is a polynomial-time algorithm that finds solution to the s - t the path movement problem such that if there exists a solution that forms a path of length ℓ and moves each pebble along at most M edges, then the algorithm finds a path of length at most ℓ and which moves each pebble along at most $7M-4$ edges.*

Note that we do not move pebbles more than a constant factor of the maximum movement in the optimal solution, and at the same time we use no more pebbles than the optimal solution to construct a path between s and t . So our algorithm can be seen as a bicriteria approximation algorithm: it is optimal in terms of the number of pebbles used in the path, and it is a constant factor approximation with respect to the maximum movement. The fact that we are optimal in terms of the number of pebbles used in the path helps us later to find approximation algorithms for the Steiner connectivity movement problem.

Our algorithm for the s - t path movement problem has two main parts which can be described as follows (with a bit of oversimplification):

1. Find a minimum length locally consistent path \mathbb{P} . A pebble can be moved along at most $3M-2$ edges to a node of \mathbb{P} , but we allow to move a pebble to multiple nodes on \mathbb{P} , provided that they are sufficiently far apart (about $14M$ edges).
2. Convert \mathbb{P} to a consistent path \mathbb{Q} as follows: for each pebble moved to multiple nodes of \mathbb{P} select one of them, which creates gaps in the path; then fill the gaps by moving some pebbles along additional $(7M-4)-(3M-2) = 4M-2$ edges. The length of the path cannot increase.

Because the optimum solution, say of length ℓ , is locally consistent, the length of the locally consistent path that we will find cannot be larger than ℓ , and as we shall see, local consistency is easier to assure than actual consistency. In the same time, under our assumption each inconsistency offers an opportunity of making a shortcut: if a pebble can be moved by distance at most $3M-2$ to two locations that are $14M$ apart on the path, a section of the path with length $14M$ can be replaced with a shorter path with length $6M-4$, provided that we move pebbles from the longer section to the shortcut. This presents a challenge, of course, how to do it in a consistent manner.

2.1 Straighter Paths

We will use the following notation: $d(u, v)$ is the distance from u to v (the length of a shortest path), disk $D(u, R)$ is the set $\{v \in V : d(u, v) \leq R\}$, and a disk set $P(u, R)$ is the set of pebbles with the initial location within $D(u, R)$.

A valid solution has a path $\mathbb{P} = (s = u_0, u_1, \dots, u_\ell = t)$ so that for $i = 0, \dots, \ell$ there exists a pebble p_i such that $p_i \in P(u_i, M)$, moreover, $p_i \neq p_j$ for $0 \leq i < j \leq \ell$.

Given such a solution \mathbb{P} , we can find another, \mathbb{Q} , which we will call *the straighter path*.

We define *milestones* of \mathbb{P} as a subsequence v_1, \dots, v_f of \mathbb{P} , and in turn they define \mathbb{Q} , a straighter version of \mathbb{P} , that connects s, t and the milestones using the shortest paths: from s to v_1 , from v_1 to v_2 , etc., and finally from v_f to t . A milestone v_i is responsible for the section of \mathbb{Q} that is contained in $D(v_i, M-1)$ and which consists of r_i nodes, where $r_i = 2M-1$ for $i < f$, while $r_f \leq 2M-1$ is the distance from t to $D(v_{f-1}, M-1)$.

The initial milestone v_1 is u_j such that $j = \max\{k : d(s, u_k) \leq M-1\}$. Assume that milestone v_i is defined. If $d(v_i, t) \leq M-1$ then v_i is the final milestone. If $M \leq d(v_i, t) \leq 2M-1$ then $v_{i+1} = t$ is the final milestone. If $d(v_i, t) \geq 2M-1$ then $v_{i+1} = u_j$ where $j = \max\{k : d(v_i, u_k) \leq 2M-1\}$.

We modify the pebble movement as follows: consider a node u' of \mathbb{Q} that is a steps before milestone v (or after), where $a < M$. Then we have node u of \mathbb{P} that is a steps before v (or after) on \mathbb{P} and the pebble p that was moved to u (along at most M edges). The modified movement of p traverses a path to u , then to v and finally to u' , hence it uses at most $m + 2a \leq 3M-2$ edges.

Now suppose that we do not know \mathbb{P} or \mathbb{Q} but only the milestones v_1, \dots, v_f . We can find the \mathbb{Q} by connecting s, t and the milestones with shortest paths (between the milestones).

A solution of that form exists if and only if there exist pairwise disjoint sets of pebbles S_1, \dots, S_f such that $S_i \subset P(v_i, 2M-1)$ and $|S_i| = r_i$. This justifies the following definition:

A *consistent solution* is a sequence of nodes v_1, \dots, v_f and a sequence of sets S_1, \dots, S_f such that

1. $d(v_1, s) = M-1$ and $d(v_i, v_{i+1}) = 2M-1$ for $i = 1, \dots, f-2$;
2. either $d(v_{f-1}, v_f) = 2M-1$ and $d(v_f, t) < M$ or $v_f = t$ and $d(v_{f-1}, t) < 2M$;
3. $r_i = 2M-1$ for $i = 1, \dots, f-1$ and $r_f = d(v_f, v_{f-1}) + d(v_f, t) - M + 1$;
4. $S_i \subset P(v_i, 2M-1)$ and $|S_i| = r_i$;
5. $S_i \cap S_j = \emptyset$ for $i \neq j$.

A locally consistent solution satisfies a weaker version of condition 5: $S_i \cap S_j = \emptyset$ for $i < j < i + 7$. The length (or the number of pebbles that are used) is $\sum_{i=1}^f r_i$.

Lemma 1. *There exists a polynomial time algorithm that find a locally consistent solution with the minimum length.*

Lemma 2. *In polynomial time we can transform a locally consistent solution \mathbb{Q} of length ℓ to a solution that uses at most ℓ pebbles that are moved along at most $7M-4$ edges.*

To conclude the proof of Theorem [1](#), observe that we can perform the algorithm described in Lemma [1](#) for different values of M , to find the smallest value for which it concludes successfully, and then we finish using the algorithm described in Lemma [2](#).

3 Connectivity Movement Problem

In the *connectivity movement problem*, we want to move pebbles so that their final positions induce a connected subgraph of graph G . The goal is to minimize the maximum

movement of pebbles. Without loss of generality we can assume that a vertex r is given, and the induced subgraph of the final positions of pebbles should contain r , i.e., there should be a path from the root r to all pebbles in their subgraph. The following theorem is the main result of this section:

Theorem 2. *Given a polynomial-time λ -approximation algorithm for the s - t path movement problem, we can construct a polynomial-time $(8\lambda + 80)$ -approximation algorithm for the connectivity movement problem.*

Definition 1. *Suppose there are m pebbles: p_1, p_2, \dots, p_m . Let u_i be the starting position of pebble p_i , and v_i be its final destination in the optimum solution. Vertices v_1, v_2, \dots, v_m form a connected subgraph in G . This connected subgraph has a spanning tree. Let T be one of its spanning trees. So tree T has at most m vertices because there is at least one pebble on each vertex in T in the optimal solution. In Figure 1 you can see a sample graph before the movements on the left, and with maximum movement one edge, all pebbles form a connected subgraph. Tree T is shown on right by bold edges.*

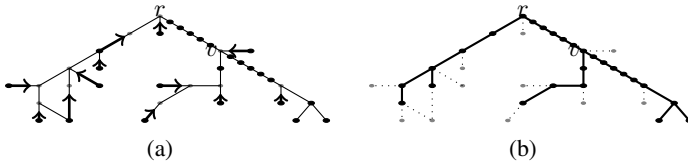


Fig. 1. Each black node contains a pebble, and gray nodes are vertices without pebbles. The arrows show the movements of pebbles in the optimum solution. The maximum movement is one.

Because of lack of space, we present main ideas in this paper, and include detailed description and proofs in the complete version.

3.1 A Constant Factor Approximation for Connectivity Movement Problem

During our algorithm, the set S consists of all pebbles that are connected to root r (via other pebbles) up to now, i.e., this set might be empty at the beginning. We try to add pebbles into S by moving them. Our algorithm basically has three main phases. In Phase 1, we define Operations 1 and 2. We perform these two operations iteratively as long as we can perform them. During these two operations, whenever we move a pebble p , and add it to set S , we make sure that there exists a pebble $p' \in S$ such that its current position has distance at most λM from the starting position of pebble p . Note that pebble p' might or might not be the same as pebble p (in Operation 1 they are the same). When we cannot perform any of these two operations anymore, we can prove some interesting properties about the structure of the remaining pebbles (pebbles outside S).

Then in the second phase, we introduce new Operations 3 and 4, and iteratively perform them. We should note that Operations 3 and 4 are very similar to Operations

1 and 2, but existence of these two similar phases is essential in our algorithm. In the second phase when we move a pebble p , and add it to S , we make sure that there exists a pebble $p' \in S$ that its current position has distance at most xM from the starting position of pebble p where x is a parameter we define later in the second phase.

In the last phase, we present Operation 5, and perform it once. Then we show how to decompose the remaining pebbles into some groups, and take care of each group. In this phase we connect all remaining pebbles to set S , and therefore our connecting task is complete. The maximum movement is a constant times the optimum maximum movement M .

In Operation 2 of the first phase, we define dense vertices, and while there exists a dense vertex, we find a path from it to root r . Using this path we add some pebbles into set S . We iteratively perform Operations 1 and 2 if one of them is possible to do. If both of them are possible to do, Operation 1 has priority. Before performing the following operations, we move all pebbles with distance at most M from root r toward r . We also add them to set S . This way we make sure that S is not empty, and the following operations are feasible.

Operation 1. While there is a pebble $p \notin S$ whose distance to current position of some pebble $p' \in S$ is at most $\lambda M + 1$, we move pebble p towards pebble p' to make them neighbors. We can do this by moving p at most λM edges. Now we can add p to S .

Operation 2. Here we explain the main ideas. Vertex v is called dense, if there are at least $(2\lambda + 2)M$ pebbles outside set S with distance at most λM from v . Let A be the set of the pebbles outside S with distance at most λM from v . If there exists a dense vertex v (with $|A| \geq (2\lambda + 2)M$), we do the following. We gather all pebbles of A on vertex v . This can be done by moving these pebbles along at most λM edges. If the distance of v from at least one pebble in S is at most $(2\lambda + 2)M$, we can use the pebbles in A to connect vertex v to set S . This way we add pebbles of A to set S by moving them along at most $\lambda M + (2\lambda + 2)M = (3\lambda + 2)M$.

Otherwise (if v has distance greater than $(2\lambda + 2)M$ from all pebbles in S), we know that a subset of pebbles outside S and A form a path from a pebble in A to a pebble in S in the optimal solution with maximum movement M . This path should start from a vertex with distance at most $\lambda M + M$ from v to a vertex with distance at most $M + \lambda M$ from set S . So we use our path movement algorithm to build this path. Then we use the pebbles of A to connect v to the first part of this path, and also connect the last part of this path to set S . So we can connect and join all these pebbles to S with maximum movement at most $\lambda M + 2(\lambda + 1)M = (3\lambda + 2)M$.

We keep doing the above operations until there is no dense vertex, and no pebble close to set of pebbles S . Following we show some interesting facts about the structure of the optimal solution after deleting pebbles of S when the first phase is finished, and we cannot perform Operations 1 and 2.

We keep vertices in tree T (defined in Definition [II](#)) that have at least one pebble outside set S and remove the rest of them. The remaining graph is a forest, call it F . Suppose this forest has k connected components T_1, T_2, \dots, T_k . Each of these k connected components is a subtree of T .

Lemma 3. *For each tree T_i , at least one of the following two conditions holds: (1) T_i has diameter less than $2(\lambda - 1)M$. (2) Every vertex in T_i has distance less than $4M$ from the longest path in T .*

Now that we cannot perform Operations 1 and 2 anymore, we start the second phase. In this phase, we define Operations 3 and 4, and we perform them iteratively. Like before, we stop when we cannot perform Operations 3 and 4 anymore. Define x to be $2\lambda + 22$ in the rest of the algorithm.

Operation 3. Consider a vertex v with distance $dis(v, S)$ from set S , i.e., $dis(v, S)$ is the minimum distance of v from pebbles in S . For $0 \leq i \leq xM$, let $N(v, i)$ be the number of pebbles outside S that has distance at most i from v . If $N(v, i)$ is at least $dis(v, S) - (xM - i)$ for some $0 \leq i \leq xM$, and $dis(v, S)$ is at most $(2x + 2)M$, we can do the following. We can gather all pebbles with distance at most i from v , on vertex v , and move them along the shortest path from v to set S to form a path attaching to the current set S . This way we can add some pebbles to S . We cannot necessarily fill the shortest path from v to S completely, but because $N(v, i)$ is at least $dis(v, S) - (xM - i)$, we lack at most $xM - i$ pebbles. Because we moved each pebble at most i in the gathering part, we can say that the starting position of each moved pebble has distance at most $i + (xM - i) = xM$ from the current position of a pebble in S (consider the last pebble in the path we just made and attached to set S). The maximum movement in this operation is at most $i + dis(v, S)$ which is at most $xM + (2x + 2)M = (3x + 2)M$. The interesting property of this operation is that we can use it for different values of $0 \leq i \leq xM$.

Operation 4. Vertex v is called dense with diameter xM , if there are at least $(2x + 2)M$ pebbles outside set S with distance at most xM from v . Like Operation 2, let A be the set of the pebbles outside S with distance at most xM from v . If there exists a dense vertex v (with $|A| \geq (2x + 2)M$), we do the following. We gather all pebbles of set A on vertex v . If the distance of v from at least one pebble in S is at most $(2x + 2)M$, we can use these pebbles in A to connect vertex v to set S . This way we add pebbles of A to set S by moving them along at most $xM + (2x + 2)M = (3x + 2)M$.

Similarly to Operation 2, if the distance of v from set S is more than $(2x + 2)M$, we can construct a path from a vertex w_1 with distance at most $(x + 1)M$ from vertex v to a vertex w_2 with distance at most $(x + 1)M$ from set S using pebbles outside S and A . Like before we can shift this path toward set S , and use the $(2x + 2)M$ pebbles gathered on v to fill in the empty vertices of the path we are building from v to set S . The maximum movement in this case is at most $\max\{(3x + 2)M, \lambda M + (2x + 2)M\} = (3x + 2)M$.

We now want to investigate the structure of the pebbles outside S in the optimal tree T . So we again delete all vertices from tree T that contain some pebbles of set S . We note that tree T is a spanning tree of the subgraph induced on the final positions of the pebbles in the optimum solution. Like before let $T'_1, T'_2, \dots, T'_{k'}$ be the resulting subtrees of the forest we obtain after removing the above vertices from tree T .

Now we are ready to finish our algorithm in the third phase which is a non-iterative phase. We explain the main idea at first. If we knew which pebbles belong to each T'_i (the pebbles that move to vertices of T'_i in the optimum solution) for every $1 \leq i \leq k'$,

we would do the following. For every tree T'_i we know that it either has diameter less than $2(\lambda - 1)M$ or is a tree such that all its pebbles are close to its longest path. If it has small diameter, we can say that all its pebbles have distance at most $(x + 2\lambda)M$ from some pebble in S . We prove this claim later in Lemma 4. About the second type trees, we can find a path connecting two vertices of T'_i with some specific properties. Then we can claim that every pebble in tree T'_i is close to some vertices of the path we find. This is just a raw idea without details. We will show how to distinguish between pebbles of different subtrees, and how to handle each tree (specially the second type trees).

At first we note that two adjacent pebbles in a subtree have distance at most $M + 1 + M = 2M + 1$ from each other. So if we construct a graph with pebbles as its vertices, and connect each pair of pebbles that have distance at most $2M + 1$ from each other, we can say that the pebbles of a subtree T'_i are in the same connected component in this graph. But there might be more than one subtree in a connected component. So we still have the distinguishing problem in a connected component.

The following trick helps us deal with this problem. We do the following operation to get rid of the small diameter subtrees. Note that the following operation is non-iterative; we do it once.

Operation 5. We mark all pebbles outside set S that has distance at most $(x + 2\lambda)M$ from the current position of at least a pebble in S . After marking pebbles, we move all marked pebbles toward some pebbles in S with maximum movement at most $(x + 2\lambda)M$.

Lemma 4. *Every pebble in a tree T'_i with small diameter (less than $2(\lambda - 1)M$) is marked and inserted into set S .*

After Operation 5, we know that all small diameter subtrees are taken care of. Now we might be able to separate different subtrees with large diameter. There are two main issues here. The first one is that some pebbles of a tree T'_i with large diameter might be deleted. The second issue is that two pebbles in two different large diameter subtrees might have distance at most $2M + 1$ (and therefore adjacent in the graph of pebbles we build).

Following we consider the graph of remaining pebbles, and show that both of these problems can be handled in a sophisticated way. Construct graph H with the remaining pebbles (outside S) as its vertices. And we connect two pebbles via an edge if its distance is at most $2M + 1$. If this graph has multiple connected components, we treat each connected component separately. So we assume that this graph has only one connected component. At first we prove some properties of the remaining subtrees. This helps us understand the structure of the graph H .

Definition 2. *Define P_i to be the longest path of subtree T'_i . A subtree T'_i is called long tree, if P_i has length more than $2xM$. Let l'_i be the length of path P_i , and $q_{i,j}$ be the j th pebble of the path P_i for $1 \leq j \leq l'_i$. Also define $v'_{i,j}$ be the j th vertex of the path P_i . Note that there might be also some medium diameter trees other than small diameter and long trees.*

Following we prove that one of the last vertices of the longest paths of every remaining tree (with diameter at least $2(\lambda - 1)M$) is close to set S .

Lemma 5. *For every tree T'_i with diameter at least $2(\lambda - 1)M$, there exists a vertex v in tree T such that v has distance at most $10M$ from vertex v'_{i,l'_i} , the last vertex of the longest path in T'_i , and vertex v is also the final position of some pebble in S in the optimum solution. Therefore vertex v'_{i,l'_i} has distance at most $10M + M + xM = (x + 11)M$ from set S .*

Now we prove that the middle pebbles of long trees (which are the main parts of these trees) are not removed in Operation 5.

Lemma 6. *Consider a long tree T'_i . Pebble $q_{i,j}$ is unmarked (and therefore is not deleted) for all values of $xM + 1 \leq j \leq l'_i - xM$.*

Now that we have more information about the structure of the remaining subtrees, we can take care of them. We consider two cases. Note that H is the graph we constructed with the remaining pebbles as its vertices.

Case 1. Every pebble in H has distance at most $M + 2xM + M + xM = (3x + 2)M$ from some pebble in S . In this case we can move all pebbles of H and connect them to S . The problem is solved completely in this case and the maximum movement of these remaining pebbles would be at most $(3x + 2)M$. Note that if there is no long tree among our trees, our problem will be solved in this case. Because we know that in a subtree, there exists a vertex v which is adjacent to the final position of some pebble p in S in the optimum solution (refer to the beginning of proof of Lemma 5). If the longest path of this subtree is at most $2xM$, it means that the current position of each pebble has distance at most $M + 2xM$ from vertex v . We also know that it takes $M + xM$ edges to reach set S from vertex v because this subtree is obtained after the first four operations. The total distance is not more than $M + 2xM + M + xM = (3x + 2)M$. We conclude that if we have no long tree, the problem is solved in this case.

Case 2. As proved above, in this case we have some long trees. In Lemma 6, we proved that the middle parts of the longest paths of long trees do not get marked, and still exist in graph H . We just need to somehow add the pebbles of these middle parts of the long trees to set S . Because every pebble in a long tree has distance at most $4M$ from a pebble in the longest path in the tree. Each pebble in the longest path is either in the middle part of the path or has distance at most $M + xM + M = (x + 2)M$ from a pebble in the middle part. So every pebble in a long tree would be close (with distance at most a constant times M) to set S , if we add the pebbles in the middle parts of the longest paths of long trees to S . We also know that a tree that is not long has diameter at most $2xM$. This is enough to see that every pebble in a not long tree (medium tree) is close to set S already. So the main problem is finding a way to add pebbles of the middle parts of the longest paths of long trees into S .

Before starting we note that we still might have pebbles from several trees in our graph H . But we know that all middle pebbles of a long tree exist in the same connected component. In fact every pair of pebbles in graph H that are adjacent in tree T , are also adjacent in H . We now show that there cannot be pebbles from three different trees in our connected graph H . There might be pebbles of two trees in H , but we show how to separate pebbles of these two trees from each other. The following lemma shows

the limitations of the edges between pebbles of different trees. In fact, the following Lemma is the main reason that we are able to separate different subtrees.

Lemma 7. *If there is an edge in graph H between two pebbles $q \in T'_1$ and $q' \in T'_2$, we have that the distance between $v'_{1,1}$ and q is at most $11M$, and the distance between $v'_{2,1}$ and q' is also at most $11M$.*

Using Lemma 7 we can prove that there cannot be pebbles from three different trees in H as follows.

Lemma 8. *There can be pebbles from at most two trees in graph H .*

Now we know that there are pebbles from at most two trees in our graph H . We also know that all edges between pebbles of these two trees are close to the tail vertices of the longest paths of two trees. So we can get rid of all these edges by removing vertices around one of this tail vertices. Formally we do the following.

Let T'_1 and T'_2 be the two trees that contribute in H (this approach also works when there is only one tree). We can assume that we know vertex $v'_{1,1}$ because we can guess it (there are at most n possible guesses and one of them is correct).

We remove all pebbles of distance at most $11M$ from $v'_{1,1}$. We now have a graph H' with probably several connected components. Following we show that the middle parts of the long paths is safe. In fact we prove that we might remove at most the first $20M$ pebbles of the longest path, and we do not remove the pebbles $q_{1,20M+1}, q_{1,20M+2}, \dots$. Otherwise we will have a dense vertex which is a contradiction.

Lemma 9. *After deleting pebbles with distance at most $11M$ from $v'_{1,1}$, we do not have any edge between pebbles of our two different trees. We also do not delete any of pebbles $v'_{1,20M+1}, v'_{1,20M+2}, \dots$ if tree T'_1 is a long tree. And we do not delete pebbles $v'_{2,20M+1}, v'_{2,20M+2}, \dots$ if tree T'_2 is a long tree.*

So we do not delete any pebble from middle parts of the long trees. We also removed a set of pebbles in Operation 4. But in Lemma 6 we proved that the middle parts of the longest paths of long trees survive. We conclude the following general lemma.

Lemma 10. *In the graph of remaining pebbles, graph H' , pebbles $q_{1,xM+1}, q_{1,xM+2}, \dots, q_{1,l'_1-xM}$ exist and are in the same connected component for a long tree T'_1 .*

So there might be at most two connected components containing the middle pebbles of longest paths of long trees. Remember these are the only parts we should handle, the rest of the pebbles are either close to set S or close to these two middle parts. In this part, we again treat pebbles of each connected component of graph H' separately. If all pebbles of a connected component in H' are close to set S , we can treat it like case 1 (we can move them directly toward set S). Otherwise this connected component has the middle pebbles of the longest path of either T'_1 or T'_2 (and not both of them for sure). Without loss of generality, assume that it has the middle pebbles of T'_1 . So we know that all pebbles $q_{1,xM+1}, q_{1,xM+2}, \dots, q_{1,l'_1-xM}$ are in the this connected component. We can assume that we know vertices $v'_{1,xM+1}$ and v'_{1,l'_1-xM} (we can guess, and there are at most n^2 possibilities). We know that there is a way to move some pebbles of this connected component to connect s to t with maximum movement at most

M . Using our algorithm for path movement problem, we can move some pebbles to connect these two vertices with maximum movement at most λM . We prove that all pebbles $q_{1,xM+1}, q_{1,xM+2}, \dots, q_{1,l'_1-xM}$ are either used in the path we constructed, or are close to some pebble that we used in this path. At first we note that there is no edge between two pebbles from different trees. So all pebbles that we use are in the tree T'_1 .

Lemma 11. *For every $xM + 1 \leq i \leq l'_1 - xM - 2(\lambda - 1)M$, pebble $q_{1,i}$ has distance at most $(2\lambda + 20)M$ from either the starting position of one of the pebbles we used in the path we constructed from vertex $v'_{1,xM+1}$ to vertex v'_{1,l'_1-xM} , or one of the vertices of this path.*

We can gather all pebbles of our connected component on the path we constructed as follows.

Lemma 12. *Every pebble of our connected component in graph H' , has distance at most $(x + 2\lambda + 3)M$ from some vertex of the path we constructed.*

Using Lemma 12 we can gather all pebbles of the connected component on our path. Now we have to move this path and connect it to S . Note that we can shift the path we constructed to make it connected to S . Vertex v'_{1,l'_1-xM} has distance at most xM from v'_{1,l'_1} . Vertex v'_{1,l'_1} has distance at most $(x+11)M$ from set S using Lemma 5. So we can shift the pebbles of this path along at most $xM + (x+11)M = (2x+11)M$ edges to make them connected to set S . So the maximum movement of all these parts (gathering and shifting) is at most $(x+2\lambda+3)M + (2x+11)M = (3x+2\lambda+14)M$ which is equal to $(8\lambda+80)M$. Now everything is connected to S , and no pebble is remained.

References

1. Basagni, S., Carosi, A., Petrioli, C.: Heuristics for Lifetime Maximization in Wireless Sensor Networks with Multiple Mobile Sinks. In: Proceedings of IEEE International Conference on Communications (ICC 2009), pp. 1–6 (2009)
2. Basagni, S., Carosi, A., Petrioli, C., Phillips, C.A.: Coordinated and Controlled Mobility of Multiple Sinks for Maximizing the Lifetime of Wireless Sensor Networks. To appear in ACM/Springer Wireless Networks (WINET)
3. Basagni, S., Carosi, A., Petrioli, C., Phillips, C.A.: Moving multiple sinks through wireless sensor networks for lifetime maximization. In: Proceedings of IEEE International Conference on Mobile Ad Hoc and Sensor Systems (MASS 2008), pp. 523–526 (2008)
4. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms, 3rd edn. The MIT Press, Cambridge
5. Demaine, E.D., Hajiaghayi, M., Mahini, H., Gharan, S.O., Sayedi-Roshkhar, A., Zadimoghaddam, M.: Minimizing movement. ACM Transactions on Algorithms 5(3), Article 30 (July 2009); Preliminary version appeared at SODA 2007
6. Demaine, E.D., Hajiaghayi, M., Marx, D.: Minimizing movement: Fixed-parameter tractability. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 718–729. Springer, Heidelberg (2009)
7. West, D.B.: Introduction to Graph Theory. Prentice-Hall, Englewood Cliffs (2001)
8. Friggstad, Z., Salavatipour, M.R.: Minimizing movement in mobile facility location problems. In: Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2008), pp. 357–366 (2008)
9. Niedermeier, R.: Invitation to Fixed Parameter Algorithms. Oxford Lecture Series in Mathematics and its Applications. Oxford University Press, Oxford (2006)

Optimal Lower Bounds for Universal and Differentially Private Steiner Trees and TSPs

Anand Bhalgat*, Deeparnab Chakrabarty, and Sanjeev Khanna**

Department of Computer and Information Science, University of Pennsylvania
{bhalgat,deepc}@seas.upenn.edu, sanjeev@cis.upenn.edu

Abstract. Given a metric space on n points, an α -approximate *universal* algorithm for the Steiner tree problem outputs a distribution over rooted spanning trees such that for any subset X of vertices containing the root, the expected cost of the induced subtree is within an α factor of the optimal Steiner tree cost for X . An α -approximate *differentially private* algorithm for the Steiner tree problem takes as input a subset X of vertices, and outputs a tree distribution that induces a solution within an α factor of the optimal as before, and satisfies the additional property that for any set X' that differs in a single vertex from X , the tree distributions for X and X' are “close” to each other. Universal and differentially private algorithms for TSP are defined similarly. An α -approximate universal algorithm for the Steiner tree problem or TSP is also an α -approximate differentially private algorithm. It is known that both problems admit $O(\log n)$ -approximate universal algorithms, and hence $O(\log n)$ approximate differentially private algorithms as well.

We prove an $\Omega(\log n)$ lower bound on the approximation ratio achievable for the universal Steiner tree problem and the universal TSP, matching the known upper bounds. Our lower bound for the Steiner tree problem holds even when the algorithm is allowed to output a more general solution of a distribution on paths to the root. We then show that whenever the universal problem has a lower bound that satisfies an additional property, it implies a similar lower bound for the differentially private version. Using this converse relation between universal and private algorithms, we establish an $\Omega(\log n)$ lower bound for the differentially private Steiner tree and the differentially private TSP. This answers a question of Talwar [19]. Our results highlight a natural connection between universal and private approximation algorithms that is likely to have other applications.

1 Introduction

Traditionally, in algorithm design one assumes that the algorithm has complete access to the input data which it can use unrestrictedly to output the optimal, or near optimal, solution. In many applications, however, this assumption does not hold and the traditional approach towards algorithms needs to be revised. For

* Supported in part by NSF Awards CCF-0635084 and IIS-0904314.

instance, let us take the problem of designing the cheapest multicast network connecting a hub node to a set of client nodes; this is a standard network design problem which has been studied extensively. Consider the following two situations. In the first setting, the actual set of clients is unknown to the algorithm, and yet the output multicast network must be “good for all” possible client sets. In the second setting, the algorithm knows the client set, however, the algorithm needs to ensure that the output preserves the privacy of the clients. Clearly, in both these settings, the traditional algorithms for network design don’t suffice.

The situations described above are instances of two general classes of problems recently studied in the literature. The first situation needs the design of *universal* algorithms; algorithms which output solutions when parts of the input are uncertain or unknown. The second situation needs the design of *differentially private* algorithms; algorithms where parts of the input are controlled by clients whose privacy concerns constrain the behaviour of the algorithm. A natural question arises: how do the constraints imposed by these classes of algorithms affect their performance?

In this paper, we study universal and differentially private algorithms for two fundamental combinatorial optimization problems: the Steiner tree problem and the travelling salesman problem (TSP). The network design problem mentioned above corresponds to the Steiner tree problem. We resolve the performance question of universal and private algorithms for these two problems completely by giving lower bounds which match the known upper bounds. Our techniques and constructions are quite basic, and we hope these could be applicable to other universal and private algorithms for sequencing and network design problems.

Problem formulations. In both the Steiner tree problem and the TSP, we are given a metric space (V, c) on n vertices with a specified root vertex $r \in V$. Given a subset of terminals, $X \subseteq V$, we denote the cost of the optimal Steiner tree connecting $X \cup r$ by $\text{opt}_{ST}(X)$. Similarly, we denote the cost of the optimal tour connecting $X \cup r$ by $\text{opt}_{TSP}(X)$. If X is known, then both $\text{opt}_{ST}(X)$ and $\text{opt}_{TSP}(X)$ can be approximated up to constant factors.

A *universal* algorithm for the Steiner tree problem, respectively the TSP, does not know the set of terminals X , but must output a distribution \mathcal{D} on rooted trees T , respectively tours σ , spanning all vertices of V . Given a terminal set X , let $T[X]$ be the minimum-cost rooted subtree of T which contains X . Then the cost of the universal Steiner tree algorithm on terminal set X is $\mathbf{E}_{T \leftarrow \mathcal{D}}[c(T[X])]$. We say the universal Steiner tree algorithm is α -*approximate*, if for all metric spaces and all terminal sets X , this cost is at most $\alpha \cdot \text{opt}_{ST}(X)$. Similarly, given a terminal set X , let σ_X denote the order in which vertices of X are visited in σ , and let $c(\sigma_X)$ denote the cost of this tour. That is, $c(\sigma_X) := c(r, \sigma_X(1)) + \sum_{i=1}^{|X|-1} c(\sigma_X(i), \sigma_X(i+1)) + c(\sigma_X(|X|), r)$. The cost of the universal TSP algorithm on set X is $\mathbf{E}_{T \leftarrow \mathcal{D}}[c(\sigma_X)]$, and the approximation factor is defined as it is for the universal Steiner tree algorithm.

A *differentially private* algorithm for Steiner trees and TSPs, on the other hand, knows the set of terminals X ; however, there is a restriction on the solution

that it can output. Specifically, a differentially private algorithm for the Steiner tree problem with privacy parameter ε , returns on any input terminal set X a distribution \mathcal{D}_X on trees spanning V , with the following property. Let X' be any terminal set such that the symmetric difference of X' and X is exactly one vertex. Then,

$$\Pr_{\mathcal{D}_{X'}}[T] \cdot \exp(-\varepsilon) \leq \Pr_{\mathcal{D}_X}[T] \leq \Pr_{\mathcal{D}_{X'}}[T] \cdot \exp(\varepsilon),$$

where $\Pr_{\mathcal{D}}[T]$ is the probability of getting tree T when drawn from distribution \mathcal{D} . The cost of the algorithm on set X is $\mathbf{E}_{T \leftarrow \mathcal{D}_X}[c(T[X])]$ as before, and the approximation factor is defined as that for universal trees. Differentially private algorithms for the TSP are defined likewise. To gain some intuition as to why this definition preserves privacy, suppose each vertex is a user and controls a bit which reveals its identity as a terminal or not. The above definition ensures that even if a user changes its identity, the algorithm's behaviour does not change by much, and hence the algorithm does not leak any information about the user's identity. This notion of privacy is arguably the standard and strongest notion of privacy in the literature today; we point the reader to [4] for an excellent survey on the same. We make two simple observations; (a) any universal algorithm is a differentially private algorithm with $\varepsilon = 0$, (b) if the size of the symmetric difference in the above definition is k instead of 1, then one can apply the definition iteratively to get $k\varepsilon$ in the exponent.

For the Steiner tree problem, one can consider another natural and more general solution space for universal and private algorithms, where instead of returning a distribution on trees spanning V , the algorithm returns a distribution \mathcal{D} on collections of paths $P := \{p_v : v \in V\}$, where each p_v is a path from v to the root r . Given a single collection P , and a terminal set X , the cost of the solution is $c(P[X]) := c(\bigcup_{v \in X} E(p_v))$, where $E(p_v)$ is the set of edges in the path p_v . The cost of the algorithm on set X is $\mathbf{E}_{P \leftarrow \mathcal{D}}[c(P[X])]$. Since any spanning tree induces an equivalent collection of paths, this solution space is more expressive, and as such, algorithms in this class may achieve stronger performance guarantees. We show that this more general class of algorithms has the same asymptotic lower bound as the class of algorithms that are restricted to output a spanning tree.

1.1 Previous Work and Our Results

A systematic study of universal algorithms was initiated by Jia et al. [12], who gave $O(\log^4 n / \log \log n)$ -approximate universal algorithms for both the Steiner tree problem and the TSP. Their algorithms were deterministic and returned a single tree and tour respectively. The authors also noted that results of [2, 5] on probabilistically embedding general metrics into tree metrics imply randomized $O(\log n)$ -approximate universal algorithms for these problems. Using properties of the embeddings of [5], Gupta et al. [7] gave deterministic $O(\log^2 n)$ -approximate universal algorithms for both problems.

Jia et al. [12] observe that a lower bound for online Steiner tree algorithms implies a lower bound for universal Steiner tree algorithms; thus, following the result of Imase and Waxman [11], one obtains a lower bound of $\Omega(\log n)$ for any universal Steiner tree algorithm. It is not hard to see that the [11] lower bound also holds for algorithms returning a collection of vertex-to-root paths. Jia et al. [12] explicitly leave lower bounds for the universal TSP as an open problem. Hajiaghayi et al. [9] make progress on this by showing an $\Omega\left(\sqrt[6]{\log n / \log \log n}\right)$ lower bound for universal TSP; this holds even in the two dimensional Euclidean metric space. [9] conjecture that for general metrics the lower bound should be $\Omega(\log n)$; in fact, they conjecture this for the shortest path metric of a constant degree expander. Very recently, this conjecture was proven by Gorodezky et al. [6]; we discuss and compare this particular result and ours at the end of this subsection.

When the metric space has certain special properties (for instance if it is the Euclidean metric in constant dimensional space), Jia et al. [12] give an improved universal algorithms for both Steiner tree and TSP, which achieves an approximation factor of $O(\log n)$ for both problems. Furthermore, if the size of the terminal set X is k , their approximation factor improves to $O(\log k)$ – a significant improvement when $k \ll n$. This leads to the question whether universal algorithms exist for these problems whose approximation factors are a non-trivial function of k alone. A k -approximate universal Steiner tree algorithm is trivial; the shortest path tree achieves this factor. This in turn implies a $2k$ -approximate universal TSP algorithm. Do either of these problems admit an $o(k)$ -approximate algorithm? The constructions of [11] achieving a lower bound of $\Omega(\log n)$ for universal Steiner tree require terminal sets that are of size $n^{\Omega(1)}$, and do not rule out the possibility of an $O(\log k)$ -approximation in general. In fact, for many network optimization problems, an initial polylog(n) approximation bound was subsequently improved to a polylog(k) approximation (e.g., sparsest cut [13,14], asymmetric k -center [18,11], and more recently, the works of Moitra et al. [16,17] on vertex sparsifiers imply such a result for other many cut and flow problems). It is thus conceivable that a polylog(k)-approximation could be possible for the universal algorithms as well.

We prove $\Omega(\log n)$ lower bounds for the universal TSP and the Steiner tree problem, even when the algorithm returns vertex-to-root paths for the latter (Theorems 2 and 7). Furthermore, the size of the terminal sets in our lower bounds is $\Theta(\log n)$, ruling out any $o(k)$ -universal algorithm for either of these problems.

Private vs universal algorithms. The study of differentially private algorithms for combinatorial optimization problems is much newer, and the paper by Gupta et al. [8] gives a host of private algorithms for many optimization problems. Since any universal algorithm is a differentially private algorithm with $\varepsilon = 0$, the above stated upper bounds for universal algorithms hold for differentially private algorithms as well. For the Steiner tree problem and TSP, though, no better differentially private algorithms are known. Talwar, one of the authors of [8], recently posed an open question whether a private $O(1)$ -approximation exists for the Steiner tree problem, even if the algorithm is allowed to use a more

general solution space, namely, return a collection of vertex-to-root paths, rather than Steiner trees [19].

We observe that a simple but useful converse relation holds between universal and private algorithms: “strong” lower bounds for universal algorithms implies lower bounds for differentially private algorithms. More precisely, suppose we can show that for any universal algorithm for the Steiner tree problem/TSP, there exists a terminal set X , such that the probability that a tree/tour drawn from the distribution has cost less than α times the optimal cost is $\exp(-\varepsilon|X|)$ for a certain constant ε . Then we get an $\Omega(\alpha)$ lower bound on the performance of any ε -differentially private algorithm for these problems. (Corollary 1). Note that this is a much stronger statement than merely proving a lower bound on the expected cost of a universal algorithm. The expected cost of a universal algorithm may be $\Omega(\alpha)$, for instance, even if it achieves optimal cost with probability $1/2$, and α times the optimal cost with probability $1/2$. In fact, none of the earlier works mentioned above [11, 12, 9, 6] imply strong lower bounds. The connection between strong lower bound on universal algorithms and lower bounds for differentially private algorithms holds for a general class of problems, and may serve as a useful tool for establishing lower bounds for differentially private algorithms (Section 3).

All the lower bounds we prove for universal Steiner trees and TSP are strong in the sense defined above. As corollaries, we get lower bounds of $\Omega(\log n)$ on the performance of differentially private algorithms TSP and the Steiner tree problem, even when the algorithm returns a collection of paths. This answers the question of Talwar [19] negatively. (Corollaries 1 and 2).

The metric spaces for our lower bounds on universal Steiner tree and TSP are shortest path metrics on constant degree expanders. To prove the strong lower bounds on distributions of trees/tours, it suffices, by Yao’s lemma, to construct a distributions on terminal sets such that any fixed tree/tour pays, with high probability, an $\Omega(\log n)$ times the optimum tree/tour’s cost on a terminal set picked from the distribution. We show that vertices on a sufficiently long random walk suffices in the Steiner tree case, while for TSP, we choose the client set from two independent random walks.

Comparison of our results with [6]: As mentioned above, Gorodezky et al. [6] obtain an $\Omega(\log n)$ lower bound for universal TSP. Their result also gives an $\Omega(k)$ lower bound on the performance of a universal TSP algorithm where k is the number of terminals. Although [6] do not address universal Steiner tree problem directly, the $\Omega(k)$ lower bound for universal TSP implies an $\Omega(k)$ lower bound for universal Steiner tree as well, only when the algorithm returns spanning trees. However, this doesn’t work for algorithms that return collections of vertex-to-root paths. Our result gives the first $\Omega(k)$ lower bound for the universal Steiner tree problem when the algorithm is allowed to return a collection of vertex-to-root paths.

Furthermore, even though our approach is somewhat similar, our proofs are simpler and the results are stronger in that we prove that the probability any randomized algorithm pays $o(\log n)$ times the optimum for a certain subset is ex-

ponentially small in the size of the client set. As explained earlier, these stronger lower bounds are crucial to our technique for proving privacy lower bounds. In particular, to our knowledge, no lower bounds for differentially private Steiner tree (even for weaker algorithms returning spanning trees instead of vertex-to-root paths) and TSP can be deduced from results of [6].

Organization. In Section 2, we establish an $\Omega(\log n)$ lower bound for the universal Steiner tree problem and the universal TSP. As mentioned above, the lower bound for the Steiner tree problem is for a more general class of algorithms which return a collection of paths instead of a single tree. The lower bound established are strong in the sense defined earlier, and thus give an $\Omega(\log n)$ lower bound for private Steiner tree as well as private TSP. We formalize the connection between strong lower bounds for universal problems and approximability of differentially private variants in Section 3. Finally, in interest of space, certain proofs have been omitted from the abstract and can be found in the full version of the paper [3].

2 Lower Bound Constructions

The metric spaces on which we obtain our lower bounds are shortest path metrics of expander graphs. Before exhibiting our constructions, we state a few known results regarding expanders that we use. An (n, d, β) expander is a d regular, n vertex graph with the second largest eigenvalue of its adjacency matrix $\beta < 1$. The girth g is the size of the smallest cycle and the diameter Δ is the maximum distance between two vertices. A t -step random walk on an expander picks a vertex uniformly at random, and at each step moves to a neighboring vertex uniformly at random.

Lemma 1. [15] *For any constant k , there exist (n, d, β) expanders, called Ramanujan graphs, with $d \geq k$, $\beta \leq \frac{2}{\sqrt{d}}$, girth $g = \Theta(\log n / \log d)$, and diameter $\Delta = \Theta(\log n / \log d)$.*

Lemma 2. (Theorem 3.6, [10]) *Given an (n, d, β) expander, and a subset of vertices B with $|B| = \alpha n$, the probability that a t -step random walk remains completely inside B is at most $(\alpha + \beta)^t$.*

Lemma 3. (Follows from Theorem 3.10, [10]) *Given an (n, d, β) expander, a subset of vertices B with $|B| = \alpha n$, and any $\gamma, 0 \leq \gamma \leq 1$, the probability that a t -step random walk visits more than γt vertices in B is at most $2^t \cdot (\alpha + \beta)^{\gamma t}$.*

2.1 Steiner Tree Problem

We consider a stronger class of algorithms that are allowed to return a distribution \mathcal{D} on collections of paths $P := \{p_v : v \in V\}$, where each p_v is a path from v to the root r . As stated in the introduction, this class of algorithms captures as a special case algorithms that simply return a distribution on collection of spanning trees, since the latter induces a collection of paths. We prove the following theorem.

Theorem 1. *For any constant $\varepsilon > 0$ and for large enough n , there exists a metric space (V, c) on n vertices such that for any distribution \mathcal{D} on collections of paths, there is a terminal set X of size $\Theta(\log n)$, such that*

$$\Pr_{P \leftarrow \mathcal{D}} \left[c(P[X]) = o\left(\frac{\log n}{1 + \varepsilon}\right) \text{opt}_{ST}(X) \right] \leq \frac{1}{2} \exp(-\varepsilon|X|) \quad (1)$$

At a high-level, the idea underlying our proof is as follows. We choose as our underlying graph a Ramanujan graph G , and consider the shortest path metric induced by this graph. We show that for any fixed collection P of vertex-to-root paths, a terminal set generated by a random walk q of length $\Theta(\log n)$ in G has the following property with high probability: the edges on q frequently “deviate” from the paths in the collection P . These deviations can be mapped to cycles in G , and the high-girth property is then used to establish that the cost of the solution induced by P is $\Omega(\log n)$ times the optimal cost. Before proving Theorem 1, we establish the following corollaries of it.

Corollary 1. *(a) There is no $o(\log n)$ -approximate universal Steiner tree algorithm. (b) There is no $o(k)$ -approximate universal Steiner tree algorithm where k is the size of the terminal set. (c) For any $\varepsilon > 0$, there is no $o(\log n/(1 + \varepsilon))$ -approximate private algorithm with privacy parameter ε .*

Proof. The proofs of (a) and (b) are immediate by fixing ε to be any constant. The universal algorithm pays at least $\Omega(\log n)$ times the optimum with high probability, thus giving a lower bound of $\Omega(\log n)$ on the expected cost. To see (c), consider a differentially private algorithm \mathcal{A} with privacy parameter ε . Let \mathcal{D} be the distribution on the collection of paths returned by \mathcal{A} when the terminal set is \emptyset . Let X be the subset of vertices corresponding to this distribution in Theorem 1. Let $\mathcal{P} := \{P : c(P[X]) = o(\frac{\log n}{1 + \varepsilon}) \cdot \text{opt}_{ST}(X)\}$; we know $\Pr_{P \leftarrow \mathcal{D}}[P \in \mathcal{P}] \leq \frac{1}{2} \exp(-\varepsilon|X|)$. Let \mathcal{D}' be the distribution on the collection of paths returned by \mathcal{A} when the terminal set is X . By the definition of ε -differential privacy, we know that $\Pr_{P \leftarrow \mathcal{D}'}[P \in \mathcal{P}] \leq \exp(\varepsilon \cdot |X|) \cdot (\frac{1}{2} \exp(-\varepsilon|X|)) \leq 1/2$. Thus with probability at least $1/2$, the differentially private algorithm returns a collection of path of cost $\Omega\left(\frac{\log n}{1 + \varepsilon}\right) \cdot \text{opt}_{ST}(X)$, implying the lower bound.

Note that the statement of Theorem 1 is much stronger than what is needed to prove the universal lower bounds. The proof of part (c) of the above corollary illustrates our observation that showing strong lower bounds for universal problems imply lower bounds for privacy problems. This holds more generally, and we explore this more in Section 3. We now prove Theorem 1.

Proof of Theorem 1: Consider an (n, d, β) expander as in Lemma 1 with degree $d \geq 2^{K(1 + \varepsilon)}$, where K is a large enough constant. The metric (V, c) is the shortest path metric induced by this expander. The root vertex r is an arbitrary vertex in V . We now demonstrate a distribution \mathcal{D}' on terminal sets X such that $\varepsilon|X| \leq C_0 \log n$, for some constant C_0 , and for any fixed collection of paths P ,

$$\Pr_{X \leftarrow \mathcal{D}'} \left[c(P[X]) = o\left(\frac{\log n}{1 + \varepsilon}\right) \text{opt}_{ST}(X) \right] \leq \frac{1}{2} \exp(-C_0 \log n). \quad (2)$$

The lemma below is essentially similar to Yao's lemma [20] used for establishing lower bounds on the performance of randomized algorithms against oblivious adversaries; its proof is omitted.

Lemma 4. *Existence of a distribution \mathcal{D}' satisfying (2) proves Thm 1.*

The distribution \mathcal{D}' is defined as follows. Recall that the girth and the diameter of G are denoted by g and Δ respectively, and both are $\Theta\left(\frac{\log n}{\log d}\right)$. Consider a random walk q of t -steps in G , where $t = g/3$, and let X be the set of distinct vertices in the random walk. This defines the distribution on terminal sets. Note that each X in the distribution has size $|X| = O(\log n / \log d)$. We define C_0 later to be a constant independent of d , and thus since d is large enough, $\varepsilon|X| \leq C_0 \log n$.

Fix a collection of paths P . Since we use the shortest path metric of G , we may assume that P is a collection of paths in G as well. Let (v, v_1) be the first edge on the path p_v , and let $F := \{(v, v_1) : v \in V\}$ be the collection of all these first edges. The following is the crucial observation which gives us the lower bound. Call a walk $q = (u_1, \dots, u_t)$ on t vertices *good* if at most $t/8$ of the edges of the form (u_i, u_{i+1}) are in F , and it contains at least $t/2$ distinct vertices.

We are now ready to complete the proof using the lemma below.

Lemma 5. *Let G be an (n, d, β) expander where d is a large constant ($\geq 2^{100}$, say) and $\beta = \frac{2}{\sqrt{d}}$. Suppose we mark an arbitrarily chosen subset of n edges in G as bad. Then the probability that a t step random walk contains at most $t/8$ bad edges and covers at least $t/2$ distinct vertices is at least $(1 - d^{-\Omega(t)})$.*

Lemma 6. *Let q be a good walk of length $t = g/3$ and let X be the set of distinct vertices in q . Then $c(P[X]) = \Omega(|X|g)$.*

Proof. Let X' be the vertices in X which do not traverse edges in F in the random walk q . Thus $|X'| \geq |X| - 2t/8 \geq |X|/2$. We now claim that $c(P[X']) \geq |X'|g/3$ which proves the lemma. For every $u \in X'$, let p'_u be the first $g/3$ edges in the path p_u (if p_u 's length is smaller than $g/3$, $p'_u = p_u$). All the p'_u 's are vertex disjoint: if p'_u and p'_v intersect then the union of the edges in p'_u, p'_v and the part of the walk q from v to u contains a cycle of length at most g contradicting that the girth of G is g . Thus, $c(P[X'])$, which is at least $c(\bigcup_{u \in X'} p'_u) \geq |X'|g/3 \geq |X|g/6$.

Call the set of edges F *bad*; note that the number of bad edges is at most n . Lemma 5 implies that the probability a t -step random walk is good is at least $(1 - d^{-\Omega(t)})$. Observe that this expression is $(1 - \exp(-C_0 \log n))$ for a constant C_0 independent of d . Furthermore, whenever q is a good walk, the set of distinct vertices X in q are at least $t/2$ in number; therefore $\text{opt}_{ST}(X) \leq t + \Delta = \Theta(|X|)$ since one can always connect X to r by travelling along q and then connecting to r . On the other hand, Lemma 6 implies that $c(P[X]) = \Omega(|X|g) = \Omega\left(\frac{\log n}{\log d}\right) \cdot \text{opt}_{ST}(X) = \Omega\left(\frac{\log n}{1+\varepsilon}\right) \cdot \text{opt}_{ST}(X)$, by our choice of d . This gives that

$$\Pr_{X \leftarrow \mathcal{D}'} [c(P[X]) \leq o\left(\frac{\log n}{1+\varepsilon}\right) \text{opt}_{ST}(X)] \leq \frac{1}{2} \exp(-C_0 \log n)$$

where C_0 is independent of d . Thus, \mathcal{D}' satisfies (2), implying, by Lemma 4, Theorem 1. \square

2.2 Traveling Salesman Problem

We now show an $\Omega(\log n)$ lower bound for the traveling salesman problem. In contrast to our result for the Steiner tree problem, the TSP result is slightly weaker in that it precludes the existence of $o(\log n)$ -approximate private algorithms for arbitrarily small constant privacy parameters only.

We remark here that a lower bound for universal TSP implies a similar lower bound for any universal Steiner tree algorithm which returns a distribution on spanning trees. However, this is not the case when the algorithm returns a collection of paths; in particular, our next theorem below does not imply Theorem 1 even in a weak sense, that is, even if we restrict the parameter ε to be less than the constant ε_0 .

Theorem 2. *There exists a metric space (V, c) and a constant ε_0 , such that for any distribution \mathcal{D} on tours σ of V , there exists a set $X \subseteq V$ of size $\Theta(\log n)$ such that*

$$\Pr_{\sigma \leftarrow \mathcal{D}} [c(\sigma_X) = o(\log n) \cdot \mathbf{opt}_{TSP}(X)] \leq \frac{1}{2} \exp(-\varepsilon_0 |X|)$$

At a high level, the idea as before is to choose as our underlying graph a Ramanujan graph G , and consider the shortest path metric induced by this graph. We show that for any fixed permutation σ of vertices, with high probability a pair of random walks, say q_1, q_2 , has the property that they frequently alternate with respect to σ . Moreover, with high probability, every vertex on q_1 is $\Omega(\log n)$ distance from every vertex in q_2 . The alternation along with large pairwise distance between vertices of q_1 and q_2 implies that on input set defined by vertices of q_1 and q_2 , the cost of the tour induced by σ is $\Omega(\log n)$ times the optimal cost.

As stated in the Introduction, Gorodezky et al. [6] also consider the shortest path metric on Ramanujan expanders to prove their lower bound on universal TSP. However, instead of taking clients from two independent random walks, they use a single random walk to obtain their set of ‘bad’ vertices. Seemingly, our use of two random walks makes the proof easier, and allows us to make a stronger statement: the RHS in the probability claim in Theorem 2 is exponentially small in $|X|$, while [6] implies only a constant. This is not sufficient for part (c) of the following corollary.

As in the case of Steiner tree problem, we get the following corollaries of the above theorem.

Corollary 2. (a) *There is no $o(\log n)$ -approximate universal TSP algorithm.*
 (b) *There is no $o(k)$ -approximate universal TSP algorithm where k is the size of the terminal set.* (c) *There exists $\varepsilon_0 > 0$ such that there is no $o(\log n)$ -approximate private algorithm with privacy parameter at most ε_0 .*

3 Strong Universal Lower Bounds Imply Privacy Lower Bounds

Suppose Π is a minimization problem whose instances are indexed as tuples (I, X) . The first component I represents the part of the input that is accessible to the algorithm (and is public); for instance, in the Steiner tree and the TSP example, this is the metric space (V, c) along with the identity of the root. The second component X is the part of the input which is either unknown beforehand, or corresponds to the private input. We assume that X is a subset of some finite universe $U = U(I)$. In the Steiner tree and TSP example, X is the set of terminals which is a subset of all the vertices. An instance (I, X) has a set of feasible solutions $\mathcal{S}(I, X)$, or simply $\mathcal{S}(X)$ when I is clear from context, and let $\mathcal{S} := \bigcup_{X \subseteq U} \mathcal{S}(X)$. In the case of Steiner trees, $\mathcal{S}(X)$ is the collection of rooted trees containing X ; in the case of TSP it is the set of tours spanning $X \cup r$. Every solution $S \in \mathcal{S}$ has an associated cost $c(S)$, and $\text{opt}(X)$ denotes the solution of minimum cost in $\mathcal{S}(X)$.

We assume that the solutions to instances of Π have the following *projection* property. Given any solution $S \in \mathcal{S}(X)$ and any $X' \subseteq X$, S induces a unique solution in $\mathcal{S}(X')$, denoted by $\pi_{X'}(S)$. For instance, in case of the Steiner tree problem, a rooted tree spanning vertices of X maps to the unique minimal rooted tree spanning X' . Similarly, in the TSP, an ordering of vertices in X maps to the induced ordering of X' . In this framework, we now define approximate universal and differentially private algorithms.

An α -*approximate universal algorithm* for Π takes input I and returns a distribution \mathcal{D} over solutions in $\mathcal{S}(U)$ with the property that for any $X \subseteq U$, $\mathbf{E}_{S \leftarrow \mathcal{D}}[c(\pi_X(S))] \leq \alpha \cdot \text{opt}(I, X)$. An α -*approximate differentially private algorithm* with *privacy parameter* ε for Π takes as input (I, X) and returns a distribution \mathcal{D}_X over solutions in $\bigcup_{Y \supseteq X} \mathcal{S}(Y)$ that satisfies the following two properties. First, for all (I, X) , $\mathbf{E}_{S \leftarrow \mathcal{D}_X}[c(\pi_X(S))] \leq \alpha \cdot \text{opt}(I, X)$. Second, for any set of solutions \mathcal{F} and for any pair of sets X and X' with symmetric difference exactly 1, we have

$$\exp(-\varepsilon) \cdot \Pr_{S \leftarrow \mathcal{D}_{X'}}[S \in \mathcal{F}] \leq \Pr_{S \leftarrow \mathcal{D}_X}[S \in \mathcal{F}] \leq \exp(\varepsilon) \cdot \Pr_{S \leftarrow \mathcal{D}_{X'}}[S \in \mathcal{F}]$$

It is easy to see that any α -approximate universal algorithm is also an α -approximate differentially private algorithm with privacy parameter $\varepsilon = 0$; the distribution $\mathcal{D}_X := \mathcal{D}$ for every X suffices. We now show a converse relation: lower bounds for universal algorithms with a certain additional property imply lower bounds for private algorithms as well. We make this precise.

Fix $\rho : [n] \rightarrow [0, 1]$ to be a non-increasing function. We say that an (α, ρ) *lower bound* holds for universal algorithms if there exists I with the following property. Given any distribution \mathcal{D} on $\mathcal{S}(U)$, there exists a subset $X \subseteq U$ such that

$$\Pr_{S \leftarrow \mathcal{D}}[c(\pi_X(S)) \leq \alpha \cdot \text{opt}(I, X)] \leq \rho(|X|) \tag{3}$$

We say that the set X achieves the (α, ρ) lower bound. It is not hard to see that when ρ is a constant function bounded away from 1, an (α, ρ) lower bound is equivalent to an $\Omega(\alpha)$ lower bound on universal algorithms.

Theorem 3. *Suppose there exists a (α, ρ) lower bound for universal algorithms for a problem Π . Then any ε -private algorithm for Π with*

$\varepsilon \leq \varepsilon_0 := \inf_X \frac{1}{|X|} \ln \left(\frac{1}{2\rho(|X|)} \right)$ has an approximation factor of $\Omega(\alpha)$.

Proof. Let I be an instance that induces the (α, ρ) lower bound. Consider the output of a differentially private algorithm \mathcal{A} with privacy parameter $\varepsilon < \varepsilon_0$, on the input pair (I, \emptyset) . Let \mathcal{D} be the distribution on the solution set \mathcal{S} . We first claim that all S in the support of \mathcal{D} lie in $\mathcal{S}(U)$. Suppose not and suppose there is a solution $S \in \mathcal{S}(Z) \setminus \mathcal{S}(U)$, for some $Z \subset U$, which is returned with non-zero probability. By the definition of differential privacy, this solution must be returned with non-zero probability when \mathcal{A} is run with (I, U) , contradicting feasibility since $S \notin \mathcal{S}(U)$.

Thus, \mathcal{D} can be treated as a universal solution for Π . Let X be the set which achieves the (α, ρ) lower bound for \mathcal{D} , and let $\mathcal{F} := \{S \in \mathcal{S}(X) : c(S) \leq \alpha \cdot \text{opt}(I, X)\}$. By the definition of the lower bound, we know that $\Pr_{S \leftarrow \mathcal{D}}[S \in \mathcal{F}] \leq \rho(|X|)$. Let \mathcal{D}' be the output of the algorithm \mathcal{A} when the input is (I, X) . By definition of differential privacy, $\Pr_{S \leftarrow \mathcal{D}'}[S \in \mathcal{F}] \leq \exp(\varepsilon \cdot |X|) \cdot \rho(|X|) \leq 1/2$, from the choice of ε . This shows a lower bound on the approximation factor of any differential private algorithm for Π with parameter $\varepsilon < \varepsilon_0$.

Acknowledgements. We would like to thank Kunal Talwar for posing the question on differentially private Steiner trees at Bellairs workshop on approximation algorithms which started all this work, Bruce Shepherd, Adrian Vetta, and Mohit Singh for inviting DC to the workshop, and Anupam Gupta for his valuable comments on a previous draft.

References

1. Archer, A.: Two $O(\log^* k)$ -approximation algorithms for the asymmetric k -center problem. In: Proceedings, MPS Conference on Integer Programming and Combinatorial Optimization (IPCO), pp. 1–14 (2010)
2. Bartal, Y.: On approximating arbitrary metrics by tree metrics. In: ACM Symp. on Theory of Computing (STOC), pp. 161–168 (1998)
3. Bhalgat, A., Chakrabarty, D., Khanna, S.: Optimal lower bounds for universal and differentially private steiner trees and tsp. Technical report, <http://arxiv.org/abs/1011.3770>
4. Dwork, C.: Differential privacy. In: Proceedings, International Colloquium on Automata, Languages and Processing, pp. 1–12 (2006)
5. Fakcharoenphol, J., Rao, S., Talwar, K.: A tight bound on approximating arbitrary metrics by tree metrics. In: ACM Symp. on Theory of Computing (STOC), pp. 448–455 (2003)

6. Gorodezky, I., Kleinberg, R.D., Shmoys, D.B., Spencer, G.: Improved lower bounds for the universal and a priori tsp. In: Proceedings, International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pp. 178–191 (2010)
7. Gupta, A., Hajiaghayi, M., Räcke, H.: Oblivious network design. In: Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 970–979 (2006)
8. Gupta, A., Ligett, K., McSherry, F., Roth, A., Talwar, K.: Differentially private approximation algorithms. In: Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1106–1125 (2010)
9. Hajiaghayi, M., Kleinberg, R., Leighton, F.T.: Improved lower and upper bounds for universal tsp in planar metrics. In: Proceedings, ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 649–658 (2006)
10. Hoory, S., Linial, N., Wigderson, A.: Expander graphs and their applications. *Bull. of the Amer. Soc.* 43(4), 439–561 (2006)
11. Imai, M., Waxman, B.M.: Dynamic steiner tree problem. *SIAM J. Discrete Math.* 4(3), 369–384 (1991)
12. Jia, L., Lin, G., Noubir, G., Rajaraman, R., Sundaram, R.: Universal approximations for tsp, steiner tree, and set cover. In: ACM Symp. on Theory of Computing (STOC), pp. 386–395 (2005)
13. Leighton, F.T., Rao, S.: An approximate max-flow min-cut theorem for uniform multicommodity flow problems with application to approximation algorithms. In: Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS), pp. 422–431 (1988)
14. Linial, N., London, E., Rabinovich, Y.: The geometry of graphs and some of its algorithmic applications. *Combinatorica* 15(2), 215–246 (1995)
15. Lubotzky, A., Phillips, R., Sarnak, P.: Ramanujan graphs. *Combinatorica* 4, 261–277 (1988)
16. Moitra, A.: Approximation algorithms for multicommodity-type problems with guarantees independent of the graph size. In: Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS), pp. 3–12 (2009)
17. Moitra, A., Leighton, F.T.: Extensions and limits to vertex sparsification. In: ACM Symp. on Theory of Computing (STOC), pp. 47–56 (2010)
18. Panigrahy, R., Vishwanathan, S.: An $O(\log^* n)$ approximation algorithm for the asymmetric p-center problem. *J. Algorithms* 27(2), 259–268 (1998)
19. Talwar, K.: Problem 1. In: Open Problem in Bellairs Workshop on Approximation Algorithms, Barbados (2010), <http://www.math.mcgill.ca/~vetta/Workshop/openproblems2.pdf>
20. Yao, A.C.-C.: Probabilistic computations: Towards a unified measure of complexity. In: Proceedings, IEEE Symposium on Foundations of Computer Science (FOCS), pp. 222–227 (1977)

Social Welfare in One-Sided Matching Markets without Money

Anand Bhalgat*, Deeparnab Chakrabarty, and Sanjeev Khanna**

Department of Computer and Information Science,
University of Pennsylvania

bhalgat@seas.upenn.edu, deepc@seas.upenn.edu, sanjeev@cis.upenn.edu

Abstract. We study social welfare in one-sided matching markets where the goal is to efficiently allocate n items to n agents that each have a complete, private preference list and a unit demand over the items. Our focus is on allocation mechanisms that do not involve any monetary payments. We consider two natural measures of social welfare: the *ordinal welfare factor* which measures the number of agents that are at least as happy as in some unknown, arbitrary benchmark allocation, and the *linear welfare factor* which assumes an agent's utility linearly decreases down his preference lists, and measures the total utility to that achieved by an optimal allocation.

We analyze two matching mechanisms which have been extensively studied by economists. The first mechanism is the random serial dictatorship (RSD) where agents are ordered in accordance with a randomly chosen permutation, and are successively allocated their best choice among the unallocated items. The second mechanism is the probabilistic serial (PS) mechanism of Bogomolnaia and Moulin [8], which computes a fractional allocation that can be expressed as a convex combination of integral allocations. The welfare factor of a mechanism is the infimum over all instances. For RSD, we show that the ordinal welfare factor is asymptotically $1/2$, while the linear welfare factor lies in the interval $[\frac{526}{1000}, \frac{2}{3}]$. For PS, we show that the ordinal welfare factor is also $1/2$ while the linear welfare factor is roughly $2/3$. To our knowledge, these results are the first non-trivial performance guarantees for these natural mechanisms.

1 Introduction

In the one-sided matching market problem¹, the goal is to efficiently allocate n items, I , to n unit-demand agents, A , with each agent a having a complete and private preference list \geq_a over these items. The problem arises in various applications such as assigning dormitory rooms to students, time slots to users of a common machine, organ allocation markets, and so on. Since the preferences

* Supported in part by NSF Awards CCF-0635084 and IIS-0904314.

¹ In the literature, the problem has been alternately called the *house allocation* or *assignment* problem.

are private, we focus on *truthful* (strategyproof) mechanisms in which agents do not have an incentive to misrepresent their preferences. One class of such mechanisms involve monetary compensations/payments among agents. However, in many cases (e.g., in the examples cited above), monetary transfer may be infeasible due to reasons varying from legal restrictions to plain inconvenience. Hence, we focus on truthful mechanisms without money.

A simple mechanism for the one-sided matching problem is the following: agents arrive one-by-one according to a fixed order σ picking up their most preferred unallocated item. This is called as a *serial dictatorship* mechanism. The *random serial dictatorship* (RSD) mechanism picks the order σ uniformly at random among all permutations. Apart from being simple and easy to implement, RSD has attractive properties: it is truthful, fair, anonymous/neutral, non-bossy², and returns a Pareto optimal allocation. In fact, it is the only truthful mechanism with the above properties [26], and there is a large body of economic literature on this mechanism (see Section 1.2).

Despite this, an important question has been left unaddressed: how *efficient* is this mechanism? To be precise, what is the guarantee one can give on the social welfare obtained by this algorithm when compared to the optimal social welfare? As computer scientists, we find this a natural and important question, and we address it in this paper.

The usual recourse to measure the social welfare of a mechanism is to *assume* the existence of *cardinal* utilities u_{ij} of agent i for item j with the semantic that agent i prefers item j to ℓ iff $u_{ij} > u_{i\ell}$. A mechanism has *welfare factor* α if for every instance the utility of the matching returned is at least α times that of the optimum utility matching. There are a couple of issues with this. Firstly, nothing meaningful can be said about the performance of RSD if the utilities are allowed to be arbitrary. This is because the optimum utility matching might be arising due to one particular agent getting one particular item (a single edge), however with high probability, any random permutation would lead to another agent getting the item and lowering the total welfare by a lot³. Secondly, the assumption of cardinal utilities inherently ties up the performance of the algorithm with the ‘cardinal numbers’ involved; the very quantities whose existence is only an assumption. Rather, what is needed is an *ordinal* scale of analyzing the quality of a mechanism; a measure that depends only on the order/preference lists of the agents rather than the precise utility values.

In this paper, we propose such a measure which we call the *ordinal social welfare* of a mechanism. Given an instance of items and agents with their preference lists, we assume that there exists some benchmark matching M^* , unknown to the mechanism. We stress here this can be *any* matching. We say that the *ordinal welfare factor* of a (randomized) mechanism is α , if for any instance and

² A mechanism is neutral if the allocation of items doesn’t change with renaming, and is non-bossy if no agent can change his preference so that his allocation remains unchanged while someone else’s changes.

³ The reader may notice similarities of RSD with online algorithms for bipartite matching problems. We elaborate on the connection in Section 2.2.

every matching M^* , at least αn agents (in expectation) get an item which they prefer at least as much as what they get in M^* .

A discussion of this measure is in order. Firstly, the measure is *ordinal* and is well defined whenever the utilities are expressed via preference lists. Secondly, the notion is independent of any ‘objective function’ that an application might give rise to since it measures the ordinal social welfare with respect to any desired matching. One disadvantage of the concept is that it is global: it counts the fraction of the *total* population which gets better than their optimal match. In other words, if everyone is ‘happy’ in the benchmark matching M^* , then a mechanism with the ordinal welfare factor α will make an α fraction of the agents happy. However if M^* itself is *inefficient*, say only 1% of the agents are ‘happy’ in M^* , then the ordinal welfare factor does not say much. For instance, it does not help for measures like “maximize number of agents getting their first choice”, for in some instances, this number could be tiny in any M^* . Furthermore, it does not say anything about the “fairness” of the mechanism, e.g. a mechanism may have the ordinal welfare factor close to 1, but there may exist an agent who is almost always allocated an item that he prefers less than M^* . Finally, we observe that the ordinal welfare factor of any mechanism, even ones which know the true preference lists, cannot be larger than $1/2$. The reason for this is that the allocation must be competitive with respect to all benchmark matchings simultaneously, and it can be seen (Theorem 8) that in the instance when all agents have the same preference list, if M^* is chosen to be a random allocation, then no mechanism can have an ordinal welfare factor better than $1/2$. Our first result is that the ordinal welfare factor of RSD is in fact asymptotically $1/2$.

Theorem 1. *The ordinal welfare factor of RSD is at least $1/2 - o(1)$.*

Till now we have focussed on the RSD mechanism since it is a simple (and essentially unique) truthful mechanism for the matching market problem. A mechanism is called truthful if misrepresenting his preference list doesn’t strictly increase the total *utility* of an agent, where the utility is defined as the cardinal utility obtained by the agent on getting his allocated item. However, when the utilities of agents are represented as preference lists, one needs a different definition. In light of this, Bogomolnaia and Moulin [8] proposed a notion of truthfulness based on the *stochastic dominance*: for an agent a random allocation rule stochastically dominates another if the probability of getting one of his top k choices in the first rule is at least that in the second, for any k . A mechanism is called (weakly) truthful if no agent can obtain a stochastically dominating allocation by misreporting his preference list. With this definition, the authors propose a mechanism called the *probabilistic serial* (PS) algorithm, and prove that it is weakly truthful; the mechanism is illustrated in Section 1.1.

PS and RSD are incomparable and results on RSD do not a priori imply those for PS, nevertheless, PS has an ordinal welfare factor of $1/2$ as well.

Theorem 2. *The ordinal welfare factor of PS algorithm is at least $1/2$.*

Ordinal Welfare Factor and Popular Matchings Our notion of ordinal welfare factor is somewhat related to the notion of *popular* matchings [14, 31, 21]. Given

preference lists of agents, a matching M is said to be more popular than M' if the number of agents getting *strictly* better items in M is at least the number of agents getting strictly better items in M' . A matching is popular if no other matching is more popular than it. Thus while comparing a matching M to M' , the notion of popular matchings distinguishes between agents that prefer M and agents that are neutral, unlike in the case of ordinal welfare factor.

It can be easily seen that any popular matching has an ordinal welfare factor of at least $1/2$, however, (a) not every input instance has a popular matching, and (b) no truthful algorithms are known to compute them when they exist. A few modified measures such as unpopularity factor, unpopularity margin and popular mixed matching have also been studied in the literature [22,17,21].

Linear Utilities. We also analyze the performance of RSD and PS mechanisms when agents' utilities are linear - arguably, one of the most commonly studied special case of cardinal utilities. In this model, we assume that the utility for an agent for his i^{th} preference is $\frac{n-i+1}{n}$. Observe that *any* serial dictatorship mechanism achieves a welfare of at least $(n+1)/2$ since the agent at step t gets his t^{th} choice or better, giving him a utility of at least $(1 - (t-1)/n)$. How much better does RSD do? Intuitively, one would expect the worst case instance would be one where each agent gets one of his top $o(n)$ choices; that would make the optimum value $n - o(n)$. We call such instances as *efficient* instances since there is an optimum matching where every one gets their (almost) best choice. We show that for efficient instances, RSD's utility is at least $\frac{2n}{3} - o(n)$, and there exists instances where RSD does no better. These bounds hold for PS as well.

Theorem 3. *With linear utilities and efficient instances, RSD has linear welfare at least $2/3 - o(1)$, and there exist efficient instances for which this is tight.*

Theorem 4. *With linear utilities and efficient instances, PS has linear welfare at least $2/3 - o(1)$, and there exist efficient instances for which this is tight.*

The following theorem summarizes our results on general instances, and we refer the reader to the full version of this paper [6] for its proof.

Theorem 5. *On general instances, the linear welfare factors of RSD and PS algorithms are at least 0.526 and 0.6602 respectively.*

Extensions. We consider two extensions to our model and focus on the performance of RSD, leaving that of PS as an open direction. In the first, we let the preference lists be incomplete. The proof of Theorem 1 implies that the ordinal welfare factor of RSD remains unchanged. For linear utilities, we generalize the definition as follows: for an agent with a preference list of length ℓ , the i^{th} choice gives him a utility of $(\ell - i + 1)/\ell$. We show that RSD doesn't perform very well.

Theorem 6. *For linear utilities, RSD gets at least $\tilde{\Omega}(n^{-1/3})$ fraction of the social optimum. Furthermore, there are instances, where the welfare of RSD is at most $\tilde{O}(n^{-1/3})$ fraction of the social optimum.*

In the second extension, we let the demand of an agent be for sets of size K or less, for some $K \geq 1$. Agents now arrive and pick their best ‘bundle’ among the unallocated items. The ordinal welfare factor of a mechanism is now α if at least an α fraction of agents get a bundle that is as good (assuming there is a complete order on the set of bundles) as what they got in an arbitrary benchmark allocation. We show that RSD has ordinal welfare factor $\Theta(1/K)$.

Theorem 7. *In the case when each agent has a maximum demand of K items, the ordinal welfare factor of RSD is $\Theta(1/K)$.*

1.1 Preliminaries

Utility Models, Truthful Mechanisms, Welfare Factors. As stated above, we consider two models for utilities of agents. In the *cardinal utility model*, each agent a has a utility function $u_a : I \rightarrow \mathbb{R}_{\geq 0}$, with the property that $j >_a \ell$ iff $u_a(j) > u_a(\ell)$. Given a distribution on the matchings, the utility of agent a is $u_a(M) := \sum_{M \in \mathcal{M}} p(M) u_a(M(a))$, where $p(M)$ is the probability of matching M . In this paper, we focus on the special case of *linear utility model* where the i^{th} ranked item for any agent a gives him a utility of $(1 - (i - 1)/n)$. We call an instance *efficient*, if there is a matching which matched every agent to an item in his top $o(n)$ (for concreteness, let’s say this is $n^{1/5}$) choices. In the *ordinal utility model*, each agent a represents his utility only via his complete preference list \succeq_a over the items. A mechanism \mathcal{A} is *truthful* if no agent can misrepresent his preference and obtain a better item. In the cardinal utility model this implies that for all agents a and utility functions u_a, u'_a , we have $u_a(M) \geq u_a(M')$ where $M = \mathcal{A}(u_1, \dots, u_n)$ and $M' = \mathcal{A}(u_1, \dots, u'_a, \dots, u_n)$. A mechanism has *linear welfare factor* of α if for all instances the (expected) sum of linear utilities of agents obtained from the allocation of the mechanism is at least α times the optimal utility allocation for that instance. A mechanism has *ordinal welfare factor* of α if for all instances, and for *all matchings* M^* , at least α fraction of agents (in expectation) get an item at least as good as that allocated in M^* .

The Probabilistic Serial Mechanism. The *probabilistic serial* (PS) mechanism was suggested by Bogomolnaia and Moulin [8]. The mechanism fractionally allocates items to agents over multiple phases, we denote the fraction of the item i allocated to an agent a by $x(a, i)$. These fractions are such that $\sum_{a \in A} x(a, i) = \sum_{i \in I} x(a, i) = 1$ for all agents a and items i . Thus this fractional allocation defines a distribution on integral matchings. Initially, $x(a, i) = 0$ for every agent a and item i . We say that an item i is allocated if $\sum_{a \in A} x(a, i) = 1$, otherwise we call it to be available. The algorithm grows $x(a, i)$ ’s in phases, and in each phase one or more items get completely allocated. During a phase of the algorithm, each agent a grows $x(a, i)$ at the rate of 1 where i is his best choice in the set of available items. The current phase completes and the new phase starts when at least one item that was available in the current phase, gets completely allocated. The algorithm continues until all items are allocated.

We make a few observations about the above algorithm which will be useful in our analysis: (a) the algorithm terminates at time $t = 1$, at which time all agents

are fractionally allocated one item, that is, $\sum_{i \in I} x(a, i) = 1$, (b) any phase lasts for time at least $1/n$ and at most 1, and (c) by time $< j/n$ for any $1 \leq j \leq n$, at most $(j - 1)$ phases are complete.

1.2 Other Related Work

There is a huge amount of literature on matching markets starting with the seminal paper of Gale and Shapley [13], see [24,25,2] for detailed surveys. The one-sided matching market design problem was first studied by Hylland and Zeckhauser [18] who propose a mechanism to find a distribution on matchings via a market mechanism. Their mechanism returns Pareto optimal, envy-free solutions, but is not truthful. Zhou [27], showed that there can be no truthful mechanism which is anonymous/neutral and satisfies *ex ante* Pareto optimality. Svensson [26] showed that serial dictatorship mechanisms are the only truthful mechanisms which are (ex post) Pareto optimal, non bossy, and anonymous.

The study of mechanisms with ordinal utilities for this problem was started by Bogomolnaia and Moulin [8]. The PS mechanism was proposed in an earlier paper by Cres and Moulin [11]. Following the work of [8], there was a list of work characterizing stochastic dominance [19], and generalizing it to the case of incomplete preference lists [20], and to multiple copies of items [10]. The study of mechanism design without money has also been of recent interest in the computer science community, see e.g. [23,5,12,16].

2 Ordinal Welfare Factor of RSD and PS Mechanisms

In this section, we prove Theorems 1 and 2. We first show that the ordinal welfare factor of *any* mechanism is at most $1/2$ in the instance where every agent has the same preference list.

Theorem 8. *If every agent has the same preference list $(1, 2, \dots, n)$, then the ordinal welfare factor of any mechanism is at most $1/2 + 1/2n$.*

Proof. A mechanism returns a probability distribution on matchings which we will interpret as a distribution of permutations. Let \mathcal{D} be that distribution. We choose the benchmark matching M^* to be a random perfect matching. It suffices to show that for any fixed permutation $\sigma \in \mathcal{D}$, the expected number of agents a such that $\sigma(a) \leq \pi(a)$ is $(n + 1)/2$. Since π is chosen uniformly at random, the probability that $\pi(a) < \sigma(a)$ is precisely $(\sigma(a) - 1)/n$, and so the expected number of happy people for the permutation σ is $(n + 1)/2$.

2.1 Ordinal Welfare Factor of RSD

In this section, we prove Theorem 1. Let M^* be the unknown benchmark matching. We call an agent a *dead* at time t if he hasn't arrived yet and all items as good as $M^*(a)$ in his preference list has been allocated. Let D_t be the expected number of dead agents at time t . Let ALG_t be the expected number of agents who

get an item as good as their choice in M^* by time t . From the above definition, we get

$$\text{ALG}_{t+1} - \text{ALG}_t = 1 - \frac{D_t}{n-t} \quad (1)$$

We will now bound D_t from above which along with (1) will prove the theorem.

Lemma 1. $D_t \leq \frac{(t+2)(n-t)}{n+1}$ for $1 \leq t \leq n$.

Before proving the lemma, note that adding (1) for $t = 1$ to $n-1$ gives $\text{ALG}_n - \text{ALG}_1 \geq \sum_{t=1}^{n-1} \left(1 - \frac{t+2}{n+1}\right)$, implying $\text{ALG}_n - \text{ALG}_1 \geq n/2 - 2n/n$. This proves that the ordinal welfare factor of RSD is at least $1/2 - o(1)$ proving Theorem 1.

Proof. Let us start with a few definitions. For an item i and time t , let $\text{ALL}_{i,t}$ be the event that item i is allocated by time t . For an agent a and time t , let $\text{LATE}_{a,t}$ be the event that a arrives after time t . The first observation is this: if an agent a is dead at time t , then the event $\text{ALL}_{M(a),t}$ and $\text{LATE}_{a,t}$ must have occurred. Therefore we get

$$D_t \leq \sum_{a \in A} \Pr[\text{ALL}_{M(a),t} \wedge \text{LATE}_{a,t}] \quad (2)$$

Note that $\Pr[\text{LATE}_{a,t}]$ is precisely $(1 - t/n)$. Also, note that $\sum_{i \in I} \Pr[\text{ALL}_{i,t}] = t$. This is because all agents are allocated *some* item. Now suppose *incorrectly* that $\text{ALL}_{M(a),t}$ and $\text{LATE}_{a,t}$ were independent. Then, (2) would give us

$$D_t \leq \left(1 - \frac{t}{n}\right) \sum_{a \in A} \Pr[\text{ALL}_{M(a),t}] = \left(1 - \frac{t}{n}\right) \sum_{i \in I} \Pr[\text{ALL}_{i,t}] = \frac{t(n-t)}{n} \quad (3)$$

which is at most the RHS in the lemma. However, the events are not independent, and one can construct examples where the above bound is indeed incorrect. To get the correct bound, we need the following claim.

Claim.

$$\frac{\Pr[\text{ALL}_{M(a),t} \wedge \text{LATE}_{a,t}]}{(n-t)} \leq \frac{\Pr[\text{ALL}_{M(a),t+1} \wedge \overline{\text{LATE}_{a,t+1}}]}{(t+1)}$$

Proof. This follows from a simple charging argument. Fix a relative order of all agents other than a and consider the orders obtained by placing a in the n possible positions. Observe that if the event $\text{ALL}_{M(a),t} \wedge \text{LATE}_{a,t}$ occurs at all, it occurs exactly $(n-t)$ times when a 's position is $t+1$ to n . Furthermore, crucially observe that if the position of a is 1 to $t+1$, the item $M(a)$ will still be allocated. This is because the addition of a only leads to worse choices for agents following him and so if $M(a)$ was allocated before, it is allocated even now. This proves that for every $(n-t)$ occurrences of $\text{ALL}_{M(a),t} \wedge \text{LATE}_{a,t}$, we have $(t+1)$ occurrences of the event $\text{ALL}_{M(a),t+1} \wedge \overline{\text{LATE}_{a,t+1}}$. The claim follows as it holds for every fixed relative order of other agents.

Now we can finish the proof of the lemma. From Claim [2.1](#), we get

$$\frac{t+1}{n-t} \cdot \Pr[\text{ALL}_{M(a),t} \wedge \text{LATE}_{a,t}] \leq \Pr[\text{ALL}_{M(a),t+1}] - \Pr[\text{ALL}_{M(a),t+1} \wedge \text{LATE}_{a,t+1}]$$

Taking the second term of the RHS to the LHS, adding over all agents, and invoking [\(2\)](#), we get

$$\frac{t+1}{n-t} \cdot D_t + D_{t+1} \leq t+1 \tag{4}$$

Using the fact that $D_{t+1} \geq D_t - 1$ (the number of dead guys cannot decrease by more than 1), and rearranging, proves the lemma.

2.2 RSD and Online Bipartite Matching

In this section, we highlight the relation between RSD and algorithms for online bipartite matching. In fact, the analysis of RSD above can be seen as a generalization of online bipartite matching algorithms.

In the online bipartite matching problem, vertices of one partition (think of them as agents) are fixed while vertices of the other partition (think of them as items) arrive in an adversarial order. Karp, Vazirani and Vazirani [\[19\]](#) gave the following algorithm (KVV) for the problem: fix a *random* ordering of the agents, and when an item arrives give it to the first unmatched agent in this order. They proved [\[1\]](#) that the expected size of the matching obtained is at least $(1 - 1/e)$ times the optimum matching. The KVV theorem can be ‘flipped around’ to say the following. Suppose each agent has the preference list which goes down its desired items in the order of entry of items. Then, if agents arrive in a random order and pick their best, unallocated, desired item, in expectation an $(1 - 1/e)$ fraction of agents are matched. That is, if we run RSD on this instance (with incomplete lists), an $(1 - 1/e)$ fraction of agents will get an item.

The above result does not a priori imply an analysis of RSD, the reason being that in our problem an agent a , when he arrives, is allocated an item even if that item is *worse* than what he gets in the benchmark matching M^* . This might be bad since the allocated item could be ‘good’ item for agents to come. In particular, if the order chosen is not random but arbitrary, the performance of the algorithm is quite bad; in contrast, the online matching algorithm still has a competitive ratio of $1/2$. Nevertheless, similar techniques prove both the results and our analysis can be tailored to give a proof of the online bipartite matching result (See [\[6\]](#) for details).

2.3 Ordinal Welfare Factor of PS

In this section, we prove Theorem [2](#). We suggest the reader to refer to the algorithm and its properties as described in Section [1.1](#). In particular, we will use the following observation.

⁴ In 2008, a bug was found in the original extended abstract of [\[19\]](#), but was soon resolved. See [\[15,74\]](#) for discussions and resolutions.

Observation 1: By time $< j/n$, for any $1 \leq j \leq n$, at most $(j - 1)$ items are completely allocated.

Let M^* be the unknown benchmark matching. For an agent a , let t_a be the time at which the item $M^*(a)$ is completely allocated. Observe that the probability agent a gets an item $M^*(a)$ or better is precisely t_a , since till this time $x(a, i)$ increases for items $i \geq_a M^*(a)$. Summing up all agents, we see that the ordinal welfare factor of the PS mechanism is $\sum_a t_a$. The observation above implies at most $(j - 1)$ agents have $t_a < j/n$. So, $\sum_a t_a \geq \sum_{j=1}^n (n - j + 1)/n \geq n/2 + 1/2$. This completes the proof of Theorem 2.

3 Linear Welfare Factor of RSD and PS

In this section, we establish bounds on the linear welfare factor of RSD and PS mechanisms. We first prove Theorem 3 in two lemmas. Recall that an instance is called efficient if there exists a matching in which every agent is matched to an item in his top $o(n)$ choices.

Lemma 2. *When the instance is efficient, the linear welfare factor of RSD is at least $(2/3 - o(1))$.*

Proof. The proof follows from Lemma 1. Let U_t denote the expected utility obtained by time t . Consider the agent coming at time $t + 1$. If he is not dead already, then he will get a utility of at least $(1 - o(1))$ (since the instance is efficient). If he is dead, then he will get a utility of at least $(1 - t/n)$. This is because only t items have been allocated and this agent takes an item $(t + 1)$ th ranked or higher. Therefore,

$$U_{t+1} - U_t \geq \left(1 - \frac{D_t}{n-t}\right) \cdot (1 - o(1)) + \frac{D_t}{n-t} \cdot (1 - t/n) \geq 1 - o(1) - \frac{t}{n} \cdot \frac{D_t}{n-t}$$

Using Lemma 1, we get $U_{t+1} - U_t \geq 1 - o(1) - \frac{t(t+2)}{n(n+1)}$. Summing over all t , we get that the total utility of RSD is at least $(1 - o(1))n - (n/3 + o(n)) = (2/3 - o(1))n$.

The above analysis can be modified via a ‘balancing trick’ to give a strictly better than 50% guarantee for all instances. We refer the reader to [6] for details.

Lemma 3. *When the utilities are linear, there exists an efficient instance for which RSD gets a utility of at most $(2/3 + o(1))n$.*

Proof. Partition n agents and items into t blocks of size n/t each, where $t = n^{1/5}$. We denote the j^{th} block of agents and items by A_j and I_j respectively, and they number from $\left(\frac{(j-1)n}{t} + 1\right)$ to $\frac{jn}{t}$.

We now illustrate the preference lists of agents. Fix an agent a in block A_j . Let he be the k^{th} agent in the block, where $1 \leq k \leq n/t$, i.e. his agent number is $(j-1)n/t + k$. A random set of t^3 items is picked from each of blocks I_1, \dots, I_{j-1} , and these form the first $(j-1)t^3$ items in his preference list, in increasing order

of item number. The item $(j-1)n/t + k$ is his $((j-1)t^3 + 1)^{th}$ choice. His remaining choices are the remaining items considered in increasing order. This completes the description of the preference lists of the agents.

Note that if every agent a is assigned the corresponding item with the same number, then each agent gets one of his top t^4 choices, leading to a utility of at least $(1 - \frac{t^4}{n}) = 1 - o(1)$. So, the instance is indeed efficient. We now show that RSD gets utility at most $2n/3 + o(1)$.

Let σ be a random permutation of the agents. We divide σ into t chunks of n/t agents, with the j^{th} chunk, S_j , consisting of agents $\sigma(\frac{(j-1)n}{t} + 1)$ to $\sigma(\frac{jn}{t})$. Note that with high probability ($\geq (1 - 1/t^3)$), we have that for any block A_j and chunk S_i , $|A_j \cap S_i| \in [(1 - \frac{1}{t^2})\frac{n}{t^2}, (1 + \frac{1}{t^2})\frac{n}{t^2}]$. Since agents prefer items in 'higher' blocks to 'lower' blocks, we claim the following.

Claim. With high probability, at least $(1 - \frac{1}{t^3})$ fraction of the items in the first i blocks have been allocated after arrival of first i chunks. (Proof omitted; see [6].)

Now we are ready to analyze RSD. Consider the $(i+1)^{th}$ chunk of agents. With high probability, there are at least $\frac{n}{t^2}(1 - \frac{1}{t^2})$ agents from each block A_1, \dots, A_i in S_{i+1} . Since only in/t^3 items remain from the first i block of items, at least $\frac{in}{t^2}(1 - \frac{1}{t^2}) - \frac{in}{t^3}$ of these agents must get an item from blocks $(i+1)$ or higher. However, this gives them utility at most $(1 - \frac{in/t}{n}) \geq 1 - i/t$. That is, the *drop* in their utility to what they get in the optimum is at least i/t . Summing the total drop over all agents and all chunks, we get that the difference between RSD and the optimum is at least

$$\sum_{i=1}^t \frac{in}{t^2} (1 - \frac{1}{t}) \frac{i}{t} = (1 - o(1)) \frac{n}{t^3} \sum_{i=1}^n i^2 = n/3$$

Therefore, the social welfare of RSD is at most $(2/3 + o(1))n$.

Linear Welfare Factor of PS Mechanism We establish the lower bound in this abstract, and the upper bound instance, which is similar to that for RSD, can be found in [6]. As in the case of RSD, we focus on efficient instances.

Lemma 4. *For efficient instance, the linear welfare factor of PS $\geq 2/3 - o(1)$.*

Proof. Let o_a denote the utility obtained by agent a in the utility optimal matching. Since the instance is efficient, $o_a = 1 - o(1)$ for all agents a .

Consider the j^{th} phase of PS, and suppose it lasts for time Δ_j . Observation 1 implies that $\sum_{j \leq \ell} \Delta_j \geq \ell/n$. Furthermore, in phase j , at least $(n-j+1)$ agents obtain utility at a rate higher than their utility in the optimal matching. This is because at most $(j-1)$ items have been allocated. Also, the remaining $(j-1)$ agents are getting utility at a rate at least $(1 - (j-1)/n)$ since they are growing their $x(a, i)$ on their j^{th} choice or better. So, the total utility obtained by PS is at least $\sum_{j=1}^n \Delta_j \cdot ((n-j+1) \cdot (1 - o(1)) + (j-1) \cdot (1 - \frac{j-1}{n}))$ which evaluates to $\sum_{j=1}^n \Delta_j \left(\frac{n^2 - (j-1)^2}{n} \right) - o(n)$

The above summation is smallest if Δ_1 is as small as possible, modulo which, Δ_2 is as small as possible and so on. Given the constraint on Δ_j 's, we get that this is at least $\sum_{j=1}^n \frac{n^2 - (j-1)^2}{n^2} = 2n/3 - o(n)$.

4 Concluding Remarks

We first give very brief sketches of the proofs of Theorems 6 and 7. Full proofs can be found in 6.

Incomplete Preference Lists. The ordinal welfare factor of RSD remains the same, however, the linear welfare factor of RSD drops to $\tilde{\Theta}(1/n^{1/3})$. This is because some agents can have ‘long’ preference lists and some agents have ‘short’ preference lists, and in a random order the long preference list agents can take away items of the short preference list ones. However, if the lengths of the preference lists of the ‘long agents’ are ‘too long’, they get an item with high enough linear utility. The correct balancing argument gives the $\tilde{\Theta}(\frac{1}{n^{1/3}})$ factor.

Non-unit demands. Note that a single agent’s choice can disrupt the choices of K other agents. Therefore, it is not too difficult to construct an example which shows that the ordinal welfare factor of RSD is $O(1/K)$. On the other hand, by the time t agents arrive, at most Kt agents are disrupted, and so in a random permutation the $(t + 1)$ th agent is unhappy with probability $\leq \frac{(K+1)t}{n-t}$. Integrating, this gives that $\frac{n}{2K} - o(\frac{n}{K})$ agents are happy in expectation.

To conclude, in this paper we studied the social welfare of two well studied mechanisms, RSD and PS, for one-sided matching markets. We focussed on two measures: one was the ordinal welfare factor, and the other was the linear utilities measure. We performed a tight analysis of the ordinal welfare factors of both mechanisms, and the linear welfare factor in the case of efficient instances. An open problem is to perform a tighter analysis of linear welfare factor in general instances. We think the notion of ordinal welfare factor will be useful for other problems as well where the utilities are expressed as preference lists rather than precise numbers. Examples which come to mind are scheduling, voting, and ranking.

References

1. Abdulkadiroglu, A., Sönmez, T.: Ordinal efficiency and dominated sets of assignments. *Journal of Economic Theory* 112, 157–172 (2003)
2. Abdulkadiroglu, A., Sönmez, T.: Matching markets: Theory and practice. Prepared for the Econometric Society World Congress, China (2010), available at first author’s website
3. Abraham, D., Irving, R., Kavitha, T., Melhorn, K.: Popular matchings. *SIAM Journ. of Computing* 37, 1030–1045 (2007)
4. Aggarwal, G., Goel, G., Karande, C., Mehta, A.: Online vertex-weighted bipartite matching and single-bid budgeted allocations. In: *SODA* (2011)

5. Ashlagi, I., Fischer, F., Kash, I., Procaccia, A.D.: Mix and Match. In: Proceedings of the ACM Conference on Electronic Commerce, EC 2010 (2010)
6. Bhargat, A., Chakrabarty, D., Khanna, S.: Social welfare in one-sided matching markets without money. Technical report, <http://arxiv.org/abs/1104.2964>
7. Birnbaum, B., Mathieu, C.: On-line bipartite matching made simple. ACM SIGACT News 39 (2008)
8. Bogomolnaia, A., Moulin, H.: A new solution to the random assignment problem. *Journal of Economic Theory* 100, 295–328 (2001)
9. Bogomolnaia, A., Moulin, H.: A simple random assignment problem with a unique solution. *Economic Theory* 19, 623–635 (2002)
10. Budish, E., Che, Y.K., Kojima, F., Milgrom, P.: Designing random allocation mechanisms: Theory and applications. Unpublished Manuscript available at first author's personal webpage (October 2010)
11. Cres, H., Moulin, H.: Scheduling with opting out: Improving upon random priority. *Operations Research* 49, 565–577 (2001)
12. Dughmi, S., Ghosh, A.: Truthful assignment without money. In: Proceedings of the 11th ACM Conference on Electronic Commerce, EC 2010 (2010)
13. Gale, D., Shapley, L.: College admissions and the stability of marriage. *American Mathematical Monthly* 69, 9–15 (1962)
14. Gardenfors, P.: Match making: assignments based on bilateral preferences. *Behavioral Sciences* 20, 166–173 (1975)
15. Goel, G., Mehta, A.: Online budgeted matching in random input models with applications to adwords. In: Symposium on Discrete Algorithms, SODA (2008)
16. Guo, M., Conitzer, V.: Strategy-proof allocation of multiple items between two agents without payments or priors. In: 9th International Joint Conference on Autonomous Agents and Multi Agent Systems, AAMAS 2010 (2010)
17. Huang, C.-C., Kavitha, T., Michail, D., Nasre, M.: Bounded unpopularity matchings. In: Proceedings of the 11th Scandinavian Workshop on Algorithm Theory (2008)
18. Hylland, A., Zeckhauser, R.: The efficient allocation of individuals to positions. *American Mathematical Monthly* 69, 9–15 (1962)
19. Karp, R., Vazirani, U., Vazirani, V.: An optimal algorithm for online bipartite matching. In: 22nd Annual ACM Symposium on Theory of Computing (1990)
20. Katta, A.K., Sethurman, J.: A solution to the random assignment problem on the full preference domain. *Journal of Economic Theory* 131, 231–250 (2006)
21. Kavitha, T., Mestre, J., Nasre, M.: Popular mixed matchings. *Theoretical Computer Science*, Article in Press (2010)
22. McCutchen, R.M.: The least-unpopularity-factor and least-unpopularity-margin criteria for matching problems with one-sided preferences. In: Laber, E.S., Bornstein, C., Nogueira, L.T., Faria, L. (eds.) LATIN 2008. LNCS, vol. 4957, pp. 593–604. Springer, Heidelberg (2008)
23. Procaccia, A.D., Tennenholtz, M.: Approximate mechanism design without money. In: Proceedings of the 10th ACM Conference on Electronic Commerce, EC 2009 (2009)
24. Roth, A.E., Sotomayor, M.: Two-Sided Matching: A Study on Game-Theoretic Modelling. Cambridge University Press, Cambridge (1990)
25. Sönmez, T., Ünver, U.: Matching, allocation, and exchange of discrete resources. In: Handbook of Social Economics, forthcoming, available at second author's website
26. Svensson, L.: Strategyproof allocation of indivisible goods. *Social Choice and Welfare* 16, 557–567 (1999)
27. Zhou, L.: On a conjecture by gale about one-sided matching problems. *Journal of Economic Theory* 52, 123–135 (1990)

Primal-Dual Schema and Lagrangian Relaxation for the k -Location-Routing Problem*

Tim Carnes¹ and David Shmoys²

¹ Sloan School of Management, MIT
tcarnes@mit.edu

² School of Operations Research & Information Eng. & Dept. of Computer Science,
Cornell University
shmoys@cs.cornell.edu

Abstract. The location-routing problem arises in the context of providing integrated support for logistics in a number of transportation settings, where given a set of requests and potential depot locations, one must simultaneously decide where to locate depots as well as how to route tours so that all requests are connected to an open depot. This problem can be formulated either with specific costs incurred for choosing to open each depot, or with an upper bound k on the number of open depots, which we call the *k-location-routing problem*.

We develop a primal-dual schema and use Lagrangian relaxation to provide a 2-approximation algorithm for the k -location-routing problem; no constant performance guarantee was known previously for this problem. This strengthens previous work of Goemans & Williamson who gave a 2-approximation algorithm for the variant in which there are opening costs, but no limit on the number of depots. We give a new primal-dual algorithm and a strengthened analysis that proves a so-called Lagrangian-preserving performance guarantee. In contrast to the results of Jain & Vazirani for the uncapacitated facility location and k -median problems, our results have the surprising property that our performance guarantee for the k -location-routing problem matches the guarantee for the version in which there are depot opening costs; furthermore, this relies on a simple structural property of the algorithm that allows us to identify the critical Lagrangian value for the opening cost with a single execution of the primal-dual algorithm, rather than invoking a bisection search.

1 Introduction

The location-routing problem arises in the context of providing integrated support for logistics in a number of transportation settings. For example, in providing an air ambulance service, Ontario-based company Ornge must transport a number of non-emergency requests to transfer patients. These requests are typically known at least a day in advance, which provides ample time to determine

* Research supported partially by NSF grants DMS-0732196, CCF-0832782, CCF-1017688.

a cost-effective solution. To serve the requests, a fleet of planes is based at a set of airports, each of which may be used to pick-up and drop-off patients. An important decision is to determine the optimal locations at which to establish bases when providing such a service. For simplicity, we model each request as a single point, which must be included in some tour containing an open base, and the cost of a solution is the cost of opening the specified bases plus the length of all the tours. The *general location-routing problem* is to find the minimum-cost solution. Alternatively, Orngé may wish to constrain that at most k bases are opened, which we call the *k -location-routing problem*. This pair of problems also arise in other applications, such as deciding where to position postal fleet locations, or in Orngé's day-to-day planning operations, where they may wish to restrict that at most k planes are used for the coming day's transfers.

Our main result is a primal-dual 2-approximation algorithm for the k -location-routing problem, which is the best known approximation for this problem. Although no previous constant-factor approximation algorithm appears to have been known for this problem, we note that a simple technique does yield a 4-approximation. Consider a graph with nodes corresponding to the bases and requests, with edges between all pairs of nodes that have costs corresponding to the metric distance. We will add a dummy root node with an edge to each base node of cost zero. The optimal solution must connect all requests to bases, so the cost of the minimum spanning tree in this graph is a lower bound on the optimal value. Also, the optimal solution opens at most k bases, and we can assume that there is only one tour out of each open base, since otherwise we could combine tours by shortcutting and only reduce the cost. This means that when taking the induced subgraph of the optimal solution on just the request nodes, there are k connected components; thus, taking the cheapest set of edges between requests to form k components is also a lower bound, and we can find this set with an abbreviated run of Kruskal's algorithm. If we take this set of edges along with the edges in the minimum spanning tree above, we connect each of the k components to a base, and the cost is within a factor of two of optimal. By shortcutting, we can form tours from each of the k components, and hence have a performance guarantee of 4.

The location-routing problem has been extensively studied in the context of finding exact solutions, including work by Laporte et al. [4,6,7,8,9]. One may also refer to Laporte [5] and Min et al. [10] for surveys of previous work. There is a recent paper of Berger, Coullard & Daskin [1] that finds exact solutions through branch-and-price style techniques. Here there are additional constraints placed on the type of tours allowed, such as limiting their total distance, which allows the total number of variables to remain manageable for a natural IP formulation given small inputs. The general location-routing problem can be approximated using the primal-dual method of Goemans & Williamson [2] for network design problems. However, this approach does not extend to the case where there must be at most k open bases.

The two variants that we have discussed, the general location-routing problem and the k -location-routing problem, can be seen as analogues of two other classic

optimization problems: the uncapacitated facility location problem, and the k -median problem, respectively. For this latter pair of problems, we again must select depots to open, and connect each demand point to an open depot, but instead of connecting the requests by a tour, each request is connected directly to the depot (i.e., by a “star” graph, rather than a cycle). However, for this latter pair, we again have one problem for which there are specified depot opening costs (the uncapacitated facility location problem) and one problem in which we simply have an explicit constraint of k on the number of depots allowed to be opened.

Jain & Vazirani [3] showed a fundamental connection between the design of approximation algorithms for the k -median problem and for the uncapacitated facility location problem. Jain & Vazirani designed an approximation algorithm for the k -median problem, by first designing an algorithm for the uncapacitated facility location problem with a stronger form of performance guarantee, which they called *Lagrangian-preserving*. This property is that the approximation factor they find is still valid even if they first scale the cost of the facilities in the solution found by the approximation factor. The essence of the connection between the problems is very simple: to find a solution to the k -median problem, suppose that we can specify a uniform opening cost λ for each depot such that the algorithm for the uncapacitated facility problem opens exactly k depots; this gives a feasible solution to the k -median problem, and the Lagrangian-preserving property would allow the performance guarantee of one problem to apply to the other. Unfortunately, for the primal-dual algorithm of Jain & Vazirani, such a value λ need not exist, and this required some additional technicalities that resulted in their 3-approximation algorithm for the uncapacitated facility problem degrading to a 6-approximation algorithm for the k -median problem.

Goemans & Williamson [2] gave a 2-approximation algorithm for the general location-routing problem (as part of their more general framework of results for network design problems), but their guarantee is not Lagrangian-preserving. We modify the primal-dual algorithm of Goemans & Williamson to obtain a Lagrangian-preserving 2-approximation algorithm, which allows us to use the framework of Jain & Vazirani to obtain an approximation algorithm for the k -location-routing problem. Furthermore, our primal-dual algorithm does have the property that we can always find a common opening cost λ for which k depots are opened, so that we obtain an approximation algorithm for the k -location variant without any loss in the performance guarantee. Finally, whereas Jain & Vazirani needed a bisection search to hone in on the critical range for λ , we can compute the precise value of λ with one execution of our primal-dual algorithm. In fact, we compute a frontier of good solutions for each possible common opening cost for depots.

2 The k -Location-Routing Problem

When considering finding an optimal routing, it may be the case that we could do much better if the bases were in different locations. The ideal location for

the bases will depend on how the routing is formed, which suggests we should decide the base locations and the routing simultaneously, and leads to the k -location-routing problem. Here we are given a set of bases and a set of requests, that are all contained in a metric space. We must choose at most k bases to open and include each request in a tour that contains an open base. The cost of a solution is simply the metric distance of all the tours. We will consider two different modifications to this problem. First, we consider a fixed-charge model, where instead of restricting there to be k open bases, we will allow any number of bases to be opened, but each base now has an opening cost. Second, in the routing with trees version, instead of having all the requests included in some tour, we simply require that each request be connected to an open base in a tree. We will begin by considering the location-routing problem with fixed-charge bases and routing with trees. A previous result of Goemans & Williamson [2] considers this problem, but we provide a stronger result which yields a Lagrangian-preserving performance guarantee. This allows us to lift this result to the k -location-routing problem, but again where we are routing with trees. Finally, we show how to take our routing with trees results, and produce results for routing with tours. This can be easily done by simply shortcutting the tree and losing a factor of 2, but we show that this can be done with the same performance guarantee. For all the variants we consider, we are able to show a performance guarantees of 2.

Routing with Trees, Fixed-Charge Bases In the input for the location-routing problem, we have a set of potential bases, B , and set of requests R , that coexist in a metric space. Each base, $i \in B$ has an opening cost of f_i . When routing with trees, we simply require that each request has a path to an open base, which can be modeled as a Steiner tree problem on the following graph. For each base $i \in B$ and request $j \in R$ we will have an edge $\{i, j\}$ with cost equal to the metric distance between i and j . Similarly for any two distinct requests, $j, k \in R$, we will have an edge $\{j, k\}$ with cost equal to the metric distance between the two requests. Finally, we will have a root node, r , and for each base $i \in B$ we have an edge $\{r, i\}$ with cost f_i . Thus the vertex set of this graph is $V = \{r\} \cup B \cup R$ and we define the following three edge sets. *Root edges:* $E_r := \{\{r, i\} : i \in B\}$; *base edges:* $E_B := \{\{i, j\} : i \in B, j \in R\}$; *request edges:* $E_R := \{\{j, k\} : j, k \in R, j \neq k\}$. The edge set of our graph is simply the union of the three edge sets defined above: $E = E_r \cup E_B \cup E_R$. Each of the nodes corresponding to a base can be thought of as a Steiner node, so the task is to choose a set of edges such that for each request $j \in R$ there exists a path from r to j . We can formulate this as the following integer program, where $\delta(S)$ is the set of all edges with exactly one endpoint in S .

$$\begin{aligned}
 \text{opt}_{LR} &:= \min \sum_{e \in E} c_e x_e && \text{(LR-IP)} \\
 \text{s.t.} \quad &\sum_{e \in \delta(S)} x_e \geq 1 && \forall S \in \mathcal{S} \quad (1) \\
 &x_e \in \{0, 1\} && \forall e \in E,
 \end{aligned}$$

where $\mathcal{S} := \{S \subseteq B \cup R : S \cap R \neq \emptyset\}$. Now suppose \mathcal{A} is a collection of disjoint subsets in \mathcal{S} . Let us extend our notation so that δ can be applied to a set of subsets with the meaning $\delta(\mathcal{A}) := \bigcup_{S \in \mathcal{A}} \delta(S)$. We now propose the following constraint:

$$\sum_{e \in \delta(\mathcal{A})} x_e \geq |\mathcal{A}| \quad \forall \mathcal{A} \in \mathcal{D}, \quad (2)$$

where \mathcal{D} is the set of all collections of pairwise disjoint subsets in \mathcal{S} . The proof of validity for this constraint is straightforward, and omitted due to space restrictions.

Lemma 1. *The constraints given by (2) are valid; any solution to (LR-IP) satisfies (2).*

Note that (2) implies (I) by taking \mathcal{A} to be each singleton set of a set in \mathcal{S} . Since (2) is valid then clearly the following constraint is also valid.

$$2 \sum_{e \in \delta(\mathcal{A}) \cap E_R} x_e + \sum_{e \in \delta(\mathcal{A}) \setminus E_R} x_e \geq |\mathcal{A}| \quad \forall \mathcal{A} \in \mathcal{D}.$$

We will demonstrate a primal-dual schema based on the LP that makes use of the above constraints.

$$\begin{aligned} \text{opt}_{LRP} &:= \min \sum_{e \in E} c_e x_e && \text{(LR-P)} \\ \text{s.t. } 2 \sum_{e \in \delta(\mathcal{A}) \cap E_R} x_e + \sum_{e \in \delta(\mathcal{A}) \setminus E_R} x_e &\geq |\mathcal{A}| && \forall \mathcal{A} \in \mathcal{D} \quad (3) \\ x_e &\geq 0 && \forall e \in E. \end{aligned}$$

The dual of this LP is as follows:

$$\text{opt}_{LRD} := \max \sum_{\mathcal{A} \in \mathcal{D}} |\mathcal{A}| y_{\mathcal{A}} \quad \text{(LR-D)}$$

$$\text{s.t. } 2 \sum_{\substack{\mathcal{A} \in \mathcal{D}: \\ \delta(\mathcal{A}) \ni e}} y_{\mathcal{A}} \leq c_e \quad \forall e \in E_R \quad (4)$$

$$\sum_{\substack{\mathcal{A} \in \mathcal{D}: \\ \delta(\mathcal{A}) \ni e}} y_{\mathcal{A}} \leq c_e \quad \forall e \in E \setminus E_R \quad (5)$$

$$y_{\mathcal{A}} \geq 0 \quad \forall \mathcal{A} \in \mathcal{D}.$$

The primal-dual schema proposed uses the formulation above, and will build up a set of edges, F , from which the final solution will be chosen. Initially, this set F is empty. At each point in the algorithm, we will maintain a set of clusters, \mathcal{C} , which simply corresponds to the set of connected components in the graph when restricting the edge set to F . Since F is initially empty, the set of clusters

\mathcal{C} is initially the set of all singleton nodes. We will say a cluster is *active* if it contains a request node and does not contain the root node, and otherwise it will be *inactive*. We will maintain \mathcal{A} to be the set of all active clusters in \mathcal{C} , so $\mathcal{A} := \mathcal{C} \cap \mathcal{S}$, and initially \mathcal{A} is the set of all singleton request nodes. All dual variables are implicitly set to zero to begin. The algorithm will always increase at unit rate the value of precisely one dual variable at any moment, until there are no longer any active clusters. In this sense we will keep a notion of time, which is equivalent to the sum of the current values of the dual variables.

At each point in time, we will increase the dual variable associated with \mathcal{A} until the dual constraint for one of the edges becomes tight, at which point the edge is added to F , and the two corresponding clusters are merged. If several edges become tight simultaneously, then we are free to process the edges in an arbitrary order provided that we process request edges in preference to base edges. We continue this process until \mathcal{A} is empty, so that there will no longer be any active clusters.

After all clusters have become inactive (by virtue of all request nodes being connected to the root) we begin the cleanup phase. If the root edge $\{r, i\} \in F$ for some $i \in B$, then we say that base i is *paid for*. We now simply remove all base edges $\{i, j\}$ for $i \in B, j \in R$ for which base i is not paid for. After removing all such base edges, we will call the resulting subset of edges F' .

Algorithm 1. Primal-Dual for Location-Routing

```

 $y, x \leftarrow 0$ 
 $F \leftarrow \emptyset$ 
 $\mathcal{C} \leftarrow \{\{v\} : v \in V\}$ 
 $\mathcal{A} \leftarrow \{\{j\} : j \in R\}$ 
while  $|\mathcal{A}| > 0$  do
  Increase  $y_{\mathcal{A}}$  until a dual constraint becomes tight for edge  $e$ 
   $F \leftarrow F \cup \{e\}$ 
  Remove clusters for endpoints of  $e$  from  $\mathcal{C}$  and add the union
   $\mathcal{A} \leftarrow \mathcal{C} \cap \mathcal{S}$ 
 $F' \leftarrow F \setminus \{\{i, j\} \in E_B : i \in B, \{r, i\} \notin F\};$            /* clean-up */

```

We now make a few general observations about the behavior of the primal-dual algorithm. Note that initially all request edges and base edges are present in $\delta(\mathcal{A})$ and none of the root edges are. The rate at which the costs of the request and base edges are contributed to by the dual variables remain constant as long as these edges are in $\delta(\mathcal{A})$. Thus for any request edge $e \in E_R$, either e will be added to F by time $c_e/2$ or else the endpoints of e will become part of the same cluster at this time or before. Similarly any base edge $e \in E_B$ will be added to F by time c_e or else the corresponding base and request will have become part of the same cluster at or before this time. The root edges are only contributed to once the corresponding base becomes part of an active cluster. For each base $i \in B$, let $s(i)$ be the cost of the minimum-cost edge connecting this base to a request node. Hence $s(i) := \min\{d(i, j) : j \in R\}$.

Lemma 2. *For each base $i \in B$, if there is a request node $j \in R$ such that the base edge $\{i, j\} \in F$, then $d(i, j) = s(i)$ and no other base edges in F are adjacent to i .*

Proof. Fix a base $i \in B$. Let $j \in R$ be a request for which $d(i, j) = s(i)$ and let $k \in R$ be any other request. By the definition of $s(i)$ we know $d(i, k) \geq d(i, j)$, and since d is a metric we have $d(j, k) \leq d(i, j) + d(i, k) \leq 2d(i, k)$. Since we choose request edges in preference to base edges, this means that before i connects to any request, that request will be contained in the same cluster as request j . Hence i can only connect to at most one request. Furthermore, if $\{i, k\}$ is still eligible to be added to F , then $\{i, j\}$ must be as well, so the only way $\{i, k\}$ could be added is if $d(i, k) \leq d(i, j) = s(i)$. Therefore if any base edge adjacent to i is added to F , it must have cost $s(i)$, and at most one such edge can be added. \square

The above lemma implies that either a base $i \in B$ will not get paid for, or else it will get paid for by time $s(i) + f_i$. This means that the primal-dual schema is equivalent to performing Kruskal's algorithm on the same graph, but with modified edge weights c'_e , where

$$c'_e = \begin{cases} c_e/2, & \text{if } e \in E_R \\ c_e, & \text{if } e \in E_B \\ s(i) + f_i, & \text{if } e = \{r, i\} \in E_r \end{cases}$$

The only difference is that when running Kruskal's algorithm all of the nodes will end up connected, so we may have unpaid for bases that have a root edge but no base edge. If we adjust our definition of a paid for base to be only those bases with degree 2, then running Kruskal's algorithm is equivalent.

Lemma 3. *The set of edges F' comprises a feasible solution to (LR-IP).*

Proof. The only edges that have been removed from F are the base edges corresponding to bases that are not paid for. This means there is no root edge connecting to these bases, and by Lemma 2 we know there is only one base edge adjacent to these bases. Therefore any unpaid for base with an adjacent edge in F must be a leaf in the final tree formed by F . Removing these edges simply disconnects these bases without affecting the connectivity of any of the request nodes to the root. \square

Lemma 4. *For the final solution F' returned by the primal-dual algorithm, and any \mathcal{A} corresponding to a positive dual variable, the following condition holds: $|\delta(\mathcal{A}) \cap F'| = |\mathcal{A}|$.*

Proof. Let \mathcal{I} be the set of singleton bases that are paid for but that are not present in any of the clusters in \mathcal{A} , and let C_r denote the cluster containing the root at the time when the dual variable associated with \mathcal{A} was increased. Construct a graph H where the nodes correspond to the clusters in \mathcal{I} and \mathcal{A} as

well as C_r , and the edges correspond to those in $\delta(\mathcal{A} \cup \mathcal{I} \cup \{C_r\}) \cap F'$. We will first argue that the graph H is a tree.

We know that all the edges in H correspond to edges that are also in F . Furthermore all the edges that make each cluster in H a connected component are also in F . The final edge set F is acyclic, and since $F' \subseteq F$ then we know that H is acyclic as well. Additionally, F' is a feasible solution, so every request node has a path to the root, and every base that is paid for also connects to the root. This implies that H is connected. Since H is connected and acyclic, it must be a tree.

Since there are $|\mathcal{I}| + |\mathcal{A}| + 1$ nodes, there must be $|\mathcal{I}| + |\mathcal{A}|$ edges in the tree H . However, this is including edges that go between two inactive clusters, which are precisely the edges not counted in $\delta(\mathcal{A}) \cap F'$. The only inactive clusters are C_r and those in \mathcal{I} , and the only edges between these clusters are the root edges to each paid for base in \mathcal{I} . Therefore

$$|\delta(\mathcal{A}) \cap F'| = |\delta(\mathcal{A} \cup \mathcal{I} \cup \{C_r\}) \cap F'| - |\mathcal{I}| = |\mathcal{I}| + |\mathcal{A}| - |\mathcal{I}| = |\mathcal{A}|. \quad \square$$

Theorem 1. *The final solution, F' , returned by the algorithm satisfies*

$$\sum_{e \in F' \cap E_R} c_e + 2 \sum_{e \in F' \setminus E_R} c_e \leq 2 \sum_{\mathcal{A} \in \mathcal{D}} |\mathcal{A}| y_{\mathcal{A}},$$

which implies it is a Lagrangian-preserving 2-approximation algorithm.

Proof. An edge is only added to F if its corresponding dual constraint becomes tight, so we have that

$$\sum_{e \in F' \cap E_R} c_e + 2 \sum_{e \in F' \setminus E_R} c_e = 2 \sum_{\substack{e \in F' \setminus E_R: \\ \delta(\mathcal{A}) \ni e}} \sum_{\mathcal{A} \in \mathcal{D}} y_{\mathcal{A}} = 2 \sum_{\mathcal{A} \in \mathcal{D}} y_{\mathcal{A}} |\delta(\mathcal{A}) \cap F'| = 2 \sum_{\mathcal{A} \in \mathcal{D}} |\mathcal{A}| y_{\mathcal{A}},$$

where the equalities are derived by reversing the order of summation and then applying Lemma 4. □

Routing with Trees, k -Bases The k -location-routing problem has the same constraints as the location-routing problem, but now instead of paying an opening cost for each base in the solution, we are simply limited to opening at most k bases, for some specified k . In this section we will continue considering the case where each request is only required to have a path to an open base (and hence the root). The changes needed in (LR-IP) to model this case correspond to setting the cost of each root edge to zero, and instead imposing a constraint that at most k root edges are used. Because the primal-dual algorithm shown is Lagrangian-preserving, we can use it to approximate the k -location-routing problem.

For any input to the k -location-routing problem, we can set the cost of each base to zero and apply the primal-dual algorithm. If we get lucky and end up with k or fewer bases in the final solution, then by the same analysis as before we

can see that the solution is a 2-approximation for the k -location-routing problem as well. For the rest of this section we will assume that this case does not occur.

We now wish to determine a value λ such that if we make all bases have the same cost λ , then the primal-dual algorithm will open exactly k bases. If such a value can be found, then this too would imply a 2-approximate solution to the k -location-routing problem. This can be seen by letting x_e^* be the optimal solution to the k -location-routing problem with trees, then we have by Theorem [□](#)

$$\sum_{e \in F' \cap E_R} c_e + 2 \sum_{e \in F' \cap E_B} c_e + 2k\lambda \leq 2 \sum_{e \in E_R \cup E_B} c_e x_e^* + 2\lambda \sum_{e \in E_r} x_e \leq 2 \sum_{e \in E_R \cup E_B} c_e x_e^* + 2k\lambda,$$

and subtracting the $2k\lambda$ term from both sides we see that cost of the tours in the primal-dual solution is within a factor of 2 of the cost of the tours in the optimal solution.

To determine the appropriate value of λ , we would first like to determine for each request edge $e = \{u, w\} \in E_R$, which values of λ will result in edge e being part of the final solution. We know that if edge e gets added to F , it does so at time $c_e/2$, and otherwise u and w must have already become part of the same cluster by this time. Nodes u and w will become part of the same cluster only if there is a path between u and w where each edge has c'_e value at most $c_e/2$. For some edges, satisfying this condition will be dependent on the value of λ chosen, and for others it will not. We can determine the sensitivity of each edge on the value of λ by performing the following procedure.

We will run the primal-dual algorithm by setting each base cost to be infinite. Naturally no root edge will become tight in this setting, so we will stop the algorithm once there are only two components left in \mathcal{C} corresponding to r and $B \cup R$. For each active cluster $S \in \mathcal{A}$ we will associate a value $v(S)$ corresponding to the lowest $s(i)$ value for any base i contained in the cluster. If the cluster contains no bases then $v(S)$ will simply be equal to the current time, and hence will continue to grow until a base node is added to the cluster. When a request edge $e = \{u, w\} \in E_R$ is added to the solution, and S_u and S_w correspond to the clusters containing u and w , we set $s(e) = \max\{v(S_u), v(S_w)\}$. Once the algorithm stops we set λ_k to be the largest value for which there are at most $|R| - k$ candidate edges with $c_e/2 - s(e) < \lambda_k$. We will show that λ_k is the value that the base cost should be set to in order to force our algorithm to open exactly k bases. More precisely

$$\lambda_k := \max\{\lambda : |\{e \in E_R : c_e/2 - s(e) < \lambda\}| \leq |R| - k\}.$$

Lemma 5. *Request edge $e \in E_R$ is added to the solution if $c_e/2 - s(e) < \lambda$, and if $c_e/2 = s(e)$ then this edge is added to the solution regardless of the value of λ . If $c_e/2 - s(e) = \lambda > 0$ then e can be placed in the solution or not by choice of tie-breaking.*

Proof. If $\lambda > c_e/2 - s(e)$, then the only bases that will be paid for by time $c_e/2$ are those for which $s(i) < s(e)$. By the way in which $s(e)$ was set, we know that there is at least one endpoint of e that no such base can reach by time $c_e/2$.

Thus there is an active cluster S with $e \in \delta(S)$ up until time $c_e/2$, and so e will be added to the solution.

If $c_e/2 = s(e)$, then since request edges are processed before base edges, we know that for at least one of the endpoints of e the associated cluster contained no bases. This means that no base can reach this endpoint by time $c_e/2$. So again there is an active cluster S with $e \in \delta(S)$ up until time $c_e/2$, and so e will be added to the solution.

If $c_e/2 - s(e) = \lambda > 0$, then for one of the endpoints of e the associated cluster contains a base that gets tight at time $c_e/2$. The other endpoint is either already inactive or else is in the same situation. We did not specify a preference between processing request edges and root edges. So either we add edge e first in which case it is part of the solution, or else we add the root edge (possibly root edges) first, in which case both endpoints become inactive and e is not part of the solution. \square

Since no request edge is ever deleted from the solution, then each connected component in the subgraph induced by R must have precisely one paid for base in the solution connecting it to the root. If $\lambda_k = 0$, then this means we will add more than $|R| - k$ request edges, and thus open fewer than k bases, even when there are no base-opening costs. This solution is a 2-approximation the k -location-routing problem, since we open fewer than k bases and no portion of the cost is attributed to opening the bases. Otherwise, if we set $\lambda = \lambda_k > 0$ then by tie-breaking appropriately we can ensure that exactly $|R| - k$ request edges are added to the solution. Hence the number of bases opened is precisely $|R| - (|R| - k) = k$ as desired, and as shown above this implies a 2-approximation as well.

Getting Tours From Trees with No Approximation Loss If we wanted each request to be connected to a base through a cycle as opposed to tree, we could take the solution produced by the primal-dual algorithm and duplicate each edge, and then using shortcutting produce a disjoint set of cycles, each containing a base while not more than doubling the cost. Note that it is unnecessary to duplicate the root edges, since these edges simply represent the cost of each base, and are not required to be part of a cycle.

We can actually do better and not lose a factor of 2 in the approximation guarantee. Let us start with the same underlying graph, but make the cost of all root edges equal to *half* of the cost of the corresponding base. We say a solution to the location-routing problem with cycles consists of a set of edges such that

- In the induced subgraph on R , the selected edges form a disjoint set of paths.
- The selected edges connect each such path's endpoints to some base, forming a cycle. We consider a singleton request node a path of zero-length, and there must be two copies of an edge leading from such a node to a base.
- There must be two copies selected of each root edge leading to a base that was used to connect at least one path.

The cost of this solution is the cost of the edges selected, which corresponds to the cost of each base used (since the root edge is half the cost, but we took two copies) and the cost of all edges in the cycles.

Lemma 6. *Any solution to the location-routing problem with cycles satisfies the inequality*

$$2 \sum_{e \in \delta(\mathcal{A}) \cap E_R} x_e + \sum_{e \in \delta(\mathcal{A}) \setminus E_R} x_e \geq 2|\mathcal{A}|, \forall \mathcal{A} \in \mathcal{D}.$$

Proof. The proof will be quite similar to the proof of Lemma [1](#), but in this case we now require the coefficient of 2 for each request edge. Consider a feasible solution to the location-routing problem with cycles, x , and let F be the corresponding set of edges. Note that now x_e will equal 2 for some edges e . Fix any $\mathcal{A} \in \mathcal{D}$. Again we will define two subsets of F , but slightly differently than before. Let $F_1 := (\delta(\mathcal{A}) \cap F) \setminus E_R$; $F_2 := (\delta(\mathcal{A}) \cap F) \cap E_R$.

Now initialize $\mathcal{A}' = \mathcal{A}$. For each edge $e \in F_2$, let us merge the subsets in \mathcal{A}' corresponding to the endpoints of e in an iterative process. When we have finished we will have that $|\mathcal{A}| - |\mathcal{A}'| = |F_2|$. After merging the subsets, we will have that the only edges left in $\delta(\mathcal{A}') \cap F$ are those in F_1 , though this time there may be edges leading between components of \mathcal{A}' . If we consider the subgraph consisting of the nodes R and the edges $F \cap E_R$ then we know that it consists of a disjoint set of paths. Each cluster in \mathcal{A}' must contain at least one of these disjoint paths in its entirety, since there are no longer any edges in E_R leading out of any clusters.

For each cluster in \mathcal{A}' , pick one of the disjoint request paths it contains. This path must have its endpoints connect to some base. If the base is not in the cluster, then this mean there must be two edges in F_1 associated with this cluster (or two copies of one edge in the case of a singleton request path). If the base is in the cluster, then we root edge must be in F_1 , and the solution must select two copies of it. In either case, there are at least two edges (or two copies of one edge) in F_1 associated with each cluster. Furthermore we associate each copy of an edge in F_1 with the cluster containing the request if it is a base node, and with the cluster containing the base if it is a root node. This means the association is a bijection and hence

$$2 \sum_{e \in \delta(\mathcal{A}) \cap E_R} x_e + \sum_{e \in \delta(\mathcal{A}) \setminus E_R} x_e = 2|F_2| + \sum_{e \in F_1} x_e = 2|F_2| + 2|\mathcal{A}'| \geq 2|\mathcal{A}|. \quad \square$$

We now have a LP relaxation to the location-routing problem with cycles that is identical to [\(LR-P\)](#), except that the right-hand side of each constraint is now 2 instead of 1. This means the corresponding dual is identical to [\(LR-D\)](#), except that the objective function coefficient of each dual variable is now 2 instead of 1 as well. Thus the final dual solution that the algorithm ends up with is feasible for this new dual program as well, and the corresponding solution has twice the objective function cost. In other words, any feasible solution to [\(LR-D\)](#) has value at most half of the optimal solution with cycles, but only when the costs of the root edges have been set to half the base-opening costs.

Set $c'_e = c_e/2$ for each $e \in E_r$ and $c'_e = c_e$ for each $e \in E_B \cup E_R$. Now if we run the algorithm with the costs c'_e , duplicate each edge in the solution F' and then shortcut as necessary, we produce a solution to the location-routing problem with cycles, which we will denote as x . Then we have

$$\sum_{e \in E \setminus E_r} c'_e x_e + 2 \sum_{e \in E_r} c'_e x_e \leq 2 \sum_{e \in F' \setminus E_r} c'_e + 4 \sum_{e \in F' \cap E_r} c'_e.$$

From Theorem [1](#) we know that this cost is at most 4 times the dual cost, which as argued above is at most 2 times the optimal cost. Thus this procedure is a Lagrangian-preserving 2-approximation algorithm for the location-routing problem with cycles, and so can also be used to produce a 2-approximation for the k -location-routing problem with cycles.

Acknowledgment. We would like to thank an anonymous reviewer who pointed out the simple approach to achieving a constant-factor approximation algorithm for the k -location-routing problem.

References

1. Berger, R.T., Coullard, C.R., Daskin, M.S.: Location-routing problems with distance constraints. *Transportation Science* 41(1), 29–43 (2007)
2. Goemans, M.X., Williamson, D.P.: A general approximation technique for constrained forest problems. *SIAM J. Comput.* 24(2), 296–317 (1995)
3. Jain, K., Vazirani, V.V.: Approximation algorithms for metric facility location and k -median problems using the primal-dual schema and Lagrangian relaxation. *J. ACM* 48(2), 274–296 (2001)
4. Laporte, G.: Generalized subtour elimination constraints and connectivity constraints. *J. Oper. Res. Soc.* 37(5), 509–514 (1986)
5. Laporte, G.: Location-routing problems. In: Golden, B.L., Assad, A.A. (eds.) *Vehicle Routing: Methods and Studies*, pp. 163–198. North Holland, Amsterdam (1986)
6. Laporte, G., Nobert, Y.: An exact algorithm for minimizing routing and operating costs in depot location. *Eur. J. Oper. Res.* 6(2), 224–226 (1981)
7. Laporte, G., Nobert, Y., Arpin, D.: An exact algorithm for solving a capacitated location-routing problem. *Ann. Oper. Res.* 6(2), 293–310 (1986)
8. Laporte, G., Nobert, Y., Pelletier, P.: Hamiltonian location problems. *Eur. J. Oper. Res.* 12(1), 82–89 (1983)
9. Laporte, G., Nobert, Y., Taillefer, S.: Solving a family of multi-depot vehicle routing and location-routing problems. *Transportation Sci.* 22(3), 161–172 (1988)
10. Min, H., Jayaraman, V., Srivastava, R.: Combined location-routing problems: A synthesis and future research directions. *Eur. J. Oper. Res.* 108(1), 1–15 (1998)

Scheduling Resources for Throughput Maximization

Venkatesan T. Chakaravarthy¹, Amit Kumar², Vinayaka Pandit¹,
Sambuddha Roy¹, and Yogish Sabharwal¹

¹ IBM Research - India, New Delhi

² Indian Institute of Technology, New Delhi

{vechakra,sambuddha,ysabharwal}@in.ibm.com, amitk@cse.iitd.ac.in

Abstract. We consider the problem of scheduling a set of resources over time. Each resource is specified by a set of time intervals (and the associated amount of resource available), and we can choose to schedule it in one of these intervals. The goal is to maximize the number of demands satisfied, where each demand is an interval with a starting and ending time, and a certain resource requirement. This problem arises naturally in many scenarios, e.g., the resource could be an energy source, and we would like to suitably combine different energy sources to satisfy as many demands as possible. We give a constant factor randomized approximation algorithm for this problem, under suitable assumptions (the so called no-bottleneck assumptions). We show that without these assumptions, the problem is as hard as the independent set problem. Our proof requires a novel configuration LP relaxation for this problem. The LP relaxation exploits the pattern of demand sharing that can occur across different resources.

1 Introduction

We consider the problem of scheduling jobs when the resources required for executing the jobs have limited availability. We use the terms “resource” and “machine” interchangeably. In the scheduling literature it is typical to assume that the machines are always available for scheduling and the goal is to schedule jobs so as to satisfy all the constraints of correct scheduling and optimize some objective function like makespan, flowtime, completion time, etc. This is in stark contrast to our scenario, where the machines are not available at all the times. Each machine specifies a time window within which it is available, along with a duration for which the machine can be used for. We generalize such a setting by allowing each machine to specify a list of intervals when it is available and the scheduler can pick only one interval for each machine. Therefore, the main challenge is to judiciously schedule the machines in one of their allowed intervals so as to maximize the number of jobs that can be executed in the chosen intervals.

The setting of limited machine (or resource) availability arises naturally in several scenarios. As an example, consider a workforce management scenario, where the employees specify different intervals of the day when they are available and the scheduler can only pick one of their intervals (an employee can

have only a single shift in a day). The goal of the scheduler is to pick a shift for each employee such that a maximum number of jobs can be processed (jobs are specified by intervals). The problem framework is quite general and captures many other situations arising in sensor networks, cloud computing, energy management, distributed computing and micro-grid scheduling (see, eg. [12]).

Problem Definition: We define the problem of *resource scheduling for throughput maximization* (RSTM) as follows. We assume that time is divided into discrete timeslots $\{1, 2, \dots, L\}$. We are given a set of *demands* (alternatively, *jobs*) \mathcal{D} . Each demand $j \in \mathcal{D}$ has a starting timeslot $s(j)$, ending timeslot $e(j)$, bandwidth requirement $\rho(j)$ and profit $p(j)$. There are m *resources* (alternatively, *machines*) $\mathcal{P}_1, \dots, \mathcal{P}_m$. Each resource \mathcal{P}_i is described by a set of *resource-intervals*. Each resource-interval $I \in \mathcal{P}_i$ has a starting timeslot $s(I)$, ending timeslot $e(I)$, and offers a bandwidth $h(I)$. Let $\mathcal{P} = \cup_i \mathcal{P}_i$, (the set of all the resource-intervals – note that \mathcal{P} could be a multiset).

A feasible solution \mathcal{S} selects a subset of resource-intervals $R \subseteq \mathcal{P}$ and a subset of demands $J \subseteq \mathcal{D}$ satisfying the following constraints: (i) at most one resource-interval is selected for each resource \mathcal{P}_i ; (ii) at any timeslot t , the sum of bandwidth requirements of jobs from J active at t is no more than the sum of the bandwidths offered by the resource-intervals of R active at t . The goal is to maximize the throughput; namely, sum of profits of jobs in J . We assume that starting times, ending times, bandwidth requirements of demands, bandwidths of resource-intervals are integers.

The RSTM problem is a generalization of the well-studied unsplittable flow problem (UFP) on line. In the UFP on line graphs, we are given a set of jobs; each job is specified by an interval, a bandwidth requirement and profit. For each timeslot, the input specifies the bandwidth available at the timeslot. The goal is to select a maximum profit subset of jobs such that the bandwidth constraints are not violated. The UFP is captured by the special case of the RSTM problem, wherein each resource consists of only a single resource-interval.

In addition to addressing the issue of selecting jobs as in the UFP, the RSTM problem also poses the challenge of scheduling the resources. However, the two tasks are interdependent, since the choice of resource scheduling critically determines the set of jobs that can be selected and hence, the profit of the solution. Thus, a key aspect of any algorithm for the RSTM problem lies in handling the two tasks simultaneously.

Our Results: We can show that the RSTM problem generalizes the maximum independent set problem, which is NP-hard to approximate within a factor of $n^{1-\epsilon}$, for any $\epsilon > 0$ [15].

We study a restricted version of the RSTM problem, wherein the input instance satisfies the following two constraints: (i) the length of the longest job is at most the length of the shortest resource-interval (i.e., $\max_{j \in \mathcal{D}} \ell(j) \leq \min_{I \in \mathcal{P}} \ell(I)$, where $\ell(j) = e(j) - s(j) + 1$ and $\ell(I) = e(I) - s(I) + 1$); (ii) the bandwidth requirement of any job is at most the minimum bandwidth offered by the resource-intervals (i.e., $\max_{j \in \mathcal{D}} \rho(j) \leq \min_{I \in \mathcal{P}} h(I)$). Borrowing terminology from the UFP

literature, we refer to the pair of constraints as the *no-bottleneck assumption* (NBA). Note that the second condition is analogous to the following condition required by known constant factor-approximation algorithms for UFP on the line – the maximum bandwidth requirement of any demand is at most the smallest “edge capacity”.

Our main result is a constant factor randomized approximation algorithm for the RSTM problem, under the NBA. We can show that dropping either one of the NBA constraints leads to the RSTM problem becoming as hard as the maximum independent set problem.

We now briefly discuss the main ideas behind our algorithm. A more detailed overview appears in Section 2. It can be shown that a natural LP relaxation for the problem has an unbounded integrality gap. Our constant factor approximation algorithm has two components: (i) we first show that any optimum solution can be transformed into a different solution having certain desirable structural properties (a so called “star solution”), with only a constant factor loss in profit; (ii) we next show that the optimum star solution can be approximated within constant factors in polynomial time – this is accomplished by applying randomized rounding to a suitable configuration LP.

Related Work: As discussed earlier, the RSTM problem is related to the UFP on a line. Calinescu et al. [5] and independently, Bar-Noy et al. [3] considered the uniform case of the UFP, where the bandwidth available is uniform across all time slots and gave a 3-approximation algorithm. The generalization where the bandwidth available can vary has received considerable attention. Constant factor approximation algorithms were known, under the NBA [8,9]. Bansal et al. [2] presented a quasi-PTAS for the same problem, without assuming NBA; in a recent breakthrough, Bonsma et al. [4] designed a $(7 + \epsilon)$ -approximation algorithm for this problem.

Prior work has addressed an extension of the UFP on line with “bag constraints” on the demands. The bag constraint for a demand essentially specifies a set of intervals in which the demand can potentially be scheduled and the scheduler is allowed to pick at most one of these intervals. Bar-Noy et al. [3] presented a 5-approximation for this problem, when the bandwidth available is uniform across all timeslots. For the general case with arbitrary bandwidth availability, an $O(\log(\frac{h_{max}}{h_{min}}))$ -approximation algorithm is known [7]; a constant factor approximation algorithm is known under the NBA assumption [6]. In our problem, the bag constraints are applied on the resources.

The model of limited machine availability, although rare, has been studied in the operations research literature [14,1]. But most of these works do not consider the restriction of having to pick only one of the many intervals in which a machine is potentially available. Motivated by applications in energy savings in data center operations, Khuller et al. [13] consider the following model. We are given a set of machines, where each machine has an activation cost and a limited budget is available for activating the machines. The goal is to activate the machines within the given budget and minimize the makespan of scheduling

the jobs. They give a bi-criteria approximation algorithm for this problem. But, note that once a machine is activated, it is available at all the times.

2 Outline of Our Algorithm

We begin with a simplifying assumption that $\rho(j) = 1$ for all jobs j and $h(I)$ is an integer for all resource-intervals I . We will relax the assumption in Section 5.

Fix a feasible solution \mathcal{S} . We may also assume that for each job j (picked by \mathcal{S}) and each timeslot t where j is active, the job is processed by a single resource-interval; we assume that the solution also specifies the resource-interval which processes j at each timeslot. We say that a resource-interval I *processes* a job j , if there is some timeslot t where I processes j .

As a warmup to our main result, let us first consider a simpler problem. The new problem, called single resource-interval RSTM, is the same as the RSTM problem, except that any job selected by the solution must be processed fully by a single resource-interval chosen by the solution. In contrast, a solution for the RSTM problem only needs to satisfy the bandwidth constraints, which means that a job may be jointly processed by multiple resource-intervals. It is easy to approximate the single resource-interval RSTM problem within a constant factor. The proof is omitted.

Theorem 1. *The single resource-interval RSTM problem can be approximated in polynomial time within a factor of 2.*

We now turn our attention to the RSTM problem. As discussed earlier, the main issue is that a job may be processed by multiple resource-intervals. However, we will show that the optimum solution opt can be transformed into another solution with only a constant factor loss in profit such that any job in the transformed solution is processed by at most *two* resource-intervals; call such a solution a *2-solution*. Similarly, a solution is said to be a *1-solution* if each job is processed by a single resource-interval. For a 2-solution \mathcal{X} , let $J^{(1)}(\mathcal{X})$ denote the jobs processed by a single resource-interval and similarly, let $J^{(2)}(\mathcal{X})$ denote those processed by two resource-intervals. The set $J^{(2)}(\mathcal{X})$ can be represented as a graph wherein the resource-intervals selected by \mathcal{X} form the vertices and an edge is drawn between two resource-intervals, if they jointly process some job (all such jobs can be labeled on the edge). We call this the *sharing graph* of \mathcal{X} and denote it $G(\mathcal{X})$. We will show that with only a constant factor loss in profit, the solution \mathcal{X} can be transformed further into two solutions \mathcal{S}_1^* and \mathcal{S}_2^* , where \mathcal{S}_1^* is a 1-solution and \mathcal{S}_2^* is a 2-solution whose sharing graph is a union of disjoint stars (a star consists of a *head* – or center – and a set of leaves connected to it). Furthermore, we show that in the solution \mathcal{S}_2^* , each resource-interval I will process at most $h(I)$ jobs from $J^{(2)}(\mathcal{S}_2^*)$. We call such a solution \mathcal{S}_2^* as a *height-bounded star solution*. Theorem 2 summarizes the above transformations. Corollary 1 allows us to effectively approximate \mathcal{S}_1^* .

Theorem 2. *There exist a 1-solution \mathcal{S}_1^* and a height-bounded star solution \mathcal{S}_2^* such that $\text{profit}(\mathcal{S}_1^*) + \text{profit}(J^{(2)}(\mathcal{S}_2^*)) \geq \text{profit}(\text{opt})/c$, for a constant c .*

Corollary 1. *There exists a polynomial time algorithm that outputs a solution \mathcal{S}' such that $\text{profit}(\mathcal{S}') \geq \text{profit}(\mathcal{S})/2$, for any 1-solution \mathcal{S} .*

Later, we shall design an algorithm for approximating the profit of \mathcal{S}_2^* within constant factors and prove the following theorem (see Section 4).

Theorem 3. *There exists a polynomial time algorithm that outputs a solution \mathcal{S}' such that $\text{profit}(\mathcal{S}') \geq \text{profit}(J^{(2)}(\mathcal{S}))/c$, for any height-bounded star solution \mathcal{S} , where c is some constant.*

The overall algorithm will simply output the best of the two solutions output by Corollary 1 and Theorem 3. It can be seen that the output solution \mathcal{S}_{out} has profit at least a constant factor of the profit of both \mathcal{S}_1^* and $J^{(2)}(\mathcal{S}_2^*)$. This implies that $\text{profit}(\mathcal{S}_{out})$ is at least a constant factor of the profit of opt .

We now briefly outline the ideas behind the proof of Theorem 3. We write a *configuration LP* for finding the optimal height-bounded star solution. The LP essentially has exponentially many variables, one for each star. The separation oracle for this LP relies on finding the best *density* star solution – such a problem turns out to be a special case of submodular maximization subject to a matroid and two knapsack constraints [10]. Finally, we show that a simple randomized rounding algorithm with greedy updates gives a good integral solution.

3 Height-Bounded Star Solutions: Proof of Theorem 2

We will first show that the optimum solution opt can be transformed into a new 2-solution \mathcal{S} with only a constant factor loss in profit. The proof exploits the first condition of the NBA (regarding the lengths of jobs). Furthermore, \mathcal{S} will satisfy some additional properties.

3.1 Transformation to 2-Solutions

Lemma 1. *There exists a 2-solution \mathcal{S} having profit at least $\text{profit}(\text{opt})/4$. Furthermore, the solution \mathcal{S} has the following properties:*

- For any job $j \in J^{(2)}(\mathcal{S})$ processed by two resource-intervals I_1, I_2 , where $s(I_1) \leq s(I_2)$, we have that $[s(j), e(j)]$ is not contained in either $[s(I_1), e(I_1)]$ and $[s(I_2), e(I_2)]$ (i.e., $s(I_1) < s(j) < s(I_2) \leq e(I_1) < e(j) < e(I_2)$); moreover, j is processed by I_1 till time $s(I_2)$ and by I_2 after this point of time.
- The total number of jobs from $J^{(2)}(\mathcal{S})$ processed by a resource-interval I is at most $2 \cdot h(I)$.

Proof. We start by first proving a claim. Let X be a set of jobs. For a time t , let $A_t(X)$ denote the set of all jobs from X active at time t , i.e., the jobs j for which $s(j) \leq t \leq e(j)$.

Claim. Given a set of jobs X , there exists a subset of jobs $Y \subseteq X$ such that for all time t , $|A_t(Y)| \leq 2$ and $|A_t(Y)| \geq \min\{1, |A_t(X)|\}$.

Proof. We initialize a set Y to \emptyset . First select the job $j \in X$ with the smallest value of $s(j)$. In case there are more than one such jobs, choose one that has the largest value of $e(j)$. Add the selected job j to Y . We now repeatedly add jobs as follows. Let j denote the last job that was added to Y . Let \mathcal{J} be the set of jobs j' for which $e(j') > e(j)$ and $s(j') \leq e(j)$. If \mathcal{J} is not empty, add the job with the largest $e(j')$ value in \mathcal{J} to Y . In case \mathcal{J} is empty, consider the smallest time greater than $e(j)$ which is the start time of some job – let t denote this time. Let \mathcal{J}' be the set of jobs which start at time t . Add the job in \mathcal{J}' with largest end time to Y . This process is repeated as long as possible.

We now show that the set Y constructed above has the claimed property. Clearly, at every time t , $|A_t(Y)| \geq \min\{1, |A_t(X)|\}$. Now suppose that there are three jobs j_1, j_2 and j_3 (added in this order to Y) active at some time t . Consider the stage when j_3 was added to Y . As j_1 and j_2 were already present in Y , we have that $e(j_3) > e(j_2)$. Therefore, $s(j_3) \leq e(j_1) < e(j_3)$. Now consider the stage when j_2 was added. Notice that j_3 overlaps with j_1 (and hence, also with the last job added to Y). Moreover, j_3 extends further than j_2 . Hence, j_3 should have been added instead of j_2 . This proves the claim. \square

We now complete the proof of the lemma. For each resource-interval I selected by `opt`, we replace it by $h(I)$ new resource-intervals of unit capacity each – the start and end times of these new resource-intervals are same as those of I respectively. We call these resource-intervals *slices* of I . Let \mathcal{B} denote the set of all slices obtained in this way. For a subset of slices $L \subseteq \mathcal{B}$, let $A_t(L)$ denote the set of slices active at time t . Using an argument similar to that of Claim 3.1, we can find a subset of slices $L \subseteq \mathcal{B}$ such that at each time $|A_t(L)| \leq 2$ and $|A_t(L)| \geq \min\{1, |A_t(\mathcal{B})|\}$. This is achieved using the same procedure as in the proof of Claim 3.1.

Let J be the set of all jobs selected by `opt`. We next apply Claim 3.1 twice: apply the claim on the set J to get a set of jobs Y_1 and then apply the claim again on the set $J - Y_1$ to get a set of jobs Y_2 . Let $Y = Y_1 \cup Y_2$. The set Y satisfies the property that at each time $|A_t(Y)| \leq 4$ and $|A_t(Y)| \geq \min\{2, |A_t(J)|\}$.

At any time t , at most four jobs from Y are active. This means that the set Y can be colored with four colors, i.e., Y can be partitioned into four subsets such that no two jobs from the same subset are active at the same time. Let Y' be the subset having the maximum profit among these four subsets. Then, $\text{profit}(Y') \geq \text{profit}(Y)/4$. We now assign jobs in Y' to the slices in L . Consider a job $j \in Y'$. Since this job is part of the feasible solution \mathcal{S} , $A_t(\mathcal{B}) \geq 1$ for all $t \in [s_j, e_j]$. Therefore, for all such time t , $|A_t(L)| \geq 1$. Among the slices $\ell \in L$ active at $s(j)$, pick the slice having the largest $e(\ell)$. If ℓ fully contains j , then simply assign j to ℓ . Otherwise (i.e., if $e(j) > e(\ell)$), there must be some slice $\ell' \in L$ which is active at time $e(\ell)$ and $e(\ell') > e(\ell)$. Note that $s(\ell') > s(j)$ because of our choice of ℓ . Moreover, $e(\ell') > e(j)$ because of the no bottleneck assumption. Thus, j can be assigned to ℓ and ℓ' . In this case, we say that ℓ is the *left interval* used by j and ℓ' is the *right interval* used by j . In this way, we are able to assign every job in Y' to at most two slices (and hence, the corresponding resource-intervals) in L .

Now, delete all the jobs in Y from J and all the slices in L from \mathcal{B} – let J' and \mathcal{B}' denote these new sets respectively. We note that the jobs in J' “fit” in the slices of \mathcal{B}' , i.e., at any time t , $|A_t(J')| \leq |A_t(\mathcal{B}')|$. This follows from the fact that at any time t , $|A_t(Y)| \geq \min\{|A_t(L)|, |A_t(J)|\}$. We can now apply the entire procedure iteratively with these new sets J' and \mathcal{B}' . We collect all the sets Y' generated in each of these iterations and take Z to be the union of these sets. We see that $\text{profit}(Z) \geq \text{profit}(J)/4$ and each job $j \in Z$ is assigned to at most two resources.

Each resource-interval I is used as a left interval by at most $h(I)$ jobs and as a right interval by at most $h(I)$ jobs. Note that all the properties stated in the lemma are satisfied. \square

Let \mathcal{S} be the solution guaranteed by Lemma 1. The lemma implies that the jobs serviced by the solution \mathcal{S} can only be of two types – $J^{(1)}(\mathcal{S})$, which are processed by exactly one resource-interval in \mathcal{S} ; and $J^{(2)}(\mathcal{S})$, which are serviced by two resource-intervals selected in \mathcal{S} . Let \mathcal{C} be the resource-intervals selected by \mathcal{S} . A job $j \in J^{(2)}(\mathcal{S})$ uses two resource-intervals $I_1, I_2 \in \mathcal{C}$ as stated in Lemma 1 – let I_1 be the *left interval* used by j and I_2 be the *right interval* used by j . We also use the terminology that I_1 (or I_2) services j as its left (or right) interval. We call a solution \mathcal{X} to be *nice*, if the following properties hold: (i) \mathcal{X} satisfies the conditions of Lemma 1; (ii) every resource-interval I selected by \mathcal{X} services all jobs in $J^{(2)}(\mathcal{X})$ as either their left or their right interval only; (iii) the number of jobs processed by a resource-interval I from $J^{(2)}(\mathcal{X})$ is at most $h(I)$.

Lemma 2. *Let \mathcal{S} be a solution as guaranteed by Lemma 1. Then there exists a nice solution \mathcal{S}' such that $\text{profit}(\mathcal{S}') \geq \text{profit}(J^{(1)}(\mathcal{S})) + \text{profit}(J^{(2)}(\mathcal{S}))/4$.*

Proof. We independently label each resource-interval I selected by \mathcal{S} as either \mathcal{L} or \mathcal{R} with probability half. We process a job $j \in J^{(2)}(\mathcal{S})$ if and only if its left interval gets label \mathcal{L} and its right interval gets label \mathcal{R} . Clearly, such a solution has the desired properties. The second statement in the lemma follows from the proof of Lemma 1. Each resource-interval I in \mathcal{S} acts as left interval (or right interval) for at most $h(I)$ jobs. The jobs in $J^{(1)}(\mathcal{S})$ are retained in \mathcal{S}' . Now, note that a job $j \in J^{(2)}(\mathcal{S})$ gets selected in \mathcal{S}' with probability $\frac{1}{4}$. Thus, the expected profit of \mathcal{S}' is at least $\text{profit}(J^{(1)}(\mathcal{S})) + \text{profit}(J^{(2)}(\mathcal{S}))/4$. This shows the existence of the claimed solution. \square

3.2 Transformation to Height-Bounded Star Solutions

By Lemma 2, we can assume the existence of a nice solution \mathcal{S} having profit at least a constant factor of the optimum solution. The sharing graph $G(\mathcal{S})$ is a bipartite graph with the two sides being denoted $\mathcal{L}(\mathcal{S})$ and $\mathcal{R}(\mathcal{S})$ – the resource-intervals which serve jobs as their left interval and those which serve jobs as their right interval respectively. Our next result shows that the sharing graph $G(\mathcal{S})$ can be assumed to be a star (with only a constant factor loss in profit).

Lemma 3. *Given a nice solution \mathcal{S} , there is another solution $\mathcal{S}' \subseteq J^{(2)}(\mathcal{S})$ such that \mathcal{S}' is a height-bounded star solution and the profit of \mathcal{S}' is at least half of the profit of $J^{(2)}(\mathcal{S})$.*

Proof. Let \mathcal{C} be the resource-intervals selected by \mathcal{S} . We first reduce the capacity $h(I)$ of a resource-interval in \mathcal{C} to the number of jobs in $J^{(2)}(\mathcal{S})$ that it services (Lemma 2 shows that this quantity is at most $h(I)$). Also note that this is just for the sake of studying the property of the solution \mathcal{S} . We now prove that it is possible to process the jobs in $J^{(2)}(\mathcal{S})$ in \mathcal{C} such that the sharing graph is a forest. To this end, we explicitly give a way of packing these jobs in the resource-intervals in \mathcal{S} .

We arrange the jobs j_1, \dots, j_n in ascending order of $s(j)$, i.e., $s(j_1) \leq s(j_2) \leq \dots \leq s(j_n)$. We arrange the resource-intervals in $\mathcal{L}(\mathcal{S})$ in ascending order of $e(I)$ values, i.e., $e(I_1^L) \leq e(I_2^L) \leq \dots \leq e(I_l^L)$, where $l = |\mathcal{L}(\mathcal{S})|$ and $\mathcal{L}(\mathcal{S}) = \{I_1^L, \dots, I_l^L\}$. Similarly, we arrange the resource-intervals in $\mathcal{R}(\mathcal{S}) = \{I_1^R, \dots, I_r^R\}$, $r = |\mathcal{R}(\mathcal{S})|$ in ascending order of $s(I)$ values. For an integer c , $1 \leq c \leq n$, let $L(c)$ be the unique integer i satisfying $h(I_1^L) + \dots + h(I_{i-1}^L) < c \leq h(I_i^L) + \dots + h(I_{i-1}^L) + h(I_i^L)$. Define $R(c)$ similarly for the resource-intervals in $\mathcal{R}(\mathcal{S})$.

Claim. For $1 \leq c \leq n$, the job j_c can be processed by the pair of resource-intervals $(I_{L(c)}^L, I_{R(c)}^R)$.

Proof. Let a denote $L(c)$ and b denote $R(c)$. We need to prove that j_c is contained in the span of I_a^L and I_b^R ; this means we have to prove: (i) $s(j_c) \geq s(I_a^L)$; (ii) $e(j_c) \leq e(I_b^R)$; (iii) $s(I_b^R) \leq e(I_a^L) + 1$ (i.e., there should not exist a timeslot – a gap – between the ending timeslot of I_a^L and the starting timeslot of I_b^R). We prove these below:

- Suppose, for the sake of contradiction, $s(j_c) < s(I_a^L)$. Then for any $c' \leq c$, $s(j_{c'}) < s(I_a^L)$. The no bottleneck assumption implies that $e(j_{c'}) < e(I_a^L)$ and so, $e(j_{c'}) < e(I_{a'}^L)$ for all $a' \geq a$. But then \mathcal{S} must have scheduled $j_{c'}$ using one of the resource-intervals in $\{I_1^L, \dots, I_{a-1}^L\}$. But this is not possible because there are c jobs in $\{j_1, \dots, j_c\}$ and the total capacity of $\{I_1^L, \dots, I_{a-1}^L\}$ is less than c . This shows that $s(j_c) \geq s(I_a^L)$.
- Again, suppose $e(j_c) > e(I_b^R)$. Then, the no bottleneck assumption implies that $s(j_c) > s(I_b^R)$ and so, $s(j_{c'}) > s(I_{b'}^R)$, if $c \leq c'$ and $b' \leq b$. So the jobs in $\{j_c, \dots, j_n\}$ can only be scheduled among I_{b+1}^R, \dots, I_r^R . Now, the fact $h(I_1^R) + h(I_2^R) + \dots + h(I_r^R) = n$ and the definition of b lead to a contradiction.
- Since $s(j_c) < e(j_c)$, the above two statements imply that I_a^L and I_b^R can intersect if and only if $s(I_b^R) \leq e(I_a^L) + 1$. If $e(I_a^L) < s(I_b^R)$, then $e(I_a^L) < s(I_{b'}^R)$, $a' \leq a, b' \geq b$. So, the resource-intervals in $\{I_1^L, \dots, I_a^L\}$ can intersect with resource-intervals in $\{I_1^R, \dots, I_{b-1}^R\}$ only. But the former set of resource-intervals are processing at least c jobs and the latter set can process less than c jobs – a contradiction. \square

Consider the schedule implied by the claim above. Note that the sharing graph is bipartite. Also, it is easy to check that in the sharing graph, one of the vertices

I_1^L and I_1^R is a leaf. Removing this leaf and continuing the argument gives a sequence in which we can remove leaves from the sharing graph (till all vertices are removed). So it must be a forest. Now consider a tree in the forest. We can decompose the edges into two sets of disjoint union of stars (just consider alternate levels). Then, we pick the set with the higher profit. The jobs in $J^{(1)}(\mathcal{S})$ are retained in \mathcal{S}' . \square

Proof of Theorem 2: Let \mathcal{S} be the nice solution output by Lemma 2. We take $\mathcal{S}_1^* = J^{(1)}(\mathcal{S})$ and take the solution \mathcal{S}' output by Lemma 3 as \mathcal{S}_2^* .

4 Finding Height-Bounded Star Solutions

Here, we prove Theorem 3. For a star T , let $\text{hd}(T)$ be the root vertex in T , and $\text{Leaves}(T)$ be the set of leaves in T . We say that a star T can *satisfy* a set of jobs L if it is possible to partition L into $|\text{Leaves}(T)|$ sets $- L_1, \dots, L_r$, $r = |\text{Leaves}(T)|$, such that the following conditions holds :

- $|L| \leq h(\text{hd}(T))$.
- Let T_i be the i^{th} leaf of T . Then for $i = 1, \dots, |\text{Leaves}(T)|$, $|L_i| \leq h(T_i)$ and each job in L_i can be served jointly by $\text{hd}(T)$ and T_i .

We say that a collection $\{(T^{(1)}, L^{(1)}), (T^{(2)}, L^{(2)}), \dots, (T^{(t)}, L^{(t)})\}$, is a *satisfiable star solution*, if $T^{(i)}$ can satisfy $L^{(i)}$, for all i . We now show that the optimum satisfiable star solution can be approximated within constant factors.

Theorem 4. *There exists a randomized polynomial time algorithm that outputs a satisfiable star solution whose profit is at least a constant factor of the optimum satisfiable star solution.*

It is clear any satisfiable star solution is a feasible height-bounded star solution. Furthermore, for any height-bounded star solution \mathcal{S} , $J^{(2)}(\mathcal{S})$ is a satisfiable star solution. Therefore, the above result implies Theorem 3. The rest of the section is devoted to proving Theorem 4. The algorithm is based on LP rounding.

4.1 LP Relaxation

We now write an LP relaxation for finding the optimal satisfiable star solution. For each pair (T, L) , where T is a star and L is a set of jobs which can be satisfied by T , we have a variable $y(T, L)$. For a set of jobs L , define $p(L)$ as the total profit of jobs in L . The LP relaxation is as follows :

$$\max \sum_{(T,L)} y(T, L) \cdot p(L)$$

$$\sum_{(T,L):j \in L} y(T, L) \leq 1 \quad \text{for all } j \in \mathcal{D} \quad (1)$$

$$\sum_{(T,L):T \cap \mathcal{P}_i \neq \emptyset} y(T, L) \leq 1 \quad \text{for all } \mathcal{P}_i \quad (2)$$

$$y(T, L) \geq 0 \quad \text{for all } (T, L)$$

Now, there are two issues – solving the LP (since the number of variables is exponential) and rounding a fractional solution. We approximately solve the LP by providing a approximate separation oracle for the dual. The oracle goes by framing the issue as maximizing a submodular function subject to a matroid constraint and two knapsack constraints. Recent results [11] provide approximation algorithms for the latter task. The details are omitted. We now proceed to show how to round the fractional solution.

4.2 Rounding a Fractional Solution

Let $y^*(T, L)$ be an optimal (or near-optimal) solution to the LP. We show how it can be rounded to an integral solution. We can assume that $y^*(T, L)$ will assume non-zero values for a polynomial number of pairs (T, L) only – let these be $(T^{(r)}, L^{(r)})$, $r = 1, \dots, n$. Let the leaves of $T^{(r)}$ be labeled $I_1^{(r)}, \dots, I_k^{(r)}$, where $k = |\text{Leaves}(T^{(r)})|$. Since $T^{(r)}$ can satisfy $L^{(r)}$, we divide these jobs into $|\text{Leaves}(T^{(r)})|$ sets – $L_u^{(r)}$, $u = 1, \dots, L_k^{(r)}$, such that $L_u^{(r)}$ can be processed by $\text{hd}(T^{(r)})$ and the u^{th} leaf of this star. Consider the algorithm given in Figure 4. We now argue that the expected profit of this solution is at least a constant times that of the fractional solution y^* . For a job j , define Δ_j as $\sum_{(T,L):j \in L} y^*(T, L)$. It is easy to check that $\sum_{(T,L)} y^*(T, L) \cdot p(L) = \sum_j \Delta_j \cdot p(j)$. Thus it is enough to prove that the probability we service j is $\Omega(\Delta_j)$. We fix a job j . Assume without loss of generality that j appears in $L^{(1)}, \dots, L^{(t)}$. Further, assume that j appears in the subset $I_1^{(l)}$ of $L^{(l)}$, $1 \leq l \leq t$. For sake of brevity, let z_l denote $\frac{y^*(T^{(l)}, L^{(l)})}{12}$, and z denote $\sum_l z_l$. Let X_l , $1 \leq l \leq t$, denote the 0-1 random variable which is 1 precisely when we select both $\text{hd}(L^{(l)})$ and the leaf interval $I_1^{(l)}$. Thus, we service j iff at least one of the random variables X_l is 1.

Claim. For $1 \leq l \leq t$, we have $z_l/2 \leq \mathbf{E}[X_l] \leq z_l$.

Proof. First note that $X_l = 1$ only if we choose to consider the pair $(T^{(l)}, L^{(l)})$. So $E[X_l] \leq z_l$. Now suppose we consider this pair (which happens with probability z_l). Constraint (2) implies that the probability that we do not select $\text{hd}(L^{(l)})$ is at most $\frac{1}{12}$, and the same statement holds for $I_1^{(l)}$. So, the probability that we do not select one of these two resource-intervals is at most $\frac{1}{6}$. Therefore, X_l is equal to 1 with probability at least $\frac{5}{6} \cdot z_l$. \square

Now we would like to bound the probability that all X_l are 0. Unfortunately, the random variables X_l are not independent. Define $X = \sum_l X_l$. The claim above shows that $E[X] \geq \frac{z}{2}$.

Claim. For all integers $p \geq 1$, $\mathbf{Pr}[X = p] \leq z^p$.

Proof. Fix any p random variables X_{i_1}, \dots, X_{i_p} from the set $\{X_1, X_2, \dots, X_t\}$. The probability that all of these are 1 is at most $\prod_{i=1}^p z_{i_i}$ (since the coin tosses for deciding whether we consider a pair (T, l) are independent). Now taking the sum over all such tuples, we see that $\mathbf{Pr}[X = p]$ is at most $(z_1 + \dots + z_t)^p$. \square

For $r = 1, \dots, n$ do

With probability $\frac{y^*(T^{(r)}, L^{(r)})}{12}$, consider the pair $(T^{(r)}, L^{(r)})$ in the following steps.

1. If no resource-interval from the resource containing $\text{hd}(T^{(r)})$ has been chosen far
 - (i) Select the resource-interval $\text{hd}(T^{(r)})$.
 - (ii) For $i = 1, \dots, |\text{Leaves}(T^{(r)})|$ do
 - If no resource-interval from the resource containing $I_i^{(r)}$ has been taken yet, then
 - select the resource-interval $I_i^{(r)}$ and service the jobs in $L_i^{(r)}$ using this resource-interval and $\text{hd}(T^{(r)})$.

Fig. 1. Algorithm Round

We are now almost done.

$$\begin{aligned} \frac{z}{2} \leq \mathbf{E}[X] &= \sum_{p=1}^{\infty} p \cdot \mathbf{Pr}[X = p] \leq \mathbf{Pr}[X = 1] + \sum_{p \geq 2} p \cdot z^p \\ &\leq \mathbf{Pr}[X = 1] + 3z^2 \leq \mathbf{Pr}[X = 1] + \frac{z}{4} \end{aligned}$$

where the last inequality is true since $z = \frac{\Delta_j}{12} \leq \frac{1}{12}$ (using constraint [\(II\)](#)). Thus we get $\mathbf{Pr}[X = 1] \geq \frac{z}{4} = \Omega(\Delta_j)$, which is what we wanted to prove. So, the expected profit of algorithm [Round](#) is at least a constant times that of y^* . This completes the proof of [Theorem 4](#).

5 Overall Algorithm

We finally put everything together to get the following main result.

Theorem 5. *There exists a constant factor randomized approximation algorithm for the RSTM problem (with NBA).*

Proof. The overall algorithm (called [Schedule](#)) outputs the best of the two solutions output by [Corollary 1](#) and [Theorem 3](#). Let the solution output by the our algorithm be \mathcal{S} and let opt be the optimum solution. By [Theorem 2](#), opt can be transformed into a height-bounded star solution \mathcal{S}^* , with only a constant factor loss in profit. It can be seen that the output solution \mathcal{S} has profit at least a constant factor of the profit of both $J^{(1)}(\mathcal{S}^*)$ and $J^{(2)}(\mathcal{S}^*)$. This implies that $\text{profit}(\mathcal{S})$ is at least a constant factor of the profit of \mathcal{S}^* . Hence, \mathcal{S} is a constant factor approximation to opt .

Now consider the general case when jobs can have arbitrary (but integral) $\rho(j)$ values. We will assume that $h(I)$ is an integer for all resource-intervals I . We divide a job j into $\rho(j)$ new jobs – each such job j' has $\rho(j') = 1$ and profit equal to $\frac{p(j)}{\rho(j)}$. We now run the algorithm [Schedule](#) on this instance. Now, the

problem is that a job j may get picked to a *partial* extent, i.e., the algorithm may pick $r(j)$ copies of the jobs, where $0 \leq r(j) \leq \rho(j)$, and the corresponding profit is $\frac{r(j)}{\rho(j)} \cdot p(j)$. Now, consider the resource-intervals picked by this algorithm – this gives a certain capacity to each timeslot (which is equal to the capacities of resource-intervals which get picked and contain this timeslot). So the solution can be thought of as a *fractional* solution to unsplittable-flow problem (UFP) on the line where the capacities of timeslots are as described, and a job j is picked to an extent of $\frac{s(j)}{\rho(j)}$. But we know that the standard LP relaxation for UFP has constant integrality gap [9]; so, we can find an integral solution (i.e., either we pick the entire job or none of it) of profit at least a constant times that of the solution output by algorithm **Schedule**. \square

References

1. Aggoune, R.: Minimizing the makespan for the flow shop scheduling problem with availability constraints. *European J. of Oper. Res.* 125, 534–543 (2004)
2. Bansal, N., Chakrabarti, A., Epstein, A., Schieber, B.: A quasi-ptas for unsplittable flow on line graphs. In: *STOC* (2006)
3. Bar-Noy, A., Bar-Yehuda, R., Freund, A., Noar, J., Schieber, B.: A unified approach to approximating resource allocation and scheduling. *Journal of the ACM* 48(5), 1069–1090 (2001)
4. Bonsma, P., Schulz, J., Wiese, A.: A constant factor approximation algorithm for unsplittable flow on paths. *CoRR*, abs/1102.3643 (2011)
5. Calinescu, G., Chakrabarti, A., Karloff, H., Rabani, Y.: Improved approximation algorithms for resource allocation. In: Cook, W.J., Schulz, A.S. (eds.) *IPCO 2002*. LNCS, vol. 2337, pp. 401–414. Springer, Heidelberg (2002)
6. Chakaravarthy, V., Choudhury, A., Sabharwal, Y.: A near-linear time constant factor algorithm for unsplittable flow problem on line with bag constraints. In: *FSTTCS* (2010)
7. Chakaravarthy, V., Pandit, V., Sabharwal, Y., Seetharam, D.: Varying bandwidth resource allocation problem with bag constraints. In: *IPDPS* (2010)
8. Chakrabarti, A., Chekuri, C., Gupta, A., Kumar, A.: Approximation algorithms for the unsplittable flow problem. *Algorithmica* 47(1), 53–78 (2007)
9. Chekuri, C., Mydlarz, M., Shepherd, F.: Multicommodity demand flow in a tree and packing integer programs. *ACM Transactions on Algorithms* 3(3) (2007)
10. Chekuri, C., Vondrák, J., Zenklusen, R.: Dependent randomized rounding via exchange properties of combinatorial structures. In: *FOCS* (2010)
11. Gupta, A., Roth, A., Schoenebeck, G., Talwar, K.: Constrained non-monotone submodular maximization: Offline and secretary algorithms. In: Saberi, A. (ed.) *WINE 2010*. LNCS, vol. 6484, pp. 246–257. Springer, Heidelberg (2010)
12. Hatzigargyriou, N., Asano, H., Irvani, R., Marnay, C.: Microgrids. *IEEE Power and Energy Magazine* 5(4), 78–94 (2007)
13. Khuller, S., Li, J., Saha, B.: Energy efficient scheduling via partial shutdown. In: *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, SODA* (2010)
14. Schmidt, G.: Scheduling with limited machine availability. *European J. of Oper. Res.* 121, 1–15 (2000)
15. Zuckerman, D.: Linear degree extractors and the inapproximability of max clique and chromatic number. In: *STOC* (2006)

Coloring and Maximum Independent Set of Rectangles

Parinya Chalermsook

Department of Computer Science, University of Chicago, Chicago, IL, USA

Abstract. In this paper, we consider two geometric optimization problems: RECTANGLE COLORING problem (RCOL) and MAXIMUM INDEPENDENT SET OF RECTANGLES (MISR). In RCOL, we are given a collection of n rectangles in the plane where overlapping rectangles need to be colored differently, and the goal is to find a coloring using minimum number of colors. Let q be the maximum clique size of the instance, i.e. the maximum number of rectangles containing the same point. We are interested in bounding the ratio $\sigma(q)$ between the total number of colors used and the clique size. This problem was first raised by graph theory community in 1960 when the ratio of $\sigma(q) \leq O(q)$ was proved. Over decades, except for special cases, only the constant in front of q has been improved. In this paper, we present a new bound for $\sigma(q)$ that significantly improves the known bounds for a broad class of instances.

The bound $\sigma(q)$ has a strong connection with the integrality gap of natural LP relaxation for MISR, in which the input is a collection of rectangles where each rectangle is additionally associated with non-negative weight, and our objective is to find a maximum-weight independent set of rectangles. MISR has been studied extensively and has applications in various areas of computer science. Our new bounds for RCOL imply new approximation algorithms for a broad class of MISR, including (i) $O(\log \log n)$ approximation algorithm for unweighted MISR, matching the result by Chalermsook and Chuzhoy, and (ii) $O(\log \log n)$ -approximation algorithm for the MISR instances arising in the UNSPLITTABLE FLOW PROBLEM on paths. Our technique builds on and generalizes past works.

1 Introduction

In this paper, we devise algorithms for two geometric optimization problems: RECTANGLE COLORING problem (RCOL) and MAXIMUM INDEPENDENT SET OF RECTANGLES (MISR). In RCOL, we are given a collection \mathcal{R} of n rectangles. Our objective is to find a valid coloring of rectangles such that no two overlapping rectangles get the same color, while minimizing the number of colors. Clearly, this problem is a special case of GRAPH COLORING problem: Define graph $G = (V, E)$ where the vertex set corresponds to rectangles, and there is an edge connecting two vertices that correspond to overlapping rectangles. We denote by $\omega(\mathcal{R})$ the size of maximum clique of collection \mathcal{R} and $\chi(\mathcal{R})$ its chromatic number. When \mathcal{R} is clear from context, we will often use q to denote $\omega(\mathcal{R})$. Note that $\chi(\mathcal{R}) \geq \omega(\mathcal{R})$, so an interesting question to ask is how large the ratio $\chi(\mathcal{R})/\omega(\mathcal{R})$ can be.

We denote such ratio by $\sigma(\mathcal{R})$ and define $\sigma_{\text{rect},q} = \sup_{\mathcal{R}:\omega(\mathcal{R})=q} \sigma(\mathcal{R})$. We are interested in bounding $\sigma_{\text{rect},q}$ as a function that depends on q but not on the input size. Notice that obtaining such bounds in general graphs is impossible as Erdős observed that there are family of graphs with maximum clique size 2 and arbitrarily large chromatic number [14]. However, for several interesting family of geometric intersection graphs, such as rectangles, segments and circular arc graphs, this ratio is well defined and has been studied; please refer to the survey by Kostochka [21] for more detail. For rectangle intersection graphs, not much progress has been made. In 1960, Asplund and Grünbaum show that any collection \mathcal{R} of rectangles with clique size q can be colored by at most $O(q^2)$ colors, implying the ratio $\sigma_{\text{rect},q} \leq O(q)$, where they also prove the lower bound of 3. This bound remained asymptotically best known. In this paper, we show a new bound of $\sigma(\mathcal{R}) \leq O(\gamma \log q)$ where γ is a parameter we will define later. Since our γ is at most q , in the worst case, we still have the bound of $\tilde{O}(q)$, and we get an improvement when $\gamma = o(\frac{q}{\log q})$. For a broad class of instances, γ is expected to be constant.

It turns out that this bound is enough for us to get an improved approximation factor for a large class of MISR instances. In the MISR problem, the input is the set \mathcal{R} of rectangles in the plane where each rectangle $R \in \mathcal{R}$ is associated with weight w_R , and the goal is to find a maximum weight subset of non-overlapping rectangles. Being one of the most fundamental problem in computational geometry, MISR comes up in various areas of computer science, e.g. in data mining [20,17,22], map labeling [11,13], channel admission control [23], and pricing [12]. MISR is NP-hard [16,19], and there has been a long line of attack on the problem, proposing approximation algorithms for both general cases [20,1,24,5,23] and special cases [15,10,8]. Currently the best known approximation ratio is $O(\log n / \log \log n)$ by Chan and Har-peled [11]. Through the connection between RCOL and MISR, our bounds for $\sigma(\mathcal{R})$ immediately give $O(\gamma \log \log n)$ approximation algorithms for MISR, where $\gamma \leq O(\log n)$.

Here we discuss some consequences of our results. For unweighted setting of MISR, the value of γ is one, so our result would give $O(\log \log n)$ approximation algorithm, matching the bound of [8]. For general (weighted) MISR, if γ is constant, our algorithm would give $O(\log \log n)$ approximation factor. An evidence that our result could be useful is when Bonsma, Schulz, and Wiese [7] recently showed a constant factor approximation algorithm for UNSPLITTABLE FLOW problem, and their main ingredient is an algorithm that solves a very restricted instance of MISR. Despite being very special cases, no approximation algorithms for MISR existing in the literature could directly apply to give $O(\log^{1-\epsilon} n)$ bound for it. However, it is easy to show that the instances they solve have $\gamma \leq 2$ (details are in the full version), so we obtain an $O(\log \log n)$ approximation algorithm for solving this instance; note, however, that the algorithm of [7] solves this instance exactly in polynomial time.

Our contributions: The main technical contribution of this paper is a new bound for $\sigma(\mathcal{R})$ improving upon the bound in [18] when $\gamma = o(\frac{q}{\log q})$. We now discuss this quantity precisely. For each rectangle $R \in \mathcal{R}$, we define the *containment*

depth $d(R)$ as the number of rectangles $R' \neq R$ that completely contains R , and $d(\mathcal{R}) = \max_{R \in \mathcal{R}} d(R)$. Notice that $d(\mathcal{R}) \leq q$. The set $H(R)$ is defined as $H(R) = \{R' : R' \subseteq R\}$. Then we let $h(R)$ be the size of the maximum independent set $\mathcal{S} \subseteq H(R)$ such that all rectangles in \mathcal{S} can be pierced by one horizontal line (in other words, the projection of rectangles in \mathcal{S} onto vertical line share some common point). Then define $h(\mathcal{R}) = \max_{R \in \mathcal{R}} h(R)$.

Theorem 1. *For any collection \mathcal{R} of rectangles with $\gamma = \min \{d(\mathcal{R}), h(\mathcal{R})\} + 1$, there is a polynomial time algorithm that finds $O(\gamma q \log q)$ -coloring of \mathcal{R} . In particular, $\sigma(\mathcal{R}) \leq O(\gamma \log q)$, which is $o(q)$ when $\gamma = o(\frac{q}{\log q})$.*

For purpose of solving the unweighted MISR problem, we may assume without loss of generality that there is no containment of any two rectangles in \mathcal{R} (so $d(\mathcal{R}) = 0$): Assume there are two rectangles R, R' such that R' contains R . We simply remove rectangle R' from the collection without affecting the optimal solution. Therefore, it is interesting and natural to study the ratio $\sigma_{\text{rect-nc,q}}$ defined as the maximum $\sigma(\mathcal{R})$ where \mathcal{R} is a collection of rectangles with $d(\mathcal{R}) = 0$.

Corollary 1. *For collection \mathcal{R} in which for any two rectangles R and R' , R does not contain R' , there is an $O(q \log q)$ -coloring algorithm that runs in polynomial time. In particular, $\sigma_{\text{rect-nc,q}} \leq O(\log q)$.*

Through the connection between MISR and RCOL (presented in Section 3), we get the following approximation bound for MISR.

Theorem 2. *For any collection \mathcal{R} of rectangles, let $\gamma = \min \{d(\mathcal{R}), h(\mathcal{R})\} + 1$. There is an $O(\gamma \log \log n)$ approximation algorithm for MISR.*

We observe that our work relies a lot on dealing with the ‘‘corner information’’ of the intersecting rectangles. In fact, many recent works that solve optimization problems on rectangle intersection graphs have exploited this information in one way or another [4,11,8,23]. In Section 5, we investigate special cases of RCOL and MISR by restricting the intersection types and show additional results.

Related work: The study of the ratio $\sigma_{\text{rect,q}}$ for rectangle intersection graphs started in 1948, when Bielecki [6] asked whether the value of $\sigma_{\text{rect,q}}$ is independent of the instance size n . This question was answered positively by Asplund and Grünbaum in 1960 [3], when they show that $\chi(\mathcal{R}) \leq 4q^2 - 3q$, which implies that $\sigma_{\text{rect,q}} \leq 4q - 3$. The bound was later improved to $\sigma_{\text{rect,q}} \leq 3q - 2$ by Hendler [18], while the best lower bound remains $\sigma_{\text{rect,q}} \geq 3$ by constructing a set of rectangles with clique size 2 and chromatic number 6 [3]; in fact, their result implies the exact bound $\sigma_{\text{rect,2}} = 6$. Better bounds are known for special cases. For squares, a better bound of $\sigma_{\text{squares,q}} \leq 4$ was shown by Ahlswede and Karapetyan, and independently by Perepelitsa (see [2]). Lewin-Eytan et al. show that $\sigma_{\text{rect-non-corner,q}} = 1$ where the collections of interest do not have any rectangle that contains corners of other rectangles [23]. All these upper bounds imply polynomial time algorithms for finding the coloring. We refer to the survey by Kostochka for more related work [21].

For more related works on MISR, we refer the readers to [8,11] and references therein.

Organization: In Section 2, we recall the standard terms in graph theory, stated in the context of rectangles, and define the notations. We discuss the connection between MISR and RCOL in Section 3. Then we show the coloring algorithms in section 4.

2 Preliminaries

A rectangle R is given by a quadruple $(x^l(R), x^r(R), y^t(R), y^b(R))$ of real numbers, corresponding to the x -coordinates of its left and right boundaries and the y -coordinates of its top and bottom boundaries respectively. Furthermore, we assume that each rectangle is closed, i.e. each $R \in \mathcal{R}$ is defined as follows: $R = \{(x, y) : x^l(R) \leq x \leq x^r(R) \text{ and } y^b(R) \leq y \leq y^t(R)\}$. We say that rectangles R and R' intersect iff $R \cap R' \neq \emptyset$. In both RCOL and MISR, we are given a collection \mathcal{R} of n -axis parallel rectangles. For MISR, each rectangle R is associated with weight w_R . The goal of the RCOL is to find a minimum coloring, while the goal of MISR is to find maximum-weight independent set.

We will distinguish among the three types of intersections: corner, crossing, and containment (see Figure 1) whose formal definitions are as follows. For two overlapping rectangles R, R' , we let $j(R, R')$ denote the number of corners of R contained in R' , and let $c(R, R') = \max\{j(R, R'), j(R', R)\}$. We say that the intersection between R and R' is a *corner* intersection iff $c(R, R') \in \{1, 2\}$. It is called a *crossing* iff $c(R, R') = 0$. Otherwise $c(R, R') = 4$, and we say that two rectangles have *containment* intersection.

2.1 Polynomially Bounded Weights

We argue that we can assume, by losing a constant factor in the approximation ratio, that all weights w_R are positive integers of values at most $2n$. We first scale the weights of rectangles so that the minimum weight is at least 1. Let W_{\max} be the weight of the maximum weight rectangle. For each rectangle $R \in \mathcal{R}$, we assign a new weight

$$w'_R = \left\lfloor w_R \cdot \frac{2n}{W_{\max}} \right\rfloor$$

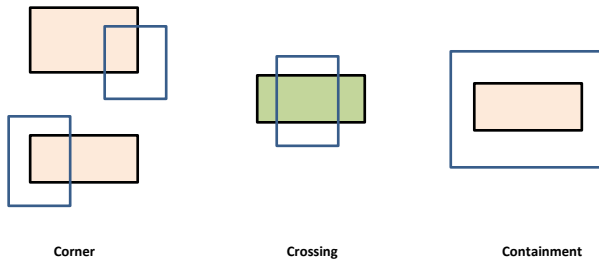


Fig. 1. Three possible intersection types

In the new instance, the weight of maximum weight rectangle becomes $2n$. It is easy to see that any γ -approximate solution to the new instance gives a 2γ -approximate solution to the original instance.

2.2 Rectangle Coloring and Degenerate Instances

Here we recall some standard terms in graph theory and state them in our context of rectangles, which is more convenient for our purpose. First, a set of rectangles $\mathcal{Q} \subseteq \mathcal{R}$ forms a *clique* if the intersection of all the rectangles in \mathcal{Q} is non-empty. Let \mathcal{R}' be a collection of rectangles. We say that \mathcal{R}' admits a c -coloring if there exists an assignment $b : \mathcal{R}' \rightarrow [c]$ such that no two overlapping rectangles in \mathcal{R}' get the same number. A collection \mathcal{R}' of rectangles is k -*degenerate* if for every sub-collection $\mathcal{R}'' \subseteq \mathcal{R}'$, there exists a rectangle $R \in \mathcal{R}''$ such that R intersects with at most k other rectangles in \mathcal{R}'' . It is a standard fact that any k -degenerate collection \mathcal{R}' is $(k+1)$ -colorable, and such coloring can be computed efficiently: Choose a rectangle $R \in \mathcal{R}'$ such that the size of neighbors of R , i.e. $|\{S \in \mathcal{R}' : S \cap R \neq \emptyset\}|$, is at most k . Recursively color the collection $\mathcal{R}' \setminus \{R\}$. Assign any color to R that does not conflict with any of R 's neighbors.

2.3 Sparse Instances

We say that a collection of rectangles \mathcal{R} is s -*sparse* if, for any rectangle $R \in \mathcal{R}$, there exist s points $p_1^R, p_2^R, \dots, p_s^R \in \mathbb{R}^2$ associated with rectangle R (to be called *representative points* of R) such that the following holds. For any overlapping rectangles $R, R' \in \mathcal{R}$, either $p_i^R \in R'$ for some i or $p_j^{R'} \in R$ for some j . We note that Chan [9] uses similar ideas to define β -fat objects. For example, any collection of rectangles with only corner and containment intersections is 4-sparse: for each rectangle R , we can define $\{p_1^R, p_2^R, p_3^R, p_4^R\}$ to be the set of the four corners of R , so whenever two rectangles overlap, one rectangle contains some representative point of another.

Now we generalize the lemma in [23], again restated in our terms. The proof follows along the same lines and is omitted from this extended abstract.

Lemma 1. *Let \mathcal{R}' be an s -sparse instance with maximum clique size q . Then \mathcal{R}' is $(2sq)$ -degenerate, and therefore is $(2sq + 1)$ -colorable.*

3 Independent Set and Coloring

In this section, we discuss the connections between RCOL and MISR. We remark that any c -coloring algorithm for \mathcal{R} trivially implies an algorithm that finds an independent set of size $|\mathcal{R}|/c$. However, this bound is too loose for our purpose. The following theorem, whose analogous unweighted version was used implicitly in the prior work of the author with Chuzhoy [8], summarizes the connection between the two problems:

Theorem 3. *The integrality gap of natural LP relaxation of MISR is at most $\sigma_{\text{rect}, O(\log n)}$. Moreover, if there is a polynomial time algorithm that finds a valid coloring of rectangles of \mathcal{R} using at most $qf(q)$ colors, then we have a $c_1 f(\log n)$ randomized approximation algorithm for MISR where c_1 is a constant that does not depend on the input instance.*

In particular, if the right answer for $\sigma_{\text{rect}, q}$ is constant independent of q , the integrality gap for MISR would be constant as well. Or, if we can bound $\sigma(\mathcal{R})$ for a particular instance \mathcal{R} , we would get $\sigma(\mathcal{R})$ approximate solution for MISR problem on \mathcal{R} . For unweighted MISR, Theorem 3 translates to the following:

Corollary 2. *The integrality gap of natural LP relaxation of unweighted MISR is at most $\sigma_{\text{rect-nc}, O(\log n)}$. Moreover, if there is a polynomial time algorithm that finds a valid coloring of rectangles of \mathcal{R} using at most $qf(q)$ colors, then we have a $c_2 f(\log n)$ randomized approximation algorithm for unweighted MISR where c_2 is a constant that does not depend on q .*

Now we prove the theorem.

Proof. (Of Lemma 3) We first consider a natural LP relaxation of the problem. We have, for each rectangle R , an indicator variable x_R of whether R is included in the intended independent set. Let $X = \{x^l(R), x^r(R) : R \in \mathcal{R}\}$ and $Y = \{y^t(R), y^b(R) : R \in \mathcal{R}\}$ be the set of all x and y coordinates of the boundaries of input rectangles respectively. We define \mathcal{P} to be the set of all “interesting points” of the plane: $\mathcal{P} = \{(x, y) : x \in X, y \in Y\}$. Notice that $|\mathcal{P}| \leq (2n)^2$. The LP relaxation is as follows.

$$\begin{aligned}
 \text{(LP)} \quad & \max \sum_{R \in \mathcal{R}} w_R x_R \\
 \text{s.t.} \quad & \sum_{R: p \in R} x_R \leq 1 \text{ for all } p \in \mathcal{P}
 \end{aligned}$$

To avoid confusion, we will be using the term LP-value to refer to the value of specific LP variable x_R . We say LP-cost of collection \mathcal{R}' to mean the quantity $\sum_{R \in \mathcal{R}'} w_R x_R$.

Let z be an optimal LP solution with associated LP-cost OPT. Observe that if $\text{OPT} \leq O(n)$, getting a constant approximation is trivial: simply output the maximum-weight rectangle, whose weight is always $2n$. Therefore, we assume that $\text{OPT} \geq 32n$. Let $M = 64 \log n$. The next lemma states that we can convert z into solution z' that is $(\frac{1}{M})$ -integral having roughly the same LP-value with high probability. The proof only uses standard randomized rounding techniques and is deferred to the full version

Lemma 2. *There is an efficient randomized algorithm that, given an optimal LP-solution of value $\text{OPT} \geq 32n$ for \mathcal{R} , produces with high probability, a feasible solution z' for (LP) that is $(\frac{1}{M})$ -integral whose LP-value is $\Omega(\text{OPT})$.*

Given an LP solution z' , we create a multi-subset \mathcal{R}' of \mathcal{R} as follows: for each $R \in \mathcal{R}$, add $c_R = Mz'_R$ copies of R to \mathcal{R}' . Notice that we can associate each copy in \mathcal{R}' with an LP-weight of $1/M$, so the maximum clique size is at most M . Moreover the total LP-cost in \mathcal{R}' is $\sum_{R \in \mathcal{R}'} w_R x_R = \Omega(\text{OPT})$. Now assume that we have $qf(q)$ -coloring algorithm for any collection of rectangles with clique size q . By invoking this algorithm on \mathcal{R}' , we divide rectangles in \mathcal{R}' into sets $\mathcal{R}'_1, \dots, \mathcal{R}'_{Mf(M)}$ according to their colors. Let \mathcal{R}'_j be the color class having maximum total LP-cost among the sets $\{\mathcal{R}'_{j'}\}$. We have that $\sum_{R \in \mathcal{R}'_j} \frac{w_R}{M} \geq \frac{\text{OPT}}{Mf(M)}$.

Therefore, the total weight of rectangles in \mathcal{R}'_j is $\sum_{R \in \mathcal{R}'_j} w_R \geq \Omega(\text{OPT}/f(64 \log n))$, as desired. If we are satisfied with non-constructive bound, we can invoke $(\sigma_{\text{rect}, M})$ -coloring of \mathcal{R}' , and we would get the integrality gap bound of $O(\sigma_{\text{rect}, \log n})$.

4 Coloring Algorithms

In this extended abstract, we prove a weaker result which gives $O(q^{3/2})$ -coloring for a special case. This case captures most of the key challenges of the problem.

4.1 $O(q^2)$ -Coloring

We first show how to color \mathcal{R} using $O(q^2)$ colors. This coloring algorithm will be used later as a subroutine of our main result. For each rectangle $R \in \mathcal{R}$, we denote by $V(R)$ the set of all rectangles $R' \in \mathcal{R}$ such that R and R' cross each other and the width of R' is smaller than the width of R . Let $v(R)$ be the size of the maximum clique formed by the rectangles in $V(R)$. Notice that since the maximum clique size of \mathcal{R} is q , we have that $0 \leq v(R) \leq q - 1$ for all rectangle $R \in \mathcal{R}$. It is easy to see that if $v(R) = v(R')$ for a pair of rectangles R and R' , then it is impossible for R to cross R' : Assume for contradiction that R crosses R' , and the width of R is smaller than the width of R' . Let $\mathcal{Q} \subseteq V(R)$ be a clique such that $|\mathcal{Q}| = v(R)$. Then $\{R\} \cup \mathcal{Q} \subseteq V(R')$ is a clique, and therefore $v(R') \geq v(R) + 1$.

Claim. Any collection of rectangles with clique size q is $O(q^2)$ -colorable.

Proof. We compute the values $v(R)$ for all rectangles $R \in \mathcal{R}$. Partition \mathcal{R} into q subsets S_1, \dots, S_q where $R \in S_i$ iff $v(R) = i - 1$. Since each set S_i does not have crossing, it is 4-sparse (representative points are just the four corners), and so by Lemma 1 each such collection is $O(q)$ -colorable. This implies that the set \mathcal{R} is $O(q^2)$ -colorable.

4.2 An $O(q^{3/2})$ Coloring for Restricted Setting

The coloring result in the previous section uses the values $v(R)$ to define a “grouping” of rectangles such that the intersection patterns of rectangles in the

same set S_i are limited to only crossing and containment, which is 4-sparse. Now we try to push this idea further. We would like to say such things as “if $v(R)$ and $v(R')$ are close, the intersection patterns of R and R' are limited”, and we would expect that if we group rectangles with roughly the same values of $v(R)$ together, such collection should be “almost” sparse.

We start with the following lemma about the combinatorial structures of sets of intersecting rectangles. This lemma was used implicitly in [8].

Lemma 3 (Structure Lemma). *Let \mathcal{C} be a clique, and R be any rectangle such that $\mathcal{C} \subseteq V(R)$. Then we have that*

$$v(R) \geq \min_{R' \in \mathcal{C}} \{v(R')\} + \lfloor |\mathcal{C}|/2 \rfloor$$

In other words, if we have a sub-collection of rectangles \mathcal{R}' such that $|v(R) - v(R')| \leq \delta$ for all $R, R' \in \mathcal{R}'$, then any clique $\mathcal{C} \subseteq \mathcal{R}'$ of size larger than 2δ is not a subset of $V(R)$.

Proof. Let $p = (x, y)$ be any point contained in the intersection of rectangles in $\mathcal{C} \cup \{R\}$. Consider now vertical line L passing through p . Let $\mathcal{Q} \subseteq \mathcal{C}$ be the set of $\lfloor |\mathcal{C}|/2 \rfloor$ rectangles whose left boundary is closest to L in \mathcal{C} , and let $P \in \mathcal{Q}$ be the rectangle whose right boundary is closest to L among the rectangles in \mathcal{Q} . Notice that all rectangles in $\mathcal{C} \setminus \mathcal{Q}$ intersect the left boundary of P , and all rectangles in $\mathcal{Q} \setminus \{P\}$ intersect the right boundary of P . Let \mathcal{C}' be a clique of size $v(P)$ in $V(P)$. This is the clique whose rectangles contribute to the value $v(P)$. Observe that each rectangle in \mathcal{C}' belongs to $V(R)$, and that \mathcal{C} is disjoint with \mathcal{C}' since rectangles in \mathcal{C} intersect the left or the right boundary of P while rectangles in \mathcal{C}' do not. Let $p' = (x', y)$ be any point in the intersection of rectangles in $\mathcal{C}' \cup \{P\}$ (the intersection region is shown as a black stripe in Figure 2) that is horizontally aligned with point p . Assume first that $x' > x$. Then every rectangle in \mathcal{Q} contains p' because each rectangle in \mathcal{Q} contains p (so its left boundary must lie on the left side of p) and intersects the right boundary of P (so its right boundary must be on the right of p'). Therefore $\mathcal{C}' \cup \mathcal{Q} \subseteq V(R)$ form a clique of size at least $v(P) + \lfloor |\mathcal{C}|/2 \rfloor$. Similarly, if $x' \leq x$, then every rectangle in $\mathcal{C} \setminus \mathcal{Q}$ contains p' , and we have that the set $\mathcal{C}' \cup (\mathcal{C} \setminus \mathcal{Q}) \cup \{P\}$ forms a clique of size at least $v(P) + \lfloor |\mathcal{C}|/2 \rfloor$.

We show how to use the above lemma to get a better coloring result. We introduce the key definition, similar to the one used in [8].

Definition 1. *Let \mathcal{R}' be a sub-collection of rectangles. Consider rectangle R , and let $X_1, X_2 \subseteq \mathcal{R}'$ be collections of rectangles such that $|X_1| = |X_2| = \alpha$. We say that they form an α -covering of R with respect to \mathcal{R}' iff:*

- Each rectangle in X_1 (resp. X_2) intersects the top (resp. bottom) boundary of R .
- $X_1 \cup X_2 \cup \{R\}$ forms a clique.

We denote by $\alpha_{\mathcal{R}'}(R)$ the maximum integer α such that there exist $X_1, X_2 \subseteq \mathcal{R}'$ that form an α -covering of R . When the choice of \mathcal{R}' is clear from context, we write $\alpha(R)$ instead of $\alpha_{\mathcal{R}'}(R)$.

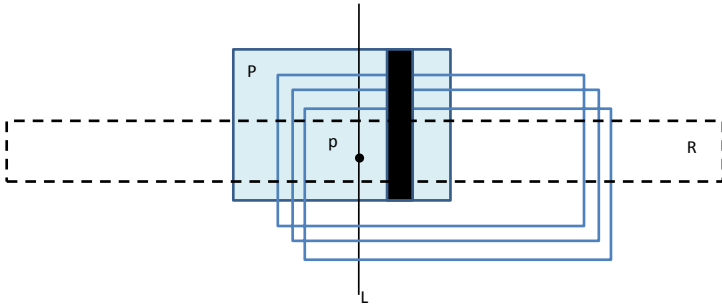


Fig. 2. Proof of Lemma 2 when $x' > x$. The black stripe shows the intersection region of $V(P)$

We will often call the set X_1 and X_2 the top and bottom α -coverage of R respectively. It is easy to see that $\alpha_{\mathcal{R}'}(R)$ can be computed in polynomial time: Fix rectangle R . For each interesting point $p \in R$, we compute the set of rectangles containing p and intersecting the top boundary of R . Denote this set by X_1^p . Set X_2^p is defined and computed similarly. Then we have that $\alpha_{\mathcal{R}'}(R) = \max_{p \in R} \min \{|X_1^p|, |X_2^p|\}$.

Claim. Consider a collection of rectangles \mathcal{R}' , and assume that \mathcal{R}' contains a clique of size q' . Then there is at least one rectangle $R \in \mathcal{R}'$ such that $\alpha_{\mathcal{R}'}(R) \geq q'/2 - 1$.

Proof. Let $\mathcal{C} \subseteq \mathcal{R}'$ be the clique of size q' , and let $X_1 \subseteq \mathcal{C}$ denote the set of $q'/2 - 1$ rectangles with highest top boundaries (breaking ties arbitrarily). Then define $X_2 \subseteq \mathcal{C}$ as the set of $q'/2 - 1$ rectangles with lowest bottom boundaries in $\mathcal{C} \setminus X_1$ (breaking ties arbitrarily). Consider any rectangle R in $\mathcal{C} \setminus (X_1 \cup X_2)$. It is easy to see that every rectangle in X_1 (resp. X_2) intersects the upper (resp. lower) boundary of R . Therefore, X_1, X_2 is a $(q'/2 - 1)$ -covering of R .

Corollary 3. For any collection \mathcal{R}' of rectangles, let $\mathcal{R}'' \subseteq \mathcal{R}'$ be a set of rectangles with $\alpha_{\mathcal{R}'}(R) \geq \nu$ for some $\nu > 2$. Then $\omega(\mathcal{R}' \setminus \mathcal{R}'') \leq 3\nu$.

Proof. Let $\tilde{\mathcal{R}} = \mathcal{R}' \setminus \mathcal{R}''$ be the set of remaining rectangles. Suppose a large clique of size 3ν remains in $\tilde{\mathcal{R}}$. Then by Claim 4.2, we would have a rectangle $R \in \tilde{\mathcal{R}}$ with $\alpha_{\tilde{\mathcal{R}}}(R) \geq 3\nu/2 - 1 > \nu$. And so we have $\alpha_{\mathcal{R}'}(R) > \nu$, a contradiction.

We are ready to describe an $O(q^{3/2})$ -coloring algorithm. For simplicity of presentation, let us for now restrict the intersection types and assume that we do not have an intersection of rectangles R and R' such that R contains at least two corners of R' . So now there are only two restricted types of intersection: (i) crossing and (ii) corner intersection where one rectangle contains exactly one corner of another. We will show in the next section how this assumption can be removed.

1. Compute the values $v(R)$ for rectangles $R \in \mathcal{R}$ in the beginning. This value is used throughout the algorithm.
2. Partition the rectangles into \sqrt{q} sets $\{S_i\}_{i=1}^{\sqrt{q}}$ where rectangle R belongs to S_i iff $(i - 1)\sqrt{q} \leq v(R) < i\sqrt{q}$. So $|v(R) - v(R')| \leq \sqrt{q}$ for all R, R' in the same set.
3. For each $i : 1 \leq i \leq \sqrt{q}$,
 - (a) Define $T_i = \{R \in S_i : \alpha_i(R) \geq 10\sqrt{q}\}$, where $\alpha_i(R)$ denotes the α -covering of R with respect to set S_i .
 - (b) $S'_i \leftarrow S_i \setminus T_i$.
 - (c) Color T_i and S'_i using $O(q)$ colors.

Assuming that step 3(c) can be implemented, it is clear that the total number of colors used is $O(q^{3/2})$. It is therefore sufficient to show that sets T_i and S'_i are $O(q)$ -colorable. For each set S'_i , notice that the clique size in S'_i is at most $O(\sqrt{q})$ after the removal of T_i from S_i , due to Corollary 3. Using the $O(\omega(S'_i)^2)$ -coloring algorithm from the previous section, we can get $O(q)$ -coloring for each set S'_i . The following claim shows that we can also color T_i .

Claim. Each set T_i is 5-sparse. Therefore, it is $O(q)$ -colorable.

Proof. Recall that each $R \in T_i$ has $(i - 1)\sqrt{q} \leq v(R) < i\sqrt{q}$. We need to define, for each $R \in T_i$, five representative points p_1^R, \dots, p_5^R . Now we fix R . Define p_1^R, \dots, p_4^R to be the four corners of R . Let (X_1, X_2) be a $10\sqrt{q}$ -coverage of R in S_i (these rectangles may not be in T_i), and $\mathcal{C} = X_1 \cup X_2$. Define p_5^R to be any common point of rectangles in \mathcal{C} .

Now we proceed to prove that the collection T_i is sparse. Consider two intersecting rectangles R and R' in T_i . If it is a corner intersection, we would be done. Otherwise, it is a crossing, and assume that the width of R' is larger than the width of R , i.e. $R \in V(R')$. We claim that $p_5^R \in R'$: If not, assume without loss of generality that p_5^R is below the bottom boundary of R' . Consider the “top α -coverage” X_1 of R . Recall that all rectangles in X_1 contain p_5^R and intersect the top boundary of R . Therefore, the only possible layout is that R' crosses every rectangle in X_1 (because of our initial assumption), or in other words, $X_1 \subseteq V(R')$. Applying Lemma 3, we have that $v(R') \geq (i - 1)\sqrt{q} + 2\sqrt{q} = (i + 1)\sqrt{q}$, which is impossible, thus concluding the proof.

Notice that the proof of this claim would fail if we do not restrict the intersection types of rectangles. To deal with the general case, we need to deal with another notion of covering. Our $O(\gamma q \log q)$ -coloring result is obtained through an iterative application of the ideas used in this section. Please see the full version for more detail.

5 Special Cases

In this section, we discuss the special cases of MISR and RCOL categorized by the types of intersections allowed. Table 1 summarizes known results on the special

Table 1. Summary of best known upper bounds . New results are marked by (*)

	no corner	no containment	no crossing	all
$\sigma(\mathcal{R})$	1*	$O(\log q)^*$	5 [23]	$3q - 2$ [18]
MISR	1*	$O(\log \log n)$ [8]	4 [23]	$O(\log n / \log \log n)$ [11]

cases. The bound of $O(\log q)$ was implied by Theorem [1] due to the fact that, without containment intersection, we have $\gamma = 1$.

We study the case when corner intersection is not allowed and prove the following theorem. Due to lack of space, the proof appears in the full version.

Theorem 4. *Let \mathcal{R} be a collection of rectangles with clique size q , in which the intersection types are only containment and crossing. Then $\chi(\mathcal{R}) = \omega(\mathcal{R})$. This implies that $\sigma(\mathcal{R}) = 1$ and maximum independent set of \mathcal{R} can be found in polynomial time.*

Acknowledgements. The author is grateful to Julia Chuzhoy for her contribution into the early stage of this paper and for comments on the draft. We thank Paolo Codenotti for many insightful discussions, Danupon Nanongkai, Bundit Laekhanukit, and anonymous referees for their suggestions and very helpful comments. We also thank Chandra Chekuri, Alina Ene, and Nitish Korula for explaining the connection between MISR and UNSPLITTABLE FLOW problem to the author.

References

1. Agarwal, P.K., van Kreveld, M.J., Suri, S.: Label placement by maximum independent set in rectangles. *Comput. Geom.* 11(3-4), 209–218 (1998)
2. Ahlswede, R., Karapetyan, I.: Intersection graphs of rectangles and segments. In: Ahlswede, R., Bäumer, L., Cai, N., Aydinian, H., Blinovskiy, V., Deppe, C., Mashurian, H. (eds.) *General Theory of Information Transfer and Combinatorics*. LNCS, vol. 4123, pp. 1064–1065. Springer, Heidelberg (2006)
3. Asplund, E., Grunbaum, B.: On a coloring problem. *Math. Scand.* 8, 181–188 (1960)
4. Bar-Yehuda, R., Hermelin, D., Rawitz, D.: Minimum vertex cover in rectangle graphs. In: de Berg, M., Meyer, U. (eds.) *ESA 2010*. LNCS, vol. 6346, pp. 255–266. Springer, Heidelberg (2010)
5. Berman, P., DasGupta, B., Muthukrishnan, S., Ramaswami, S.: Improved approximation algorithms for rectangle tiling and packing. In: *SODA*, pp. 427–436 (2001)
6. Bielecki, A.: Problem 56. *Colloq. Math.* 1, 333 (1948)
7. Bonsma, P., Schulz, J., Wiese, A.: A constant factor approximation algorithm for unsplittable flow on paths. In: *Arxiv* (2011)
8. Chalermsook, P., Chuzhoy, J.: Maximum independent set of rectangles. In: *SODA*, pp. 892–901 (2009)
9. Chan, T.M.: Polynomial-time approximation schemes for packing and piercing fat objects. *J. Algorithms* 46(2), 178–189 (2003)

10. Chan, T.M.: A note on maximum independent sets in rectangle intersection graphs. *Inf. Process. Lett.* 89(1), 19–23 (2004)
11. Chan, T.M., Har-Peled, S.: Approximation algorithms for maximum independent set of pseudo-disks. In: *Symposium on Computational Geometry*, pp. 333–340 (2009)
12. Christodoulou, G., Elbassioni, K., Fouz, M.: Truthful mechanisms for exhibitions. In: Saberi, A. (ed.) *WINE 2010. LNCS*, vol. 6484, pp. 170–181. Springer, Heidelberg (2010)
13. Doerschler, J.S., Freeman, H.: A rule-based system for dense-map name placement. *Commun. ACM* 35(1), 68–79 (1992)
14. Erdős, P.: Graph theory and probability. *Canadian J. of Mathematics* 11, 34–38 (1959)
15. Erlebach, T., Jansen, K., Seidel, E.: Polynomial-time approximation schemes for geometric graphs. In: *SODA*, pp. 671–679 (2001)
16. Fowler, R.J., Paterson, M., Tanimoto, S.L.: Optimal packing and covering in the plane are np-complete. *Inf. Process. Lett.* 12(3), 133–137 (1981)
17. Fukuda, T., Morimoto, Y., Morishita, S., Tokuyama, T.: Data mining with optimized two-dimensional association rules. *ACM Trans. Database Syst.* 26(2), 179–213 (2001)
18. Hendler, C.: Schranken für farbungs- und cliquenüberdeckungsanzahl geometrisch repräsentierbarer graphen. Master Thesis (1998)
19. Imai, H., Asano, T.: Finding the connected components and a maximum clique of an intersection graph of rectangles in the plane. *J. Algorithms* 4(4), 310–323 (1983)
20. Khanna, S., Muthukrishnan, S., Paterson, M.: On approximating rectangle tiling and packing. In: *SODA*, pp. 384–393 (1998)
21. Kostochka, A.V.: Coloring intersection graphs of geometric figures with a given clique number. *Contemporary Mathematics* 342 (2004)
22. Lent, B., Swami, A.N., Widom, J.: Clustering association rules. In: *ICDE*, pp. 220–231 (1997)
23. Lewin-Eytan, L., Naor, J(S.), Orda, A.: Routing and admission control in networks with advance reservations. In: Jansen, K., Leonardi, S., Vazirani, V.V. (eds.) *APPROX 2002. LNCS*, vol. 2462, pp. 215–228. Springer, Heidelberg (2002)
24. Nielsen, F.: Fast stabbing of boxes in high dimensions. *Theor. Comput. Sci.* 246(1–2), 53–72 (2000)

A Primal-Dual Approximation Algorithm for Min-Sum Single-Machine Scheduling Problems*

Maurice Cheung¹ and David B. Shmoys²

¹ School of Operations Research & Information Engineering
Cornell University, Ithaca NY 14853, USA
myc26@cornell.edu

² School of Operations Research & Info. Engineering & Computer Science Dept.
Cornell University, Ithaca NY 14853, USA
shmoys@cs.cornell.edu

Abstract. We consider the following single-machine scheduling problem, which is often denoted $1||\sum f_j$: we are given n jobs to be scheduled on a single machine, where each job j has an integral processing time p_j , and there is a nondecreasing, nonnegative cost function $f_j(C_j)$ that specifies the cost of finishing j at time C_j ; the objective is to minimize $\sum_{j=1}^n f_j(C_j)$. Bansal & Pruhs recently gave the first constant approximation algorithm and we improve on their 16-approximation algorithm, by giving a primal-dual pseudo-polynomial-time algorithm that finds a solution of cost at most twice the optimal cost, and then show how this can be extended to yield, for any $\epsilon > 0$, a $(2 + \epsilon)$ -approximation algorithm for this problem. Furthermore, we generalize this result to allow the machine's speed to vary over time arbitrarily, for which no previous constant-factor approximation algorithm was known.

1 Introduction

We consider the following general scheduling problem, which is denoted as $1||\sum f_j$ in the notation of scheduling problems formulated by Graham, Lawler, Lenstra, & Rinnooy Kan [4]: we are given n jobs to schedule on a single machine, where each job $j = 1, \dots, n$ has a positive integral processing time p_j , and there is a nonnegative cost function $f_j(C_j)$ that specifies the cost of finishing j at time C_j . The only restriction on the cost function $f_j(C_j)$ is that it is a nondecreasing function of C_j ; the objective is to minimize $\sum_{j=1}^n f_j(C_j)$. In a recent paper, Bansal & Pruhs [5] gave the first constant approximation algorithm for this problem; more precisely, they present a 16-approximation algorithm, that is, a polynomial-time algorithm guaranteed to be within a factor of 16 of the optimum. We improve on this result: we give a primal-dual pseudopolynomial-time algorithm that finds a solution to the scheduling problem of cost at most twice the optimal cost, and then show how this can be extended to yield, for

* Research supported partially by NSF grants CCF-0832782, CCF-1017688 and NSERC grant PGS-358528.

any $\epsilon > 0$, a $(2 + \epsilon)$ -approximation algorithm for this problem. This problem is strongly *NP*-hard (simply by considering the case of the weighted total tardiness, where $f_j(C_j) = w_j \max_{j=1, \dots, n} \{0, C_j - d_j\}$ and d_j is a specified due date of job j , $j = 1, \dots, n$). However, no hardness results are known other than this, and so it is still conceivable (and perhaps likely) that there exists a polynomial approximation scheme for this problem (though by the classic result of Garey & Johnson [10], no fully polynomial approximation scheme exists unless $P=NP$). No such scheme is known even for the special case of weighted total tardiness.

Our results are based on the linear programming relaxation of a time-indexed integer programming formulation in which the 0-1 decision variables x_{jt} indicate whether a given job $j = 1, \dots, n$, completes at time $t = 1, \dots, T$, where $T = \sum_{j=1}^n p_j$; note that, since the cost functions are nondecreasing with time, we can assume, without loss of generality, that the machine is active only throughout the interval $[0, T]$, without any idle periods. For convenience, we will set $f_j(t) = f_j(p_j)$ for each $t = 1, \dots, p_j$; this is clearly without loss of generality, since in any feasible schedule each job j cannot finish before time p_j , and we will see later that our algorithm ensures that x_{jt} will be set to 0 when $t = 1, \dots, p_j - 1$. With these time-indexed variables, it is trivial to ensure that each job is scheduled; the only difficulty is to ensure that the machine is not required to process more than one job at a time. To do this, we observe that, for each time $t = 1, \dots, n$, the jobs completing at time t or later have total processing time at least $T - t + 1$ (by the assumption that the processing times p_j are positive integers); for conciseness, we denote this demand $D(t) = T - t + 1$. This gives the following integer program:

$$\text{minimize } \sum_{j=1}^n \sum_{t=1}^T f_j(t)x_{jt} \tag{IP}$$

$$\text{subject to } \sum_{j=1}^n \sum_{s=t}^T p_j x_{js} \geq D(t), \quad \text{for each } t = 1, \dots, T; \tag{1}$$

$$\sum_{t=1}^T x_{jt} = 1, \quad \text{for each } j = 1, \dots, n; \tag{2}$$

$$x_{jt} \in \{0, 1\}, \quad \text{for each } j = 1, \dots, n, t = 1, \dots, T.$$

We first argue that this is a valid formulation of the problem. Clearly, each feasible schedule corresponds to a feasible solution to (IP) of equal objective function value. Conversely, consider any feasible solution, and for each job $j = 1, \dots, n$, assign it the due date $d_j = t$ corresponding to $x_{jt} = 1$. If we schedule the jobs in Earliest Due Date (EDD) order, then we claim that each job $j = 1, \dots, n$, completes by its due date d_j . If we consider the constraint (1) in (IP) corresponding to $t = d_j + 1$, then since each job is assigned once, we know that $\sum_{j=1}^n \sum_{t=1}^{d_j} p_j x_{jt} \leq d_j$; in words, the jobs with due date at most d_j have total processing time at most d_j . Since each job completes by its due date, and the cost functions $f_j(\cdot)$ are nondecreasing, we have a schedule of cost no more than that of the original feasible solution to (IP).

The formulation (IP) has an unbounded integrality gap: the ratio of the optimal value of (IP) to the optimal value of its linear programming relaxation can be arbitrarily large. We strengthen this formulation by introducing a class of valid inequalities called *knapsack-cover inequalities*. To understand the starting point for our work, consider the special case of this scheduling problem in which all n jobs have a common due date D , and for each job $j = 1, \dots, n$, the cost function is 0 if the job completes by time D , and is w_j , otherwise. In this case, we select a set of jobs of total size at most D , so as to minimize the total weight of the complementary set (of late jobs). This is equivalent to the minimum-cost (covering) knapsack problem, in which we wish to select a subset of items of total size at least a given threshold, of minimum total cost. Carr, Fleischer, Leung, and Phillips [8] introduced knapsack-cover inequalities for this problem (as a variant of flow-cover inequalities introduced by Padberg, Van Roy, and Wolsey [14]) and gave an LP-rounding 2-approximation algorithm based on this formulation. On the other hand, Carr et al. also showed that the LP relaxation with knapsack-cover inequalities has an integrality gap of at least $2 - \frac{2}{n}$. Our results instead generalize a primal-dual 2-approximation algorithm based on the same formulation, which was given by Carnes and Shmoys [7]. Knapsack-cover inequalities have subsequently been used to derive approximation algorithms in a variety of other settings, including the work of Bansal & Pruhs [5] for $1|ptmn, r_j| \sum f_j$, Bansal, Buchbinder, & Naor [23], Gupta, Krishnaswamy, Kumar, & Segev [12], Bansal, Gupta, & Krishnaswamy [4], and Pritchard [15]. The 16-approximation algorithm of Bansal & Pruhs [5] for $1|| \sum f_j$ relies on a 4-approximation algorithm for so-called *generalized caching problem* which relies (implicitly) on a different time-indexed LP relaxation [6].

The idea behind the knapsack-cover inequalities is quite simple. Fix a subset of jobs $A \subseteq \{1, \dots, n\}$ that contribute towards satisfying the demand $D(t)$ for time t or later; then there is a *residual demand* from the remaining jobs of $D(t, A) := \max\{D(t) - \sum_{j \in A} p_j, 0\}$. Thus, each job $j = 1, \dots, n$ can make an effective contribution to this residual demand of $p_j(t, A) := \min\{p_j, D(t, A)\}$; that is, given the inclusion of the set A , the effective contribution of job j towards satisfying the residual demand can be at most the residual demand itself. Thus, we have the constraint: $\sum_{j \notin A} \sum_{s=t}^T p_j(t, A) x_{js} \geq D(t, A)$ for each $t = 1, \dots, T$, and each $A \subseteq \{1, \dots, n\}$. The dual LP is quite natural: there are dual variables $y(t, A)$, and a constraint that indicates, for each job j and each time $s = 1, \dots, T$, that $f_j(s)$ is at least a weighted sum of $y(t, A)$ values, and the objective is to maximize $\sum_{t,A} D(t, A)y(t, A)$.

Our primal-dual algorithm has two phases: a growing phase and a pruning phase. Throughout the algorithm, we maintain a set of jobs A_t for each time $t = 1, \dots, T$. In each iteration of the growing phase, we choose one dual variable to increase, corresponding to the demand $D(t, A_t)$ that is largest, and increase that dual variable as much as possible. This causes a dual constraint corresponding to some job j to become tight for some time t' , and so that we set $x_{jt'} = 1$, and add j to each set A_s with $s \leq t'$. Note that this may result in jobs being assigned to complete at multiple times t ; then in the pruning phase we do a

“reverse delete” that both ensures that each job is uniquely assigned, and also that the solution is minimal, in the sense that each job passes the test that if it were deleted, then some demand constraint (\square) in (IP) would be violated. It will be straightforward to show that the algorithm runs in time polynomial in n and T , which is a pseudopolynomial bound. To convert this algorithm to be polynomial, we adopt an interval-indexed formulation, where we bound the change of cost of any job to be within a factor of $(1 + \epsilon)$ within any interval. This is sufficient to ensure a (weakly) polynomial number of intervals, while degrading the performance guarantee by a factor of $(1 + \epsilon)$, and this yields the desired result. One surprising consequence of this interval-indexed approach, combined with the generality of the objective function structure, is that the same algorithm can be used to obtain the same performance guarantee in the more general setting in which the machine has a time-dependent speed $s(t)$ (where we assume that we can compute the processing capacity of the machine for any time interval $(t_1, t_2]$ in polynomial time); this greatly generalizes and improves upon the randomized ϵ -approximation algorithm of Epstein, Levin, Marchetti-Spaccamela, Megow, Mestre, Skutella, and Stougie [9] for this setting in the special case where the objective function is to minimize $\sum_j w_j C_j$ (though the results in [9] have the advantage of not needing to know the speed function in advance).

The main question left open by this work is whether similar techniques can yield analogous results for the analogous problem $1|r_j, pmtn|\sum f_j$. Bansal and Pruhs gave a $O(\log \log(nT))$ -approximation algorithm for this problem, and yet there is no evidence to suggest that there does not exist an approximation algorithm with a constant performance guarantee (or potentially even a polynomial approximation scheme). Since there is no advantage to preemption if all release dates are equal to 0, it follows that, as for $1|\sum f_j$, the problem is strongly NP-hard, and hence no fully polynomial approximation scheme exists unless $P=NP$. One interesting point of contrast is the “bottleneck” or “min-max” analogue of these two problems: $1|f_{\max}$ and $1|r_j, pmtn|f_{\max}$. Both problems are solvable in polynomial time, by variants of the least cost last rule, by results of Lawler [13] and of Baker, Lawler, Lenstra, and Rinnooy Kan [1], respectively.

2 A Pseudopolynomial Algorithm for $1|\sum f_j$

We give a primal-dual algorithm that runs in pseudopolynomial time and has a performance guarantee of 2 and is based on the following LP relaxation:

$$\begin{aligned}
 \min \quad & \sum_{j=1}^n \sum_{t=1}^T f_j(t)x_{jt} & (P) \\
 \text{s.t.} \quad & \sum_{j \notin A} \sum_{s=t}^T p_j(t, A)x_{js} \geq D(t, A), & \text{for each } t = 1, \dots, T, A \subseteq \{1, \dots, n\}; \quad (3) \\
 & x_{jt} \geq 0, & \text{for each } j = 1, \dots, n, t = 1, \dots, T.
 \end{aligned}$$

Notice that the assignment constraints (2) are not included in (P). In fact, the following lemma shows that they are redundant, given the knapsack-cover inequalities. This leaves a much more tractable formulation on which to base the design of our primal-dual algorithm, and is one of the reasons that we are able to obtain an improved guarantee.

Lemma 1. *Let x be a feasible solution to the linear programming relaxation (P). Then there is a feasible solution \bar{x} of no greater cost that also satisfies the assignment constraints (2).*

Proof. First, by considering the constraint (3) with the set $A = \{1, \dots, n\} - \{k\}$ and $t = 1$, it is easy to show that for any feasible solution x of (P), we must have $\sum_{s=1}^T x_{ks} \geq 1$ for each job k .

We next show that each job is assigned at most once. We may assume without loss of generality that x is a feasible solution for (P) in which $\sum_{j=1}^n \sum_{s=1}^T x_{js}$ is minimum. Suppose, for a contradiction, that $\sum_{s=1}^T x_{js} > 1$ for some job j , and let t be the largest time index where the partial sum $\sum_{s=t}^T x_{js} \geq 1$. Consider the solution \bar{x} where $\bar{x}_{ks} := 0$ if $k = j$ and $s < t$, $1 - \sum_{s=t+1}^T x_{js}$ if $k = j$ and $s = t$, and x_{ks} otherwise. It is easy to verify that the modified solution \bar{x} is a feasible solution to (P) where $\sum_{j=1}^n \sum_{s=1}^T \bar{x}_{js} < \sum_{j=1}^n \sum_{s=1}^T x_{js}$. This gives the desired contradiction. Finally, since $\bar{x} \leq x$ componentwise and the objective $f_j(t)$ is nonnegative, it follows that \bar{x} is a solution of no greater cost than x . \square

Taking the dual of (P) gives:

$$\begin{aligned} \max \quad & \sum_{t=1}^T \sum_A D(t, A)y(t, A) & (D) \\ \text{s.t.} \quad & \sum_{t=1}^s \sum_{A: j \notin A} p_j(t, A)y(t, A) \leq f_j(s); & \text{for each } j = 1, \dots, n, s = 1, \dots, T; \\ & y(t, A) \geq 0 & \text{for each } t = 1, \dots, T, A \subseteq \{1, \dots, n\}. \end{aligned}$$

We now give the primal-dual algorithm for the scheduling problem $1||\sum f_j$. The algorithm consists of two phases: a growing phase and a pruning phase.

The growing phase constructs a feasible solution x to (P) over a series of iterations. For each $t = 1, \dots, T$, we let A_t denote the set of jobs that are set to finish at time t or later by the algorithm, and thus contribute towards satisfying the demand $D(t)$. In each iteration, we set a variable x_{jt} to 1 and add j to A_s for all $s \leq t$. We continue until all demands $D(t)$ are satisfied. Specifically, in the t^{th} iteration, the algorithm select $t^k := \operatorname{argmax}_t D(t, A_t)$, which is the time index that has the largest residual demand with respect to the current partial solution. If there are ties, we choose the *largest* such time index to be t^k (this is not essential to the correctness of the algorithm – only for consistency and efficiency). If $D(t^k, A_{t^k}) = 0$, then we must have $\sum_{j \in A_t} p_j \geq D(t)$ for each $t = 1, \dots, T$; all demands have been satisfied and the growing phase terminates. Otherwise, we increase the dual variable $y(t^k, A_{t^k})$ until some dual constraint

Algorithm 1. Primal-Dual Algorithm for $1||\sum f_j$

```

// Initialization
 $x, y, k \leftarrow 0; A_t = \emptyset, (t = 1, \dots, T);$ 
 $t^0 := \operatorname{argmax}_t D(t, A_t)$  (break ties by choosing largest time index);
// Growing phase
while  $D(t^k, A_{t^k}) > 0$  do
    Increase  $y_{t^k, A_{t^k}}$  until a dual constraint (4) with right hand side  $f_j(t)$ 
    becomes tight (break ties by choosing the largest time index);
     $x_{jt} \leftarrow 1;$ 
     $A_s \leftarrow A_s \cup \{j\}$  for each  $s \leq t;$ 
     $k \leftarrow k + 1;$ 
     $t^k := \operatorname{argmax}_t D(t, A_t)$  (break ties by choosing largest time index);
// Pruning phase
Consider  $\{(j, t) : x_{jt} = 1\}$  in reverse order in which they are set to 1;
if  $j \in A_{t+1}$  then
     $x_{jt} \leftarrow 0$ 
else if  $\sum_{j' \in A_s \setminus \{j\}} p_{j'} \geq D(s)$  for all  $s \leq t$  where  $j$  is added to  $A_s$  in the same
iteration of growing phase then
     $x_{j,t} \leftarrow 0;$ 
     $A_s \leftarrow A_s \setminus \{j\}$  for all such  $s;$ 

// Output schedule
for  $j \leftarrow 1$  to  $n$  do
     $\lfloor$  Set due date  $d_j$  of job  $j$  to time  $t$  if  $x_{jt} = 1;$ 
Schedule jobs using EDD rule;

```

(4) with right-hand side $f_j(t)$ becomes tight. We set $x_{jt} = 1$ and add j to A_s for all $s \leq t$ (if j is not yet in A_s). If multiple constraints become tight at the same time, we pick the one with the *largest* time index (and if there are still ties, just pick one of these jobs arbitrarily). However, at the end of the growing phase, we might have too many variables set to 1, thus we proceed to the pruning phase.

The pruning phase is a “reverse delete” procedure that checks each variable x_{jt} that is set to 1, in decreasing order of the iteration k in which that variable was set in the growing phase. We attempt to set x_{jt} back to 0 and correspondingly delete jobs from A_t , provided this does not violate the feasibility of the solution. Specifically, for each variable $x_{jt} = 1$, if j is also in A_{t+1} then we set $x_{jt} = 0$. It is safe to do so, since in this case, there must exist $t' > t$ where $x_{jt'} = 1$, and as we argued in Lemma 1, it is redundant to have x_{jt} also set to 1. Otherwise, if $j \notin A_{t+1}$, we check if $\sum_{j' \in A_s \setminus \{j\}} p_{j'} \geq D(s)$ for each time index s where j has been added to A_s in the same iteration of the growing phase. If so, then j is not needed to satisfy the demand at time s . Hence, we remove j from all such A_s and set $x_{jt} = 0$. We will show that at the end of the pruning phase, each job j has exactly one x_{jt} set to 1. Hence, we set this time t as the *due date* of job j .

Finally, the algorithm outputs a schedule by sequencing the jobs in Earliest Due Date (EDD) order. We give pseudo-code for this in the figure Algorithm 1 above.

Analysis Throughout the algorithm's execution, we maintain both a solution x along with the sets A_t , for each $t = 1, \dots, T$; an easy inductive argument shows that the algorithm maintains the following invariant.

Lemma 2. *Throughout the algorithm, $j \in A_s$ if and only if there exists $t \geq s$ such that $x_{jt} = 1$.*

Note that this lemma also implies that the sets A_t are nested; i.e., for any two time indices $s < t$, it follows that $A_s \supseteq A_t$. Using the above lemma, we will show that algorithm produces a feasible solution to (P) and (D).

Lemma 3. *The algorithm produces a feasible solution x to (P) that is integral \mathcal{E} satisfies the assignment constraints (2), as well as a feasible solution y to (D).*

Proof. First note that, by construction, the solution x is integral. The algorithm starts with the all-zero solution to both (P) and (D), which is feasible for (D) but infeasible for (P). Showing that dual feasibility is maintained throughout the algorithm is straightforward. Next we show that at termination, the algorithm obtains a feasible solution for (P).

At the end of the growing phase, all residual demands $D(t, A_t)$ are zero, and hence, $\sum_{j \in A_t} p_j \geq D(t)$ for each $t = 1, \dots, T$. By construction of the pruning phase, the same still holds when the algorithm terminates.

Next, we argue that for each job j there is exactly one t where $x_{jt} = 1$ when the algorithm terminates. Notice that $D(1)$ (the demand at time 1) is T , which is also the sum of processing time of all jobs; hence A_1 must include every job to satisfy $D(1)$. By Lemma 2, this implies each job has at least some time t for which $x_{jt} = 1$ when the growing phase terminates. On the other hand, from the pruning step (in particular, the first *if* statement in the pseudocode), each job j has x_{jt} set to 1 for at most one time t . However, since no job can be deleted from A_1 , by Lemma 2, we see that, for each job j , there is still at least one x_{jt} set to 1 at the end of the pruning phase. Combining the two, we see that each job j has one value t for which $x_{jt} = 1$.

By invoking Lemma 2 for the final solution x , we have that $\sum_{s=t}^T \sum_{j=1}^n p_j x_{js} \geq D(t)$. Furthermore, x also satisfies the constraint $\sum_{t=1}^T x_{jt} = 1$, as argued above. Hence, x is feasible for (IP), which implies the feasibility for (P). \square

Since all cost functions $f_j(\cdot)$ are nondecreasing, it is easy to show that given a feasible integral solution x to (P) that satisfies the assignment constraints (2), the following schedule costs no more than the objective value for x : set the due date $d_j = t$ for job j , where t is the unique time such that $x_{jt} = 1$, and sequence in EDD order (meeting all due dates).

Lemma 4. *Given a feasible integral solution to (P) that satisfies the assignment constraint (2), the EDD schedule is a feasible schedule with cost no more than the value of the given primal solution.*

Next we analyze the cost of the schedule returned by the algorithm. Given the above lemma, it suffices to show that the cost of the primal solution is no more

than twice the cost of the dual solution; the weak duality theorem of linear programming then implies that our algorithm has a performance guarantee of 2.

We first introduce some notation used in the analysis. Given the final solution \bar{x} returned by the algorithm, define $\bar{J}_t := \{j : \bar{x}_{jt} = 1\}$, and $\bar{A}_t := \{j : \exists \bar{x}_{jt'} = 1, t' \geq t\}$. In words, \bar{A}_t is the set of jobs that contribute towards satisfying the demand at time t in the final solution; hence, we say that j covers t if $j \in \bar{A}_t$. Let x^k be the partial solution of (P) at the beginning of the k^{th} iteration of the growing phase. We define J_t^k and A_t^k analogously with respect to x^k . Next we prove the key lemma in our analysis.

Lemma 5. $\sum_{s=t}^T \sum_{j \in \bar{J}_s \setminus A} p_j(s, A) < 2D(t, A)$, for each (t, A) with $y(t, A) > 0$.

Proof. Recall that the algorithm increases only one dual variable in each iteration of the growing phase. Suppose that $y(t, A)$ is the variable chosen in iteration k , i.e., $t = t^k$. Using the notation introduced above, we can write $y(t, A)$ as $y(t^k, A_{t^k}^k)$. However, for notational convenience, we shall denote the set $A_{t^k}^k$ as A^k . Then the lemma can be restated as $\sum_{j \in \bar{A}_{t^k} \setminus A^k} p_j(t^k, A^k) < 2D(t^k, A^k)$. We can interpret the set on the left-hand side as the jobs that cover the demand of t^k that are added to the solution after the start of iteration k .

First, suppose time t^k is *critical* with respect to \bar{x} and A^k , meaning there exists some job ℓ in $\bar{A}_{t^k} \setminus A^k$ such that ℓ cannot be deleted from \bar{A}_{t^k} for \bar{x} to remain feasible. This can be expressed as $\sum_{j \in \bar{A}_{t^k} \setminus (A^k \cup \ell)} p_j < D(t^k, A^k)$. Notice that each of these jobs in the above summation must have processing time less than $D(t^k, A^k)$, and thus by definition, $p_j = p_j(t^k, A^k)$. Hence, $\sum_{j \in \bar{A}_{t^k} \setminus (A^k \cup \ell)} p_j(t^k, A^k) < D(t^k, A^k)$. Also, $p_\ell(t^k, A_{t^k}^k) \leq D(t^k, A^k)$, by definition. Adding these two together gives $\sum_{j \in \bar{A}_{t^k} \setminus A^k} p_j(t^k, A^k) < 2D(t^k, A^k)$.

Hence, if t^k is critical we have the desired result. Now we will argue by contradiction that this must be the case. Suppose otherwise; then for every job ℓ in $\bar{A}_{t^k} \setminus A^k$, we have that $\sum_{j \in \bar{A}_{t^k} \setminus (A^k \cup \ell)} p_j \geq D(t^k, A^k)$. We first argue that there must exist some time t_ℓ that is critical with respect to \bar{x} and $A_{t_\ell}^k$ because of job ℓ ; i.e.,

$$\sum_{j \in \bar{A}_{t_\ell} \setminus (A_{t_\ell}^k \cup \ell)} p_j < D(t_\ell, A_{t_\ell}^k). \tag{5}$$

Suppose not; from the definition of $D(t_\ell, A_{t_\ell}^k)$, we must have that $\sum_{j \in \bar{A}_{t_\ell} \setminus (A_{t_\ell}^k \cup \ell)} p_j + \sum_{j \in A_{t_\ell}^k} p_j \geq D(t_\ell)$ for each time t_ℓ that job ℓ covers. This, combined with the fact that ℓ is considered in the pruning phase before any of the jobs in $A_{t_\ell}^k$ (since ℓ is added after the start of iteration k), implies that ℓ should have been deleted in the pruning phase, which is a contradiction.

Hence, such a time t_ℓ must exist for each job ℓ in $\bar{A}_{t^k} \setminus A^k$. If there are multiple time indices such that (5) holds for job ℓ , let t_ℓ be the earliest such time. Consider the following two cases:

Case 1 Suppose there exists some job ℓ with $t_\ell < t^k$. Notice that each job that covers t^k also covers t_ℓ . Hence, the set of jobs in the final solution that covers t_ℓ

added after the start of iteration k of the growing phase is a superset of those that covers t^k and added after the start of iteration k , i.e., $\bar{A}_{t_\ell} \setminus (A_{t_\ell}^k \cup \ell) \supseteq \bar{A}_{t^k} \setminus (A^k \cup \ell)$ and we have that

$$\sum_{j \in \bar{A}_{t_\ell} \setminus (A_{t_\ell}^k \cup \ell)} p_j \geq \sum_{j \in \bar{A}_{t^k} \setminus (A^k \cup \ell)} p_j \geq D(t^k, A^k) \geq D(t_\ell, A_{t_\ell}^k),$$

where the second inequality follows from our assumption that t^k is not critical, and the last inequality follows from the definition of t^k having the largest residual demand. This gives a contradiction to (5); thus Case 1 is impossible.

Case 2 Otherwise, all $t_\ell > t^k$. Pick ℓ so that t_ℓ is the earliest among all jobs in $\bar{A}_{t^k} \setminus A^k$. Notice that each job that covers t_ℓ must cover t^k as well. However, by the assumption that t^k is not critical and the choice of t_ℓ , the set of jobs that covers t_ℓ added after the start of iteration k of the growing phase is the same as those that cover t^k and are added after the start of iteration k , i.e., $\bar{A}_{t_\ell} \setminus (A_{t_\ell}^k \cup \ell) = \bar{A}_{t^k} \setminus (A^k \cup \ell)$. Then we can derive a contradiction similar to Case 1:

$$\sum_{j \in \bar{A}_{t_\ell} \setminus (A_{t_\ell}^k \cup \ell)} p_j = \sum_{j \in \bar{A}_{t^k} \setminus (A^k \cup \ell)} p_j \geq D(t^k, A^k) > D(t_\ell, A_{t_\ell}^k).$$

This gives a contradiction to (5); thus Case 2 is also impossible. Combining the two cases, we see that t^k must be critical with respect to \bar{x} , giving us the desired result. \square

Now we can show our main theorem.

Theorem 1. *The primal-dual algorithm produces a schedule for $1 || \sum f_j$ with cost at most twice the optimum.*

Proof. It suffices to show that the cost of the schedule is no more than twice the dual objective value. The cost of our solution is denoted by $\sum_{t=1}^T \sum_{j \in \bar{J}_t} f_j(t)$. We have that

$$\begin{aligned} \sum_{t=1}^T \sum_{j \in \bar{J}_t} f_j(t) &= \sum_{t=1}^T \sum_{j \in \bar{J}_t} \sum_{s=1}^t \sum_{A: j \notin A} p_j(s, A) y(s, A) \\ &= \sum_{s=1}^T \sum_A y(s, A) \left(\sum_{t=s}^T \sum_{j \in \bar{J}_t \setminus A} p_j(s, A) \right) \end{aligned}$$

The first line is true because we set $x_{jt} = 1$ only if the dual constraint is tight, and the second line is by interchanging the order of summations and using the relation $s \leq t$. Now, from Lemma 5 we know that $\sum_{t=s}^T \sum_{j \in \bar{J}_t \setminus A} p_j(s, A) < 2D(s, A)$. Hence it follows that

$$\sum_{s=1}^T \sum_A y_{sA} \left(\sum_{t=s}^T \sum_{j \in \bar{J}_t \setminus A} p_j(s, A) \right) < \sum_{s=1}^T \sum_A 2D(s, A) y(s, A),$$

where the right-hand side is twice the dual objective. The result now follows, since the dual objective is a lower bound of the cost of the optimal schedule. \square

3 A $(2 + \epsilon)$ -Approximation Algorithm

We now give a polynomial-time $(2 + \epsilon)$ -approximation algorithm for $1 || \sum f_j$. This is achieved by simplifying the input via rounding in a fairly standard fashion, and then running the primal-dual algorithm on the LP relaxation of the simplified input, which only has a polynomial number of interval-indexed variables. A similar approach was employed in the work of Bansal & Pruhs [5].

Fix a constant $\epsilon > 0$. We start by constructing n partitions of the time indices $\{1, \dots, T\}$, one partition for each job, according to its cost function. Focus on some job j . First, the set of time indices $I_j^0 = \{t : f_j(t) = 0\}$ are those of class 0; class 1 is the set of indices $I_j^1 = \{t : 0 < f_j(t) \leq 1\}$; finally, class $k = 2, 3, \dots$ is the set of indices $I_j^k = \{t : (1 + \epsilon)^{k-2} < f_j(t) \leq (1 + \epsilon)^{k-1}\}$. (We can bound the number of classes for job j by $1 + \log_{1+\epsilon} f_j(T)$.) Let ℓ_j^k denote the minimum element in I_j^k (if the set is non-empty), and let \mathcal{T}_j be the set of all left endpoints ℓ_j^k . Finally, let $\mathcal{T} = \cup_{j=1}^n \mathcal{T}_j$. Notice that the time $t = 1$ is in \mathcal{T} . Index the elements such that $\mathcal{T} := \{t_1, \dots, t_\tau\}$ where $1 = t_1 < t_2 < \dots < t_\tau$. We then compute a master partition of the time horizon T into the intervals $\mathcal{I} = \{[t_1, t_2 - 1], [t_2, t_3 - 1], \dots, [t_{\tau-1}, t_\tau - 1], [t_\tau, T]\}$. There are two key properties of this partition: the cost of any job changes by at most a factor of $1 + \epsilon$ as its completion time varies within an interval, and the number of intervals is a polynomial in n , $\log P$ and $\log W$. Here P denotes the length of the longest job and $W = \max_{j,t} (f_j(t) - f_j(t - 1))$, the maximum increase in cost function $f_j(t)$ in one time step over all jobs j and times t .

Next we define a modified cost function $f'_j(t)$ for each time $t \in \mathcal{T}$; in essence, the modified cost is an upper bound on the cost of job j completing in the interval for which t is the left endpoint. More precisely, for $t_i \in \mathcal{T}$, let $f'_j(t_i) := f_j(t_{i+1} - 1)$. Notice that, by construction, we have that $f_j(t) \leq f'_j(t) \leq (1 + \epsilon)f_j(t)$ for each $t \in \mathcal{T}$. Consider the following integer programming formulation with variables x'_{jt} for each job j and each time $t \in \mathcal{T}$; we set the variable x'_{jt_i} to 1 to indicate that job j completes within the interval $[t_i, t_{i+1} - 1]$. The demand $D(t)$ is defined the same way as before.

$$\text{minimize } \sum_{j=1}^n \sum_{t \in \mathcal{T}} f'_j(t) x'_{jt} \tag{IP'}$$

$$\text{subject to } \sum_{j=1}^n \sum_{s \in \mathcal{T}: s \geq t} p_j x'_{js} \geq D(t), \tag{6}$$

for each $t \in \mathcal{T}$;

$$\sum_{t \in \mathcal{T}} x'_{jt} = 1, \tag{7}$$

for each $j = 1, \dots, n$;

$$x'_{jt} \in \{0, 1\}, \tag{8}$$

for each $j = 1, \dots, n, t \in \mathcal{T}$.

The next two lemmas relate (IP') to (IP).

Lemma 6. *If there is a feasible solution x to (IP) with objective value v , then there is a feasible solution x' to (IP') with objective value at most $(1 + \epsilon)v$.*

Proof. Suppose $x_{jt} = 1$ where t lies in the interval $[t_i, t_{i+1} - 1]$ as defined by the time indices in \mathcal{T} , then we construct a solution to (IP') by setting $x'_{jt_i} = 1$. It is straightforward to check x' is feasible for (IP'), and by construction $f'_j(t_i) = f_j(t_{i+1} - 1) \leq (1 + \epsilon)f_j(t)$.

Lemma 7. *If there is a feasible solution x' to (IP') with objective value v' , then there is a feasible solution x to (IP) with objective value v' .*

Proof. Suppose $x'_{jt} = 1$, where $t = t_i$; then we construct a solution to (IP) by setting $x_{j,t_{i+1}-1} = 1$. Notice that the time $t_{i+1} - 1$ is the right endpoint to the interval $[t_i, t_{i+1} - 1]$. By construction, $f_j(t_{i+1} - 1) = f'_j(t_i)$; hence, the cost of solution x is also v' . To check its feasibility, it suffices to see that the constraint corresponding to $D(t_i)$ is satisfied. This uses the fact that within the interval $[t_i, t_{i+1} - 1]$, $D(t)$ is largest at t_i and that the constraint corresponding to $D(t)$ contains all variables x_{js} with a time index s such that $s \geq t$.

Using the two lemmas above, we see that running the primal-dual algorithm using the LP relaxation of (IP') strengthened by the knapsack-cover inequalities gives us a $2(1 + \epsilon)$ -approximation algorithm for the scheduling problem $1 || \sum f_j$. Hence we have the following result:

Theorem 2. *For each $\epsilon > 0$, there is a $(2 + \epsilon)$ -approximation algorithm for the scheduling problem $1 || \sum f_j$.*

The combination of the interval-indexed formulation along with the generality of the objective function allows us to capture even more general problems within the same framework.

Specifically, we consider the setting in which the machine runs at a time-varying speed $s(t)$, so that within the interval $(t, t']$ the machine has the capacity to process $\int_t^{t'} s(t)dt$ units of processing, and extend Theorem 2 to this setting. No constant approximation algorithm was previously known for this problem, even just in the case where the speed of the machine was either 0 or 1; that is, there are specified intervals in which the machine is not available. The details of this generalization will be given in the full version of this paper.

References

1. Baker, K.R., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Preemptive scheduling of a single machine to minimize maximum cost subject to release dates and precedence constraints. *Operations Research* 31, 381–386 (1983)
2. Bansal, N., Buchbinder, N., Naor, J.: A primal-dual randomized algorithm for weighted paging. In: *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, pp. 507–517 (2007)

3. Bansal, N., Buchbinder, N., Naor, J.: Randomized competitive algorithms for generalized caching. In: Proceedings of the 40th Annual ACM Symposium on the Theory of Computing, pp. 235–244 (2008)
4. Bansal, N., Gupta, A., Krishnaswamy, R.: A constant factor approximation algorithm for generalized min-sum set cover. In: Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1539–1545 (2010)
5. Bansal, N., Pruhs, K.: The geometry of scheduling. In: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science, pp. 407–414 (2010)
6. Bar-Noy, A., Bar-Yehuda, R., Freund, A., Naor, J., Schieber, B.: A unified approach to approximating resource allocation and scheduling. *Journal of the ACM* 48(5), 1069–1090 (2001)
7. Carnes, T., Shmoys, D.: Primal-dual schema for capacitated covering problems. In: Lodi, A., Panconesi, A., Rinaldi, G. (eds.) IPCO 2008. LNCS, vol. 5035, pp. 288–302. Springer, Heidelberg (2008)
8. Carr, R.D., Fleischer, L., Leung, V.J., Phillips, C.A.: Strengthening integrality gaps for capacitated network design and covering problems. In: Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 106–115 (2000)
9. Epstein, L., Levin, A., Marchetti-Spaccamela, A., Megow, N., Mestre, J., Skutella, M., Stougie, L.: Universal sequencing on a single machine. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 230–243. Springer, Heidelberg (2010)
10. Garey, M.R., Johnson, D.S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York (1979)
11. Graham, R.L., Lawler, E.L., Lenstra, J.K., Rinnooy Kan, A.H.G.: Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics* 5, 287–326 (1979)
12. Gupta, A., Krishnaswamy, R., Kumar, A., Segev, D.: Scheduling with outliers. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX 2009. LNCS, vol. 5687, pp. 149–162. Springer, Heidelberg (2009)
13. Lawler, E.L.: Optimal sequencing of a single machine subject to precedence constraints. *Management Science* 19, 544–546 (1973)
14. Padberg, M.W., van Roy, T.J., Wolsey, L.A.: Valid inequalities for fixed charge problems. *Operations Research* 33, 842–861 (1985)
15. Pritchard, D.: Approximability of sparse integer programs. In: Proceedings of the 17th Annual European Symposium on Algorithms, pp. 83–94 (2009)

A $(1 + \ln 2)$ -Approximation Algorithm for Minimum-Cost 2-Edge-Connectivity Augmentation of Trees with Constant Radius

Nachshon Cohen and Zeev Nutov

The Open University of Israel,
nachshonc@gmail.com, nutov@openu.ac.il

Abstract. We consider the *Tree Augmentation* problem: given a graph $G = (V, E)$ with edge-costs and a tree T on V disjoint to E , find a minimum-cost edge-subset $F \subseteq E$ such that $T \cup F$ is 2-edge-connected. *Tree Augmentation* is equivalent to the problem of finding a minimum-cost edge-cover $F \subseteq E$ of a laminar set-family. The best known approximation ratio for *Tree Augmentation* is 2, even for trees of radius 2. As laminar families play an important role in network design problems, obtaining a better ratio is a major open problem in network design. We give a $(1 + \ln 2)$ -approximation algorithm for trees of constant radius. Our algorithm is based on a new decomposition of problem solutions, which may be of independent interest.

1 Introduction

We consider the following problem:

Tree Augmentation

Instance: A graph $G = (V, E)$ with edge-costs $\{c(e) : e \in E\}$, and a tree T on V disjoint to E .

Objective: Find a minimum-cost edge-set $F \subseteq E$ such that $T \cup F$ is 2-edge-connected.

The case when T is a path reduces to the *Shortest Path* problem (c.f. [7]), and the case when T is a star is equivalent to the *Minimum-Cost Edge-Cover* problem. *Tree Augmentation* is equivalent to the problem of finding a minimum cost edge-cover of a laminar set-family; namely, given a graph $G = (V, E)$ with edge-costs and a laminar set-family \mathcal{L} on V , we seek a minimum cost edge-set $F \subseteq E$ such that for every $S \in \mathcal{L}$ there is $uv \in F$ with $u \in S$ and $v \notin S$. The problem is also equivalent to the problem of augmenting a k -edge-connected graph to be $(k + 1)$ -connected by adding a minimum cost edge-set, for odd k ; this is since when k is odd, the minimum cuts of a k -edge-connected graph form a laminar set-family. In general, laminar set-families play an important role in network design problems; see [6] and surveys in [5,7,8], for various network design problems and applications of laminar set-families.

Fredrickson and Jájá [4] gave a 2-approximation algorithm for *Tree Augmentation*, and showed that it is NP-hard even for trees of radius 2 (the radius

$R = R(T)$ of a tree T is $\lceil D/2 \rceil$, where D is the diameter of T). Cheriyan, Jordán and Ravi [1] proved that **Tree Augmentation** is NP-hard also in the case of unit costs when E is a cycle on the leaves of T , and gave a $4/3$ -approximation algorithm for this version. They also conjectured that a standard LP-relaxation for the problem has integrality gap less than 2, while in [2] it is shown that the integrality gap is at least $3/2$.

Achieving a ratio better than 2 for **Tree Augmentation**, even for the particular case of unit costs, was posed as a major open problem in connectivity network design in the survey by Khuller [7]. This open question was resolved by Nagamochi [10] that gave a $(1.875 + \epsilon)$ -approximation scheme for this version. The currently best known approximation ratio for **Tree Augmentation** with unit costs is $3/2$, by Even, Kortsarz, and Nutov [3]. Even better ratios are known for unit costs when every edge in E connects two leaves of T [9]: $17/12$ for general trees, $11/8$ for trees of radius 3, and $4/3$ for trees of radius 2.

However, for arbitrary costs, no ratio better than 2 is known, not even for trees of radius 2. We prove the following.

Theorem. *Tree Augmentation admits an algorithm that computes a $(1 + \ln 2)$ -approximate solution in time $n^{3^{h(T)}} \cdot \text{poly}(n)$, where $n = |V|$ and $h(T)$ is the radius of T . In particular, **Tree Augmentation** on trees with radius bounded by a constant admits a $(1 + \ln 2)$ -approximation algorithm.*

Our algorithm is based on a new decomposition of **Tree Augmentation** feasible solutions, which may be of independent interest.

2 Proof of the Theorem

2.1 A Local Replacement Algorithm for Set-Cover

We start by describing a generic local-replacement algorithm for the following well known problem.

Set-Cover

Instance: A collection E of subsets of a groundset T with costs $\{c(e) : e \in E\}$.

Objective: A minimum-costs subcollection $F \subseteq E$ such that $T \subseteq \bigcup_{e \in F} e$.

Definition 1. *Given an instance of Set-Cover and $S, F \subseteq E$ let*

$$R_F(S) = \left\{ f \in F : f \subseteq \bigcup_{e \in S} e \right\} .$$

We say that a set-family $\mathcal{S} \subseteq 2^E$ overlaps $F \subseteq E$ if $\bigcup_{S \in \mathcal{S}} R_F(S) = F$, namely, if for every $f \in F$ there is $S \in \mathcal{S}$ with $f \in R_F(S)$. We say that $F \subseteq E$ is q -overlapped by $F^ \subseteq E$ if F^* admits a partition \mathcal{S} into parts of size at most q that overlaps F .*

The following statement is immediate.

Fact 1. *Let $F \subseteq E$ be a feasible solution. Then $F \setminus R_F(S) \cup S$ is also a feasible solution for any $S \subseteq E$.*

The following technique was introduced by Zelikovsky in [12] in the context of the Steiner Tree problem. We reinterpret Zelikovsky’s result in a more general Set-Cover setting.

Lemma 1. *Suppose that for an instance of Set-Cover we are given an α -approximate solution F_0 that is q -overlapped by some optimal solution F^* . Then the problem admits a $(1 + \ln \alpha)$ -approximation in $m^q \cdot \text{poly}(m)$ time, where $m = |E|$.*

Proof. Let \mathcal{S} be a partition of F^* into parts of size at most q each that overlaps F . Clearly, $\sum_{S \in \mathcal{S}} c(S) = \text{opt}$. For any $F \subseteq F_0$ we have $\sum_{S \in \mathcal{S}} c(R_F(S)) \geq c(F)$, hence by an averaging argument there exist $S \in \mathcal{S}$ such that

$$\frac{c(R_F(S))}{c(F)} \geq \frac{c(S)}{\text{opt}}. \tag{1}$$

The algorithm is as follows.

Initialization: $I \leftarrow \emptyset, F \leftarrow F_0$.

While $F \neq \emptyset$ *do:*

- Find $S \subseteq E \setminus I$ with $|S| \leq q$ satisfying (I).
- If $c(R_F(S)) \leq c(S)$ then STOP and Return $F \cup I$;
- Else do $F \leftarrow F \setminus R_F(S)$ and $I \leftarrow I \cup S$.

EndWhile

Return $F \cup I$.

The time complexity is straightforward, and feasibility follows from Fact I. We prove the approximation ratio. Let S_i be the set picked at iteration i and let F_i be the set stored in F after iteration i . When the algorithm stops, either $F = \emptyset$ or $c(R_F(S)) \leq c(S)$; note that the later case implies that $c(F) \leq \text{opt}$, by (I). In both cases, there exist an iteration j such that $c(F_{j-1}) > \text{opt} \geq c(F_j)$. Hence there exists $\theta \in (0, 1]$ such that $c(F_{j-1}) - \theta \cdot c(R_{F_{j-1}}(S_j)) = \text{opt}$. Note that $c(F \cup I)$ is decreasing after each iteration, hence the cost of the solution produced by the algorithm is at most

$$c(F \cup I) \leq c(F_{j-1}) - \theta \cdot c(R_{F_{j-1}}(S_j)) + \sum_{i=1}^{j-1} c(S_i) + \theta c(S_j) = \text{opt} + \sum_{i=1}^{j-1} c(S_i) + \theta c(S_j).$$

We can assume w.l.o.g that $c(S) \geq 1$ for all $S \in E$. Since at each iteration F_i, S_i satisfy (I), we have

$$c(F_{i+1}) = \left(1 - \frac{c(R_{F_i}(S_i))}{c(F_i)}\right) c(F_i) \leq \left(1 - \frac{c(S_i)}{\text{opt}}\right) c(F_i) \leq \left(1 - \frac{1}{\text{opt}}\right)^{c(S_i)} c(F_i).$$

Applying the same argument for iteration j we get

$$\begin{aligned} \text{opt} = c(F_{j-1}) - \theta c(R_{F_{j-1}}(S)) &\leq \left(1 - \frac{1}{\text{opt}}\right)^{\theta c(S_j)} c(F_{j-1}) \\ &\leq \left(1 - \frac{1}{\text{opt}}\right)^{\sum_{i=1}^{j-1} c(S_i) + \theta c(S_j)} \cdot c(F_0) \\ &\leq \left(1 - \frac{1}{\text{opt}}\right)^{\sum_{i=1}^{j-1} c(S_i) + \theta c(S_j)} \cdot \alpha \cdot \text{opt} \end{aligned}$$

This implies

$$c(F \cup I) \leq \text{opt} + \sum_{i=1}^{j-1} c(S_i) + \theta c(S_j) \leq \text{opt} + \ln \alpha \cdot \text{opt} = (1 + \ln \alpha) \cdot \text{opt} .$$

The lemma follows. □

2.2 Algorithm for Tree Augmentation

Given an instance of **Tree Augmentation**, we will call the edges in E *links*, to distinguish them from the edges of the tree T . For $u, v \in V$ let T_{uv} denote the (unique) uv -path in T . We say that a link $uv \in E$ *covers* an edge $e \in T$ if $e \in T_{uv}$. It is well known and easy to see that $F \subseteq E$ is a feasible solution to an instance of **Tree Augmentation** if, and only if, F covers all the edges of T . Hence **Tree Augmentation** is equivalent to the problem of finding a minimum-cost link-set $F \subseteq E$ that covers all the edges of T ; namely, **Tree Augmentation** can be casted as a **Set-Cover** problem with groundset being the edge-set of T , and the collection of sets obtained by replacing every link $e = uv \in E$ by the set $T_e = T_{uv}$ of cost $c(e)$. In this setting, the restriction of Definition [1](#) to **Tree Augmentation** can be formulated as follows.

Definition 2. *Given an instance of **Tree Augmentation** and $S, F \subseteq E$ let*

$$R_F(S) = \left\{ f \in F : T_f \subseteq \bigcup_{e \in S} T_e \right\} .$$

We say that a set-family $\mathcal{S} \subseteq 2^E$ of links overlaps a set $F \subseteq E$ of links if $\bigcup_{S \in \mathcal{S}} R_F(S) = F$, namely, if for every $f \in F$ there is $S \in \mathcal{S}$ with $f \in R_F(S)$. We say that $F \subseteq E$ is q -overlapped by $F^ \subseteq E$ if F^* admits a partition \mathcal{S} into parts of size at most q that overlaps F .*

To apply Lemma [1](#), we would like to show that given an instance of **Tree Augmentation**, one can find in polynomial time a 2-approximate solution F_0 that is q -overlapped by some optimal solution F^* , for $q = 3^{h(T)-1}$. However, for this to be true, we need to apply a certain transformation to modify the instance, as is explained bellow.

Definition 3. A link $u'v'$ is a shadow of a link uv if $T_{u'v'}$ is a subpath of T_{uv} . We say that F is a shadow-minimal cover of T if for every link $uv \in F$, removing uv or replacing uv by any proper shadow of uv results in an edge-set that is not a cover of T .

Given an instance of Tree Augmentation, we can obtain an equivalent instance by applying “shadow completion”: for every existing link $e \in E$, add all its shadows, of cost $c(e)$ each (if parallel links arise, then for every inclusion-maximal set of parallel links we keep only the cheapest one). Note that shadow completion does not affect the optimal solution value, since every shadow can be replaced by some original link covering all edges of T covered by the shadow. Hence we can assume the following.

Assumption 1. If $uv \in E$ and $u', v' \in T_{uv}$ then $u'v' \in E$ and $c(u'v') \leq c(uv)$.

In the next section we will prove the following statement.

Lemma 2. Under Assumption 1, Tree Augmentation admits a polynomial time algorithm that finds a 2-approximate solution F_0 that is $3^{h(T)-1}$ -overlapped by any feasible solution F^* .

Lemmas 1 and 2 easily imply the Theorem; note that the running time is bounded by

$$m^{3^{h(T)-1}} \text{poly}(n) \leq (n^2)^{3^{h(T)-1}} \text{poly}(n) \leq n^{3^{h(T)}} \text{poly}(n) ,$$

as claimed.

In the rest of this paper we prove Lemma 2.

2.3 Proof of Lemma 2

Root the tree at a center s of T (so $|T_{sv}| \leq h(T)$ for every $v \in V$). This defines an ancestor-descendant relation (partial order) on the nodes of T , where u is an ancestor of v (and v is a descendant of u) if $u \in T_{vs}$; if also $uv \in T$ then v is a child of u and u is the parent of v .

Definition 4. We say that a link is an up-link if one of its endnodes is an ancestor of the other. We say that a cover F of T is an up-cover of T if every link in F is an up-link.

The following statement is known, and was implicitly proved in [4]. For completeness of exposition, we provide a proof-sketch.

Lemma 3. Under Assumption 1, for any cover F^* of T there exists an up-cover F of T such that $c(F) \leq 2c(F^*)$. Furthermore, a minimum-cost up-cover can be computed in polynomial time. Consequently, there exists a shadow-minimal up-cover F_0 of cost at most 2opt , and such cover can be computed in polynomial time.

Proof. Let F be obtained from F^* by replacing every link $e = uv \in F$ by the two links ua, va , where a is the least common ancestor of u, v in T . By Assumption 1,

the links ua, va exist, and the cost of each of them is at most $c(uv)$. Hence $c(F) \leq 2c(F^*)$. It is easy to see that F is a feasible solution, which concludes the proof of the first statement of the lemma. The problem of computing a minimum-cost up-cover is reduced to the Minimum-Cost Arborescence problem (which is solvable in polynomial time, c.f. [11]) as follows. It is known and easy to see [4] that F is an up-cover of T if, and only if, the directed graph obtained by directing the edges of T towards the root s and directing the links in F from ancestors to descendants, has a path from s to every other node. Hence to compute a minimum-cost up-cover do the following. Direct the edges of T towards the root, remove all links that are not up-links, and direct all up-links from ancestors to descendants. Then in the obtained directed graph compute a minimum-cost arborescence. The set of links (that are not edges of T) in the underlying graph of the computed arborescence, is a minimum-cost up-cover of T .

The last statement of the lemma follows by observing that under Assumption 1 we can replace any up-cover F of T by a shadow-minimal up-cover F_0 of no greater cost. \square

Lemma 4. *Let uv, xy be up-links such that T_{uv} and T_{xy} have an edge in common but none of them is a subpath of the other. Then one of uv, xy has a proper shadow that together with the other link they cover all edges in $T_{uv} \cup T_{xy}$.*

Proof. As T_{uv} and T_{xy} intersect, either y is an ancestor of v , or v is an ancestor of y . Assume w.l.o.g. that y is an ancestor of v . Let z be the lowest (farthest from root) node that T_{xy}, T_{uv} have in common. Then uz is a proper shadow of uv , and $\{xy, uz\}$ cover all edges in $T_{uv} \cup T_{xy}$. As the link xy covers an edge on T_{uv} , one of x, y , say x , must be an internal node of the path T_{uv} . Since none of T_{uv}, T_{xy} is a subpath of the other, y must be either a proper ancestor of v or a proper descendant of u . In the former case vy is a proper shadow of xy and $\{uv, vy\}$ cover all edges in $T_{uv} \cup T_{xy}$. In the latter case yu is a proper shadow of xy and $\{uv, yu\}$ cover all edges in $T_{uv} \cup T_{xy}$. \square

From Lemma 4 we deduce the following.

Corollary 1. *Let F be a shadow-minimal up-cover of T . Then every edge of T is covered by a unique link in F .*

Definition 5. *The height $h(v)$ of a node $v \in V$ is the distance from v to the farthest descendant of v in T (note that by our choice of s , $h(s)$ is the radius $h(T)$ of T). For $u, v \in V$ let $\text{lca}(u, v)$ denote the least common ancestor of u and v in T . For a link $e = uv$ let $\text{lca}(e) = \text{lca}(u, v)$. The height $h(e)$ of a link $e = uv$ is $h(\text{lca}(e))$.*

In the rest of this section we will prove the following statement, that together with Lemma 3 and Corollary 1 implies Lemma 2.

Lemma 5 (The Decomposition Lemma). *Let F^* be a cover and F an up-cover of a tree T rooted at s , such that every edge of T is covered by a unique*

link in F . Then F is $3^{h(s)-1}$ -overlapped by F^* , namely, there exist a partition \mathcal{S} of F^* into parts of size at most $3^{h(s)-1}$ each, such that for every $f \in F$ there exists $S \in \mathcal{S}$ with $f \in R_F(S)$.

The bound $3^{h(s)-1}$ in Lemma 5 is tight, as is shown in the next section. Namely, for any integer $h \geq 1$, there exists a tree T rooted at s and F, F^* as in Lemma 5, such any partition of F^* that overlaps F has a part of size at least $3^{h(s)-1}$.

In the rest of this section we prove Lemma 5. Define an auxiliary directed graph $J = (V_J, E_J)$ as follows. For every $f \in F$, let u^f, v^f be the endnodes of f , where u^f is a descendant of v^f . Let $I^f \subseteq F^*$ be some inclusion minimal cover of the unique $u^f v^f$ -path P^f in T . Let $k(f) = |I^f|$ and let $e_1^f, e_2^f, \dots, e_{k(f)}^f$ be an ordering of I^f obtained as follows, see Figure 1. For $i = 1, \dots, k(f)$, e_i^f is the link in I^f that covers the lowest (farthest from the root) edge of P^f not covered by $\{e_1^f, \dots, e_{i-1}^f\}$. The node set of J is $V_J = F^*$ and the edge set of J is $E_J = \{e_{i+1}^f e_i^f : f \in F, 1 \leq i \leq k(f) - 1\}$. We will prove:

Lemma 6. J is a collection of node-disjoint arborescences with at most $3^{h(s)-1}$ nodes each.

Lemma 5 easily follows from Lemma 6. The desired partition \mathcal{S} of F^* is the one defined by the arborescences of the auxiliary graph J . Note that by the construction, for every $f \in F$ the ordering $e_1^f, e_2^f, \dots, e_{k(f)}^f$ of I^f forms a directed path in J . Hence for every $f \in F$, I^f belongs to the same part (arborescence), which defines the part $S \in \mathcal{S}$ such that $f \in R_F(S)$.

In the rest of this section we prove Lemma 6. The following fact stems from the definition of I^f (see Figure 1).

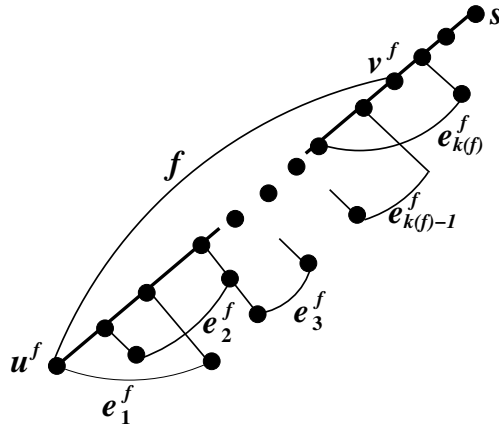


Fig. 1. Illustration to the definition of I^f and Fact 2

Fact 2. Let $f \in F$. Then for every $i = 1, \dots, k(f)$, the set of edges from P^f covered by the link-set $\{e_1^f, \dots, e_i^f\}$ is exactly the set of edges on the $u^f v_i^f$ -path in T , where $v_i^f = \text{lca}(e_i^f)$.

Thus from the minimality of I^f we have the following.

Corollary 2. Let $f \in F$ with $k(f) \geq 2$. Then for every $1 \leq i \leq k(f) - 1$, v_i^f is an inner node of P^f , $h(e_i^f) < h(e_{i+1}^f)$, and both f and e_{i+1}^f cover the parent edge of e_i^f (the edge between v_i^f and its parent) in T .

Lemma 7. J is acyclic and $\text{deg}_J^{in}(e) \leq 1$ for every $e \in F^*$. Thus J is a collection of node-disjoint arborescences.

Proof. From Corollary 2 it easily follows that J is acyclic. We will prove that $\text{deg}_J^{in}(e) \leq 1$ for every $e \in F^*$. Suppose to the contrary that there are two edges $e'e, e''e \in E_J$ entering e in J . By the definition of J , there are two distinct links $f', f'' \in F$ such that $e = e_{i'}^{f'} \in I_{f'}$ and $e = e_{i''}^{f''} \in I_{f''}$. By Corollary 2, e has a parent edge in T , and both f', f'' cover the parent edge of e in T . This contradicts the assumption that every edge of T is covered by a unique link in F . \square

Lemma 8. For every $e \in F^*$, no three neighbors of e in J have the same height.

Proof. Let e', e'' be two distinct neighbors of e in J with $h(e') = h(e'')$. By the definition of J , there exist distinct $f', f'' \in F$ such that $e' = e_{i'}^{f'}, e'' = e_{i''}^{f''}$, and $e = e_{i'+1}^{f'} = e_{i''+1}^{f''}$. By Corollary 2, f' covers the parent edge of e' and f'' covers the parent edge of e'' . Since f' and f'' are distinct and since every edge of T is covered by a unique link in F , the parent edges of e' and e'' are also distinct. Since e', e'' have the same height, the parent edges of e' and e'' also have the same height. By Fact 2, e covers the parent edge of each of e', e'' . Hence, e cannot have a third neighbor in J , since then e will cover three distinct edges in T of the same height, but no link can cover three distinct edges in T of the same height. \square

Corollary 3. For $e \in F^*$ let A_e be the set of nodes in J reachable from e . Then $|A_e| \leq 3^{h(e)-1}$. Thus every arborescence in J has at most $3^{h(s)-1}$ nodes.

Proof. We prove the statement by induction on $h(e)$. If $h(e) = 1$ then e has no neighbors in J , by Corollary 2. Hence $|A_e| = 1$ and $3^{h(e)-1} = 3^0 = 1$, and the statement is valid in this case. Suppose that $h(e) \geq 2$ and that any arborescence A' in J with root e' and $h(e') \leq h(e) - 1$ has at most $3^{h(e')-1}$ nodes.

Let e' be a neighbor of e in J . By Corollary 2, $h(e') \leq h(e) - 1$. Hence by the induction hypothesis, $|A_{e'}| \leq 3^{h(e')-1}$. By Lemma 8, no three distinct neighbors of e have the same height. Thus we get:

$$|A_e| \leq 1 + 2 \cdot \sum_{i=1}^{h(e)-1} 3^{i-1} = 1 + 2 \cdot \frac{1 - 3^{h(e)-1}}{1 - 3} = 1 + 3^{h(e)-1} - 1 = 3^{h(e)-1} .$$

\square

This finishes the proof of Lemma 6, and thus also the proof of the Theorem is now complete.

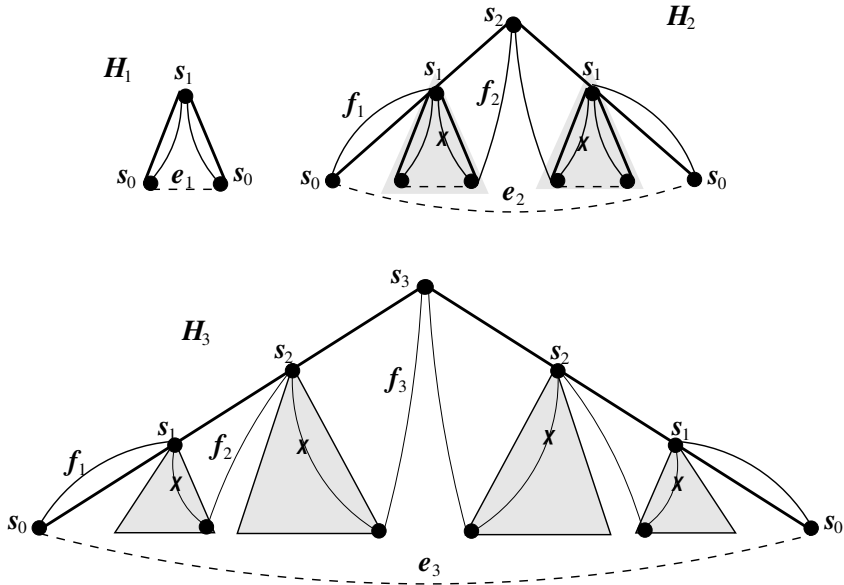


Fig. 2. Construction of a tight example for Lemma 5 with $h(s) = 1, 2, 3$. T -edges are shown by bold lines, F^* -edges are shown by dashed lines, and F -edges are shown by thin lines.

2.4 A Tight Example for the Decomposition Lemma

Here we show for any integer $h \geq 1$, there exists a tree T rooted at s and F, F^* as in Lemma 5, such any partition of F^* that overlaps F has a part of size at least $3^{h(s)-1}$.

In our example we will show that any partition \mathcal{S} of F^* that overlaps F must consist of one part $\mathcal{S} = \{F^*\}$. Define a sequence graphs H_i , each with spanning tree T_i rooted at s_i , and two edge subsets F_i, F_i^* as follows.

H_1 is depicted in Figure 2.

To obtain H_2 do the following (see Figure 2). Take a path $P_2 = s_2 - s_1 - s_0$ of T -edges, add the F -link $f_1 = s_0 s_1$, and attach a copy of H_1 via the root to s_1 . This copy has an F -link from a leaf to s_1 , and we replace the endnode s_1 of this link by s_2 , to obtain the link f_2 . Take another copy of the obtained graph and identify the node s_2 of both copies. Finally, add the F^* -link e_2 that connects the two copies of s_0 on copies of P_3 , as depicted in Figure 2. The edges of T_2 are the T_1 edges in the H_1 copies plus the edges in the P_2 copies. The F^* -links are the union of the F^* -links in the copies of H_1 and the link e_2 ; alternatively, these are the links in H_2 that connect two leaves of T_2 . The remaining links are the F -links, and these are the up-links in H_2 .

In general, the construction is recursive as follows. Given H_1, \dots, H_{i-1} , to obtain H_i do the following (see Figure 2 for the case $i = 3$). Take a path $P_i = s_i - s_{i-1} - \dots - s_0$ of T -edges, add the F -link $f_1 = s_0 s_1$, and for every $j = 1, \dots, i - 1$ attach a copy of H_j via the root to s_j . Each copy H_j has an F -link from a leaf to s_j , and we replace the endnode s_j of this link by s_{j+1} , to obtain the link f_{j+1} . Take another copy of the obtained graph and identify the node s_i of both copies. Finally, add the F^* -link e_i that connects the two copies of s_0 on copies of P_i , as depicted in Figure 2. The edges of T_i are the T_j -edges in the copies H_j plus the edges in the P_i copies. The F^* -links are the union of the F^* -links in the H_j -copies and the link e_i ; alternatively, these are the links in H_i that connect two leaves of T_i . The remaining links are the F -links, and these are the up-links in H_i .

It is not hard to verify that $|F_i^*| = 3^{i-1}$ and that $|F_i| = 2|F_i^*| = 2 \cdot 3^{i-1}$. Now let \mathcal{S} be a partition of F_i^* that overlaps F , and let $S \in \mathcal{S}$ be the part that contains e_i . We claim that $S = F^*$. For every $j = 2, \dots, i$, the tree edge $s_{j-1} s_j$ is covered by the link f_j , and e_i is the only link in F^* that covers this edge. This implies that we must have $f_j \in R_F(S)$. Using a similar argument, we can conclude that in any H_j -copy, the link e_j must be in S . Thus using induction we can show that in each H_j -copy, all the F^* links belong to S . Consequently, all the F^* -links in H_i belong to F^* , as claimed.

References

1. Cheriyan, J., Jordán, T., Ravi, R.: On 2-coverings and 2-packings of laminar families. In: Nešetřil, J. (ed.) *ESA 1999*. LNCS, vol. 1643, pp. 510–520. Springer, Heidelberg (1999)
2. Cheriyan, J., Karloff, H., Khandekar, R., Könemann, J.: On the integrality ratio for tree augmentation. *Operations Research Letters* 36(4), 399–401 (2008)
3. Even, G., Kortsarz, G., Nutov, Z.: A 1.5 approximation algorithm for augmenting edge-connectivity of a graph from 1 to 2. *Information Processing Letters* 111(6), 296–300 (2011)
4. Fredrickson, G.N., Jájá, J.: On the relationship between the biconnectivity augmentation and traveling salesman problem. *Theoretical Computer Science* 19(2), 189–201 (1982)
5. Goemans, M., Williamson, D.: The primal dual method for approximation algorithms and its applications to network design problems. In: Hochbaum, D.S. (ed.) *Approximation Algorithms for NP-hard Problems*, ch. 4, pp. 144–191. PWS (1995)
6. Jain, K.: A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica* 21(1), 39–60 (2001)
7. Khuller, S.: Approximation algorithms for finding highly connected subgraphs. In: Hochbaum, D.S. (ed.) *Approximation Algorithms for NP-hard problems*, ch. 6, pp. 236–265. PWS (1995)
8. Kortsarz, G., Nutov, Z.: Approximating minimum cost connectivity problems. In: Gonzales, T.F. (ed.) *Approximation Algorithms and Metaheuristics*, ch. 58. CRC, Boca Raton (2007)

9. Maduel, Y., Nutov, Z.: Covering a laminar family by leaf to leaf links. *Discrete Applied Mathematics* 158(13), 1424–1432 (2010)
10. Nagamochi, H.: An approximation for finding a smallest 2-edge connected subgraph containing a specified spanning tree. *Discrete Applied Mathematics* 126, 83–113 (2003)
11. Schrijver, A.: *Combinatorial Optimization, Polyhedra and Efficiency*. Springer, Heidelberg (2004)
12. Zelikovsky, A.: Better approximation bounds for the network and euclidean steiner tree problems. Technical report (1995)

Periodicity and Cyclic Shifts via Linear Sketches

Michael S. Crouch and Andrew McGregor*

Department of Computer Science, University of Massachusetts, Amherst, MA 01003

mcc@cs.umass.edu, mcgregor@cs.umass.edu

Abstract. We consider the problem of identifying periodic trends in data streams. We say a signal $\mathbf{a} \in \mathbb{R}^n$ is p -periodic if $a_i = a_{i+p}$ for all $i \in [n-p]$. Recently, Ergün et al. [4] presented a one-pass, $O(\text{polylog } n)$ -space algorithm for identifying the smallest period of a signal. Their algorithm required \mathbf{a} to be presented in the *time-series* model, i.e., a_i is the i th element in the stream. We present a more general linear sketch algorithm that has the advantages of being applicable to a) the *turnstile stream model*, where coordinates can be incremented/decremented in an arbitrary fashion and b) the *parallel* or *distributed* setting where the signal is distributed over multiple locations/machines. We also present sketches for $(1+\epsilon)$ approximating the ℓ_2 distance between \mathbf{a} and the nearest p -periodic signal for a given p . Our algorithm uses $O(\epsilon^{-2} \text{polylog } n)$ space, comparing favorably to an earlier time-series result that used $O(\epsilon^{-5.5} \sqrt{p} \text{polylog } n)$ space for estimating the Hamming distance to the nearest p -periodic signal. Our last periodicity result is an algorithm for estimating the periodicity of a sequence in the presence of noise. We conclude with a small-space algorithm for identifying when two signals are exact (or nearly) cyclic shifts of one another. Our algorithms are based on bilinear sketches [10] and combining Fourier transforms with stream processing techniques such as ℓ_p sampling and sketching [13, 11].

1 Introduction

We consider the problem of identifying periodic trends in data streams. Motivated by applications in computational biology and data mining, there has recently been a series of papers related to finding such trends in large data sets [4, 5, 9, 3, 16]. We say a signal $\mathbf{a} \in \mathbb{R}^n$ is p -periodic if it can be expressed as a concatenation $\mathbf{a} = \mathbf{x} \circ \dots \circ \mathbf{x} \circ \mathbf{x}'$ for some $\mathbf{x} \in \mathbb{R}^p$ and some $\mathbf{x}' \in \mathbb{R}^{n-p \lfloor n/p \rfloor}$ that is a prefix of \mathbf{x} . We say \mathbf{a} is *perfectly p -periodic* if \mathbf{a} is p -periodic and $p \mid n$. Given a signal $\mathbf{a} \in \mathbb{R}^n$, we define the distance to p -periodicity as

$$D_p(\mathbf{a}) \equiv \min_{\mathbf{y} \in P_{p,n}} \|\mathbf{a} - \mathbf{y}\|_2 \quad \text{where } P_{p,n} = \{\mathbf{y} \in \mathbb{R}^n : \mathbf{y} \text{ is } p\text{-periodic}\}$$

where $\|\mathbf{v}\|_2 = \sqrt{v_1^2 + \dots + v_n^2}$ denotes the ℓ_2 norm of the vector $\mathbf{v} \in \mathbb{R}^n$. (We will later discuss our choice of distance measure and observe that many of our results still hold if an alternative measure is chosen.) We denote the minimum period of a signal $\mathbf{a} \in \mathbb{R}^n$ by

$$\text{period}(\mathbf{a}) = \min\{p : \mathbf{a} \text{ is } p\text{-periodic}\}.$$

* Research supported by NSF Award CCF-0953754.

In this paper, we consider signals defined by a stream of data. Previous periodicity work assumes that the stream *is* the signal, e.g., the stream $\langle 1, 2, 3, 4 \rangle$ defines the signal $\mathbf{a} = [1, 2, 3, 4]$. However, we wish to consider a more general setting. For example, consider a sensor network in which each node is tasked with recording the times when certain local events occur. These records are forwarded through the network to some central node for processing. In this situation, there is no guarantee that the records are received in the order they were generated. Hence, we would need an algorithm that could identify patterns even if the records arrive out of order. A yet more challenging example would be if each sensor monitors the local temperature at each time step and we are interested in identifying periodic trends in the average temperature. In this case, not only can records arrive out of order but the signal will be determined by the value of multiple records.

Following the terminology of Muthukrishnan [14, pg. 12–13], we consider three different stream models in which the signal \mathbf{a} of interest can be defined. In the *time-series* model the stream $S = \langle a_0 \dots a_{n-1} \rangle$ defines the signal directly. More general is the *permutation* model where coordinates of \mathbf{a} may arrive out of order, i.e., $S = \langle (\pi(0), a_{\pi(0)}) \dots (\pi(n-1), a_{\pi(n-1)}) \rangle$, for some permutation π of $\{0, \dots, n-1\}$. Finally, in the *turnstile* model, \mathbf{a} is defined by a sequence of increments and decrements, i.e., for a stream

$$S = \langle (u_1, \Delta_1), \dots, (u_m, \Delta_m) \rangle \quad \text{where } u_i \in \{0, \dots, n-1\}, \Delta_i \in \mathbb{R}$$

we define \mathbf{a} by $a_j = \sum_{i:u_i=j} \Delta_i$. All of our algorithms work in the turnstile model and are *sketch-based*. We will discuss sketches in more detail in Sect. 2 but note here that one of their main advantages is that they work in a distributed setting where parts of the streams are monitored at different locations: after the stream concludes, it is sufficient to communicate only the sketches, as these can then be merged in order to estimate the global property of interest. This would enable data aggregation in the sensor network example outlined above.

1.1 Our Results and Related Work

Our first result is an $O(\epsilon^{-2} \text{polylog } n)$ space algorithm that $(1 + \epsilon)$ -approximates $D_p(\mathbf{a})$ for any given p (where p need not divide the length of the sequence). In contrast, an earlier paper by Ergün et al. [4] presented an algorithm using $O(\epsilon^{-5.5} \sqrt{p} \text{polylog } n)$ space for estimating the Hamming distance to the nearest p -periodic signal. They also present a single-pass, $O(\text{polylog } n)$ -space algorithm for computing $\text{period}(\mathbf{a})$ in the time-series model. Our second result generalizes this result to the turnstile model although our algorithm in this case requires that \mathbf{a} is perfectly periodic.

Next we examine estimating the periodicity of a sequence in the presence of noise. While a seemingly natural problem, defining the precise problem is subtle. For example, should we deem the noisy signal

$$\mathbf{a} = [1, 2, 3, 1, 2, 3.5, 1, 2, 3.1, 1, 2, 3.4] \tag{1}$$

to be 3-periodic, 6-periodic, or aperiodic? Our algorithm achieves a natural “gap promise” guarantee: given φ, ϵ with $0 < \varphi < \epsilon < 1$, it returns a period $p \mid n$ with

$$D_p(\mathbf{a}) \leq \epsilon \|\mathbf{a}\|_2 \quad \text{and} \quad p \leq \min\{q \mid n : D_q(\mathbf{a}) \leq (\epsilon - \varphi) \|\mathbf{a}\|_2\} .$$

(Note that there is always such a p , since any length- n signal trivially has $D_n(\mathbf{a}) = 0$.) In other words, we ensure that \mathbf{a} is close to being perfectly p -periodic and that there is no $q \leq p$ such that \mathbf{a} is “significantly closer” to being perfectly q -periodic. This algorithm operates in the general turnstile model and uses $\text{poly}(\log n, \varphi^{-1})$ space. The algorithm is based on sampling in the Fourier domain and was actually inspired by Shor’s algorithm for quantum factorization [17]. There is no analog in the recent Ergün et al. [4] paper but an earlier result [5] in the combinatorial property-testing model can be applied in the streaming setting if we may use $O(\sqrt{n} \text{polylog } n)$ space.

We conclude with a simple sketch algorithm for the related problem of identifying when two sequences are cyclic shifts of one another. This algorithm uses $O(\epsilon^{-2} \sqrt{n} \text{polylog } n)$ space and has the additional feature that it actually approximates how close the strings are to being cyclic shifts.

Notation. We write $[n] = \{0, 1, 2, \dots, n - 1\}$. We denote signals in lower-case bold and their corresponding Fourier transforms in upper-case bold. For a complex number $z \in \mathbb{C}$ we denote the real and imaginary parts by $\text{Re}(z)$ and $\text{Im}(z)$ respectively. For functions $f(n), g(n)$, we write $f(n) = \tilde{O}(g(n))$ when there is a constant k such that $f(n) = O(g(n) \log^k n)$. $I[\varphi]$ is the 0-1 indicator function which is 1 whenever φ is true.

Precision. Throughout, we will assume that the values of the signals can be exactly stored with $1/\text{poly}(n)$ precision. For example, this would be guaranteed in the turnstile model with a number of updates $m = \text{poly}(n)$ and with each $\Delta_j \in \{-M, -M + 1, \dots, M - 1, M\}$ for some $M = \text{poly}(n)$. We also assume that the approximation parameters $\epsilon, \varphi, \delta$ satisfy $1/\epsilon, 1/\delta, 1/\varphi \in O(\text{poly } n)$.

2 Fourier Preliminaries and Choice of Distance Function

In this section, we review the basic definition and properties of the discrete Fourier transform. We then discuss the utility of the transform in the context of sketch-based data stream algorithms.

2.1 Discrete Fourier Transform and Sketches

Given a signal $\mathbf{a} \in \mathbb{R}^n$, the discrete Fourier transform of \mathbf{a} , denoted $\mathbf{A} = \mathcal{F}(\mathbf{a})$, is defined as

$$\mathbf{A} = (A_0, A_1, \dots, A_{n-1}) \quad \text{where} \quad A_k = \frac{1}{\sqrt{n}} \sum_{j=0}^{n-1} a_j e^{\frac{2\pi i}{n} jk} .$$

The following proposition states some standard properties that will be of use.

Proposition 1. For any signal $\mathbf{a} \in \mathbb{R}^n$,

1. \mathbf{a} is perfectly p -periodic iff $(A_k \neq 0 \Rightarrow n/p \mid k)$.
2. $\|\mathbf{a}\|_2 = \|\mathbf{A}\|_2$ (Parseval's identity).

Of particular importance in the context of data streams is the fact that the transformation from \mathbf{a} to \mathbf{A} is a linear transformation, i.e.,

$$\mathbf{A}^T = V\mathbf{a}^T \text{ where } V \in \mathbb{C}^{n \times n} \text{ and } V_{kj} = \frac{1}{\sqrt{n}} e^{\frac{2\pi i}{n}kj} \text{ for } k, j \in [n]. \quad (2)$$

This is significant because many data stream algorithms are based on randomized linear projections called *sketches*. Suppose we are interested in a function f of $\mathbf{x} \in \mathbb{R}^n$ where each coordinate x_j is determined by the turnstile stream $S = \langle (u_1, \Delta_1), \dots, (u_m, \Delta_m) \rangle$ according to $x_j = \sum_{i:u_i=j} \Delta_i$. A sketching algorithm chooses a random linear map $W \in \mathbb{R}^{k \times n}$ such that $W\mathbf{x}^T$ can be post-processed to yield an estimate of $f(\mathbf{x})$ (with certain error and probability guarantees). The algorithm computes $W\mathbf{x}^T$ incrementally using space proportional to k rather than n :

$$W\mathbf{x}^T = (\dots(((We^{u_1}) + We^{u_2}) + We^{u_3}) + \dots) + We^{u_m}$$

where $\mathbf{e}^{u_i} = (0, \dots, 0, \Delta_i, 0, \dots, 0)^T$ has the non-zero entry in the u_i -th position. For many functions, such as quantiles and heavy hitters [2], distinct items [12], and ℓ_1 and ℓ_2 norms [8], such sketches exist where k is only polylogarithmic in n . Of course, it would still defeat the object of small-space computation if the algorithm needed to explicitly store a random $k \times n$ matrix. Instead the random matrices of interest are constructed either using limited independence or via a pseudo-random generator, e.g., Nisan [15]. Either way, the relevant entries can be reconstructed from some small seed as required.

We will make use of the simple, but very useful, observation that rather than estimating functions in the time domain, we may estimate these functions in the frequency domain by combining the change of basis matrix V with the sketch matrix W . For example, if the random sketch matrix $W \in \mathbb{R}^{k \times n}$ can be used to estimate the number of non-zero entries in \mathbf{a} then the sketch matrix $WV \in \mathbb{C}^{k \times n}$ can be used to estimate the number of non-zero entries [4] in \mathbf{A} .

2.2 Choice of Distance Function

In the context of the Fourier transform and many signal processing applications, the natural measure of dissimilarity between two signals is the ℓ_2 norm of their difference. In contrast, Ergün and coauthors [4,5] considered a measure based on

¹ To be precise, it is often necessary to separate real and imaginary parts of V . That is, we consider $W \in \mathbb{R}^{k \times 2n}$ and let $V \in \mathbb{R}^{2n \times n}$ have entries $V_{kj} = \cos(2\pi jk/n)$ for $k \in \{0, \dots, n-1\}$ and $V_{kj} = \sin(2\pi jk/n)$ for $k \in \{n, \dots, 2n-1\}$. In calculating the ℓ_2 norm this causes no difficulties, but in other cases we may need to be careful. If we counted the number of nonzero entries of V , for example, we would find the total number of non-zero real parts and non-zero imaginary parts.

the Hamming distance, $D_p^0(\mathbf{a}) \equiv \min_{\mathbf{y} \in P_{p,n}} \Delta(\mathbf{a}, \mathbf{y})$ where $\Delta(\mathbf{a}, \mathbf{y}) = |\{i \in [n] : a_i \neq y_i\}|$. While different measures are suited to different applications, many of our algorithms can also be applied to approximate the Hamming distance, at least in the permutation model.

Suppose $\Sigma = \{\sigma_1, \dots, \sigma_r\}$ and consider the mapping from $\Sigma \rightarrow \{0, 1\}^r$:

$$h(\sigma) = x_1 \dots x_r \quad \text{where } x_j = \begin{cases} 1 & \text{if } \sigma = \sigma_j \\ 0 & \text{otherwise} \end{cases} .$$

The following lemma demonstrates that $D_p^0(\mathbf{a})$ and $(D_p(h(\mathbf{a})))^2/2$ are closely related. Hence, if each element of the sequence is first transformed using h (as is possible in the permutation model) then the Hamming distance to periodicity can be approximated via the ℓ_2 distance to periodicity. The approximation is by a factor close to 1 if the sequence is close to being p -periodic. Note that we would expect this to be the more relevant case in the sense that we would be measuring the distance from periodicity of a nearly-periodic sequence.

Lemma 1. *For any $\mathbf{a} \in \Sigma^n$, with $\Sigma = \{\sigma_1, \dots, \sigma_r\}$, let $T(\mathbf{a}) = (D_p(h(\mathbf{a})))^2/2$. Then we have,*

$$\frac{1}{2} D_p^0(\mathbf{a}) \leq T(\mathbf{a}) \leq D_p^0(\mathbf{a}) . \tag{3}$$

Furthermore, if \mathbf{a} is almost periodic in the sense that at least a $1 - \epsilon$ fraction of the elements $\{a_j, a_{j+p}, \dots, a_{j+n-p}\}$ are identical for each $j \in [p]$, then $(1 - \epsilon) D_p^0(\mathbf{a}) \leq T(\mathbf{a}) \leq D_p^0(\mathbf{a})$.

We can also relate $D_p(\mathbf{a})$ to the ℓ_1 distance to the nearest p -periodic signal. For this, consider the alphabet $\Sigma = \{1, \dots, t\}$, and use the mapping $h(\sigma) = x_1 \dots x_r$ where $x_j = I[\sigma \geq j]$.

3 Periodicity

3.1 Distance from Fixed Periodicity

We first present a fast algorithm for measuring the distance between the signal and the closest (under the ℓ_2 norm) p -periodic sequence, for fixed p . In this section, we emphasize that we do not require that the length of sequence is a perfect multiple of the periods considered. For $p < n$, we write $n = dp + r$ where

$$d = \lfloor n/p \rfloor \quad \text{and} \quad r = n \bmod p .$$

Basic properties of the ℓ_2 norm imply that the p -periodic pattern that is ℓ_2 -closest to a vector \mathbf{a} is the arithmetic mean of length- p segments of the vector:

Lemma 2. *For any sequence $\mathbf{a} \in \mathbb{R}^n$, let $\mathbf{c} = \operatorname{argmin}_{\mathbf{y} \in P_{n,p}} \|\mathbf{a} - \mathbf{y}\|_2$ be the p -periodic vector which is ℓ_2 -closest to \mathbf{a} . Then $\mathbf{c} = \mathbf{b} \circ \dots \circ \mathbf{b} \circ (b_0 b_1 \dots b_{r-1})$ where*

$$b_i = \begin{cases} \sum_{j=0}^d a_{i+jp} / (d+1) & \text{for } 0 \leq i < r \\ \sum_{j=0}^{d-1} a_{i+jp} / d & \text{for } r \leq i \leq p-1 \end{cases} .$$

With this explicit form for \mathbf{c} , there is a natural algorithm using Tug-of-War sketches [11] to approximate $D_p(\mathbf{a}) = \|\mathbf{a} - \mathbf{c}\|_2$. Alon et al. showed that if the entries of a random vector $\mathbf{z} = z_0 \dots z_{n-1} \in \{-1, 1\}^n$ are chosen with 4-wise independence then the random variable $T = \sum_{i=0}^{n-1} z_i(a_i - c_i)$ satisfies $E[T^2] = \|\mathbf{a} - \mathbf{c}\|_2^2$. They show that the estimator has sufficiently low variance that, by averaging $O(\epsilon^{-2} \log \delta^{-1})$ independent estimators, we can find a $(1 + \epsilon)$ approximation for $\|\mathbf{a} - \mathbf{c}\|_2^2$. The value of T can easily be constructed in a streaming fashion: when the i th element of \mathbf{a} is incremented by Δ we increment

$$T += \left(z_i - \sum_{j:i=j \bmod p} \frac{z_j}{|\{j : 0 \leq j \leq n-1, i = j \bmod p\}|} \right) \Delta$$

A naive implementation of this update method takes $\Omega(n/p)$ time per update. To avoid this we adapt the bilinear sketch method of Indyk and McGregor [10]. This technique was originally designed to detect correlations in data streams but we can exploit the structure of this sketch to reduce the update time. Rather than view \mathbf{a} as a length n vector, we encode it as a $(d + 1) \times p$ matrix A where $A_{ij} = a_{ip+j}$ if $ip + j \leq n - 1$ and $A_{ij} = b_j$ otherwise. Similarly let C be the $(d + 1) \times p$ matrix where $C_{ij} = b_j$. E.g., for $n = 10$ and $p = 4$ we have the matrices

$$A = \begin{pmatrix} a_0 & a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 & a_7 \\ a_8 & a_9 & b_2 & b_3 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} b_0 & b_1 & b_2 & b_3 \\ b_0 & b_1 & b_2 & b_3 \\ b_0 & b_1 & b_2 & b_3 \end{pmatrix}.$$

Let $\mathbf{x} \in \{-1, 1\}^p$ and $\mathbf{y} \in \{-1, 1\}^{d+1}$ be random vectors whose entries are 4-wise independent. Indyk and McGregor extended the Alon et al. result to show that the outer product of \mathbf{x} and \mathbf{y} had sufficient randomness for a result similar to the Tug-of-War sketch. In our context, the result implies that if $T = \sum_{0 \leq i \leq d, 0 \leq j \leq p-1} x_j y_i (A_{ij} - C_{ij})$, then by appealing to Lemma 2, we have that

$$E[T^2] = \sum_{0 \leq i \leq d, 0 \leq j < p} (A_{ij} - C_{ij})^2 = D_p^2(\mathbf{a})$$

and there is still sufficiently low variance for $O(\epsilon^{-2} \log \delta^{-1})$ parallel repetitions to be sufficient for constructing a $(1 + \epsilon)$ approximation with probability $1 - \delta$. We next show that each T can be constructed in only $O(1)$ update time. To do this, decompose T as

$$T = \sum_{\substack{0 \leq i \leq d \\ 0 \leq j < p}} x_j y_i A_{ij} - \sum_{\substack{0 \leq i \leq d \\ 0 \leq j < p}} x_j y_i C_{ij} = \sum_{\substack{0 \leq i \leq d \\ 0 \leq j < p}} x_j y_i A_{ij} - \left(\sum_{0 \leq i \leq d} y_i \right) \left(\sum_{0 \leq j < p} x_j b_j \right)$$

and define $T_1 = \sum_{0 \leq i \leq d, 0 \leq j < p} x_j y_i A_{ij}$ and $T_2 = \sum_{0 \leq j < p} x_j b_j$. Since $\sum_{0 \leq i \leq d} y_i$ can be computed in pre-processing, it suffices to compute T_1 and T_2 . We initialize $T_1 = T_2 = 0$. As the stream is read T_1 and T_2 are updated in $O(1)$ time using the following rule: when the $(ip + j)$ th entry of \mathbf{a} is incremented by Δ ,

$$T_1 += \left(x_j y_i + I[j \geq r] \frac{x_j y_d}{d} \right) \Delta \quad \text{and} \quad T_2 += \left(I[j < r] \frac{x_j}{d+1} + I[j \geq r] \frac{x_j}{d} \right) \Delta$$

where $r = n \bmod p$ and I is the indicator function.

Theorem 1. $D_p(\mathbf{a})$ can be approximated up to a factor $(1 + \epsilon)$ with probability $1 - \delta$ using $\tilde{O}(\epsilon^{-2})$ space and $\tilde{O}(\epsilon^{-2})$ update time. The algorithm operates in the turnstile model using one pass.

3.2 Determining Perfect Periodicity: Noiseless Case

In this and the next section we consider *finding* the period of a sequence that is perfectly periodic, i.e., we now assume that period divides the length. In this case, a possible approach to detecting periodicity with unknown period would be to use the above algorithm to test all factors $p \mid n$ and return the minimum p such that $D_p(\mathbf{a}) = 0$ (it suffices to set $\epsilon = 1$ for this purpose). Unfortunately, in the worst case n may have $d(n) = O(\exp(\log n / \log \log n))$ factors [7, pp. 260–264] and therefore this approach would take too much time and space. However, a simple modification suffices: we check for periodicity at each prime or power-of-a-prime factor k of n . Define the set

$$K(n) = \{k : k \text{ divides } n \text{ and is the power of a prime}\} .$$

We first observe that $|K(n)| \leq O(\log n)$ (since each prime factor of n is at least 2, we have from the prime factorization $n = p_1^{r_1} p_2^{r_2} \dots p_t^{r_t}$ that $|K(n)| = \sum r_i \leq \log_2 n$). The following lemma (see the appendix for the proof) demonstrates that testing periodicity for $p \in K(n)$ is sufficient to determine $\text{period}(\mathbf{a})$.

Lemma 3. For any $\mathbf{a} \in \mathbb{R}^n$ which is perfectly periodic,

$$\text{period}(\mathbf{a}) = \text{GCD}(n/k : k \in K(n) \text{ and } \mathbf{a} \text{ is } n/k\text{-periodic}) .$$

We can thus detect the minimum p for which \mathbf{a} is perfectly p -periodic by running $|K| = O(\log n)$ parallel copies of the algorithm from Section 3.1. With $O(\log n)$ points of failure, we must ensure that each algorithm fails with probability at most $\delta / \log n$; this increases the space by a $\log \log n$ factor which is dominated by other factors in the analysis.

Theorem 2. There is a single-pass, turnstile algorithm for computing $\text{period}(\mathbf{a})$ of perfectly periodic strings that uses $O(\text{polylog } n)$ space and update time.

3.3 Determining Perfect Periodicity: Noisy Case

In this section, we present an algorithm for estimating the periodicity of a noisy signal. As a stepping stone to this result, we discuss an alternative approach for the noiseless case based on sampling. An advantage of the alternative approach is that it does not require the factorization of n to be computed thereby avoiding any (admittedly sublinear time) preprocessing. However, the guarantee achieved is weaker.

Fourier Sampling. If \mathbf{a} is perfectly periodic with period p , then the Fourier transform $\mathbf{A} = \mathcal{F}(\mathbf{a})$ has at most p nonzero components. Letting $d = n/p$, we know by Prop. [1](#) that the only non-zero coordinates of \mathbf{A} are A_{kd} for $k \in \{0, \dots, p-1\}$. For the case of general \mathbf{a} , let \mathbf{X}_p denote the restriction of \mathbf{A} to the coordinates corresponding to a perfectly p -periodic signal, i.e.,

$$\mathbf{X}_p = (A_0, 0, \dots, 0, A_d, 0, \dots, 0, \dots, A_{(p-1)d}, 0, \dots, 0) .$$

In the frequency domain, \mathbf{X}_p is the closest Fourier transform of a period- p vector to \mathbf{A} . By Plancherel’s theorem, \mathcal{F} and \mathcal{F}^{-1} preserve inner products and ℓ_2 distances. Therefore, $\mathcal{F}^{-1}(\mathbf{X}_p)$ is the p -periodic vector that is closest to \mathbf{a} in the ℓ_2 distance. This implies that

$$D_p(\mathbf{a}) = \|\mathbf{a} - \mathcal{F}^{-1}(\mathbf{X}_p)\|_2 = \|\mathbf{A} - \mathbf{X}_p\|_2 = \|\mathbf{Y}_p\|_2 = \sqrt{\sum_{d \nmid k} |A_k|^2} . \quad (4)$$

Our algorithms in this section are based on combining the above relationship with a technique for sampling in the Fourier domain.

Recently, Monemizadeh and Woodruff [\[13\]](#) presented a general approach for ℓ_p -sampling in the time-domain: for a signal $\mathbf{a} \in \mathbb{R}^n$ defined in the turnstile model, the goal here is to output k with probability in the interval

$$\left[(1 - \alpha) \frac{|a_k|^p}{\ell_p^p(\mathbf{a})}, (1 + \alpha) \frac{|a_k|^p}{\ell_p^p(\mathbf{a})} \right]$$

for some small user-defined parameter $\alpha > 0$. They show that this can be performed efficiently in space $\text{poly}(\alpha^{-1} \log n)$.[2](#)

For our purposes, rather than considering the time-series vector \mathbf{a} , we consider the vector

$$\mathbf{A}' = (\text{Re}(A_1), \dots, \text{Re}(A_n), \text{Im}(A_1), \dots, \text{Im}(A_n)) \in \mathbb{R}^{2n} .$$

defined by applying the appropriate Fourier transform matrix to the signal. If ℓ_2 -sampling is performed on \mathbf{A}' and we return the value modulo n , then the probability that k is returned is in the interval:

$$\left[(1 - \alpha) \frac{|A_k|^2}{\|\mathbf{A}\|_2^2}, (1 + \alpha) \frac{|A_k|^2}{\|\mathbf{A}\|_2^2} \right] , \quad (5)$$

because $\frac{\text{Re}(A_k)^2 + \text{Im}(A_k)^2}{\sum_{i \in [n]} \text{Re}(A_j)^2 + \text{Im}(A_j)^2} = \frac{|A_k|^2}{\|\mathbf{A}\|_2^2}$.

² There is an additive error probability of n^{-C} for arbitrarily large constant C but this can be ignored in our subsequent analysis.

To perform this sampling we follow the approach suggested in Sect. 2. Specifically we use the fact that Monemizadeh and Woodruff’s ℓ_p -sampling algorithm can be performed using a sketch matrix W and that there exists a matrix transformation $V \in \mathbb{R}^{2n \times n}$ that transforms any signal $\mathbf{a} \in \mathbb{R}^n$ into the corresponding \mathbf{A}' vector. Hence, applying the sketch matrix WV allows us to sample from \mathbf{A}' as required. We will show how to use this sampling to the next two sections.

Application to the Noiseless Case. Suppose there is no noise and that $p = \text{period}(\mathbf{a})$. Let the samples collected be $k_1, \dots, k_w \in [n]$. We know from Prop. 1 that each sample $k_i = cd$ for some $c \in [p]$. Let $q = n/\text{GCD}(k_1, \dots, k_w, n)$. We have $q = p/c'$ for some $c' \mid p$. Next we will show that for sufficiently large w , with high probability, either $q = p$ or the sequence was nearly perfectly q -periodic. (For example, in the case of the sequence in Eq. (1), perhaps we return $q = 6$.)

Choose an approximation parameter $\varphi > 0$. Assume for contradiction that $q = p/c'$ for some $c' > 1$, but that $D_q(\mathbf{a}) \geq \varphi\sqrt{1 + \alpha}\|\mathbf{a}\|_2$. Summing over bins j , by appealing to Eq. (4), we have that

$$\sum_{n/q \nmid j} \frac{|A_j|^2}{\|\mathbf{A}\|_2^2} = \frac{1}{\|\mathbf{a}\|_2^2} \sum_{n/q \nmid j} |A_j|^2 = \frac{(D_q(\mathbf{a}))^2}{\|\mathbf{a}\|_2^2} \geq \varphi^2(1 + \alpha).$$

Therefore, using the $(1 + \alpha)$ approximation to ℓ_2 -sampling, the probability that we return a sample that is not a multiple of n/q is at least φ^2 . Taking $w = O(\varphi^{-2} \log(\delta^{-1} \log p))$ samples ensures that we find some sample that is not a multiple of n/q for all $O(\log p)$ prime factors q of p . Consequently, if the algorithm does not return the exact value of $\text{period}(\mathbf{a})$, it returns a value $h \mid \text{period}(\mathbf{a})$ such that the sequence was very close to being h -periodic with high probability.

Application to the Noisy Case. For noisy signals, a natural question is to find the smallest period p such that $D_p(\mathbf{a}) \leq \epsilon\|\mathbf{a}\|_2$. Unfortunately, since $D_p(\mathbf{a})$ could be just under $\epsilon\|\mathbf{a}\|_2$ while another value $q < p$ may have $D_q(\mathbf{a})$ just larger than $\epsilon\|\mathbf{a}\|_2$, this is too much to hope for. Instead we consider two parameters ϵ, φ with $\epsilon > \varphi > 0$, and use a slight modification of the above approaches to

³ The reader might also observe that the technique of sketching in the Fourier domain gives an alternative approach to estimating the distance to perfect periodicity using any sketch-based algorithm that returns a $(1 + \epsilon)$ approximation for ℓ_2 , e.g., [8, 11]. For example, consider the Tug-of-War sketch matrix $W \in \{-1, 1\}^{t \times 2n}$ used by Alon et al. [1] for ℓ_2 estimation, and the matrix

$$U \in \mathbb{R}^{2n \times 2n} \text{ where } U_{kj} = \begin{cases} 1 & \text{for } j = k \text{ and } d \mid j \\ 0 & \text{otherwise} \end{cases}.$$

By appealing to (4), $\|UV\mathbf{a}\|_2^2 = (D_p(\mathbf{a}))^2$. Then, following the analysis of [1] for W , we find $E[(WUV\mathbf{a})^2] = D_p(\mathbf{a})^2$ if W is chosen according to the appropriate distribution. Furthermore, the variance is sufficiently low such that a $(1 + \epsilon)$ approximation can be constructed with probability $1 - \delta$, it suffices to set $t = O(\epsilon^{-2} \log \delta^{-1})$. This leads to a one-pass algorithm using $O(\epsilon^{-2} \log \delta^{-1} \text{polylog } n)$ space.

accept some $p \mid n$ such that $D_p(\mathbf{a}) \leq \epsilon \|\mathbf{a}\|_2$, and for no smaller q do we have $D_q(\mathbf{a}) \leq (\epsilon - \varphi) \|\mathbf{a}\|_2$.

Our algorithm proceeds by taking samples of the Fourier coefficients as before. It then returns the smallest value $p \mid n$ such that at least $1 - (\epsilon - \varphi/2)$ fraction of the samples are of Fourier coefficients $k = cn/p$. With probability at least $1 - \delta$, we can guarantee that this condition is satisfied for all p with $D_p(\mathbf{a}) \leq (\epsilon - \varphi) \|\mathbf{a}\|_2$, and by no p with $D_p(\mathbf{a}) > \epsilon \|\mathbf{a}\|_2$; this requires $O(\varphi^{-2} \log \delta^{-1})$ samples by an application of the Chernoff bounds.

Theorem 3. *For any $\epsilon, \varphi, \delta$, there exists a single-pass, $O(\text{poly}(\log n, \varphi^{-1}))$ -space turnstile algorithm which returns $p \mid n$ such that both of the following conditions are satisfied with high probability:*

1. $D_p(\mathbf{a}) < \epsilon \|\mathbf{a}\|_2$
2. There does not exist $q < p$ such that $q \mid n$ and $D_q(\mathbf{a}) < (\epsilon - \varphi) \|\mathbf{a}\|_2$.

4 Cyclic Shifts

In this section, we consider the problem of identifying whether two sequences $\mathbf{a}, \mathbf{b} \in \Sigma^n$ are close to being cyclic shifts of each other. We will assume for convenience that $\Sigma \subset \mathbb{R}$. Let $\text{CS}_s : \mathbb{R}^n \rightarrow \mathbb{R}^n$ be the function that “rotates” the input sequence by s positions, i.e.,

$$\text{CS}_s(a_1 a_2 \dots a_n) = a_{s+1} a_{s+2} \dots a_n a_1 \dots a_s .$$

Then \mathbf{a} and \mathbf{b} are cyclic shifts iff there exists s such that $\mathbf{b} = \text{CS}_s(\mathbf{a})$.

Our goal is to recognize cyclic shifts using linear sketches. We first note that the analogous problem in the simultaneous communication model is rather straightforward. Suppose Alice knows $\mathbf{a} \in \Sigma^n$ and Bob knows $\mathbf{b} \in \Sigma^n$. They can easily determine whether $\text{CS}_s(\mathbf{a}) = \mathbf{b}$ for some s by each transforming \mathbf{a} and \mathbf{b} into some canonical form and then using an equality test. Specifically, consider an arbitrary ordering of the sequences in Σ^n . Alice generates the cyclic shift $\hat{\mathbf{a}}$ of \mathbf{a} that is minimal under this ordering. Similarly, Bob generates the minimal cyclic shift $\hat{\mathbf{b}}$ of \mathbf{b} . Clearly $\hat{\mathbf{a}} = \hat{\mathbf{b}}$ iff \mathbf{a} is a cyclic shift of \mathbf{b} . This can be verified with $O(\log n)$ communication using standard fingerprinting techniques.

Obviously such an approach is not possible in the data stream model. In the time-series model, existing work combined with simple observations leads to an efficient algorithm for determining if two sequences are cyclic shifts. We first review this before presenting a new streaming algorithm that is sketch-based and thus applies in the turnstile steaming model. Furthermore, it can estimate the distance of two sequences from being cyclic shifts.

Time-Series Model. In the time-series model, a one-pass $O(\text{polylog } n)$ -space algorithm follows from Ergün et al.’s extensions [4] of the pattern matching algorithm of Porat and Porat [16]. The algorithm works when one of the strings precedes

the other, i.e., $S = \langle a_0, a_1, \dots, a_{n-1}, b_0, b_1, \dots, b_{n-1} \rangle$, or when the strings are interleaved, i.e., $S = \langle a_0, b_0, a_1, b_1, \dots, a_{n-1}, b_{n-1} \rangle$. (It is actually sufficient for the elements of one sequence to always precede the corresponding elements of the other; e.g., the stream $S = \langle a_0, b_0, a_1, a_2, b_1, a_3, b_2, b_3 \rangle$ is acceptable.)

The pattern-matching algorithm of [4] uses a fingerprinting function Φ to maintain a series of exponentially-lengthening fingerprints $\varphi_j = \Phi(a_0 \dots a_{2^j-1})$; by cleverly updating appropriate fingerprints of \mathbf{b} , they keep track of each match for φ_j which occurred within the last 2^j characters. When we reach the final character of \mathbf{b} , for each m such that $\Phi(b_m \dots b_{m+2^j-1}) = \Phi(a_0 \dots a_{2^j-1})$, we have access to the fingerprints $\Phi(b_0 \dots b_{m-1})$, $\Phi(b_m \dots b_{m+2^j-1})$, and $\Phi(b_{m+2^j} \dots b_{n-1})$. By adjusting the fingerprints appropriately, we can determine whether there exists $m \in [n]$ such that

$$\Phi(a_0 \dots a_{n-1}) = \Phi(b_m \dots b_{m+2^j-1} b_{m+2^j} \dots b_{n-1} b_0 \dots b_{m-1}) .$$

4.1 Cyclic Shift Distance

In this section, we present a simple turnstile algorithm for estimating how close two sequences are to being cyclic shifts. We define the cyclic shift distance, CSD, between two strings as

$$\text{CSD}(\mathbf{a}, \mathbf{b}) = \min_s \|\mathbf{a} - \text{CS}_s(\mathbf{b})\|_2 .$$

Clearly, if \mathbf{b} is a cyclic shift of \mathbf{a} then $\text{CSD}(\mathbf{a}, \mathbf{b}) = 0$.

The algorithm proceeds as follows: assume for simplicity that n is a perfect square. We will use two sets of candidate shifts, $S = \{0, 1, 2, \dots, \sqrt{n} - 1\}$ and $T = \{\sqrt{n}, 2\sqrt{n}, 3\sqrt{n}, \dots, n\}$. As we process the turnstile stream, we construct Tug-of-War sketches [1] of $\text{CS}_s(\mathbf{a})$ and $\text{CS}_t(\mathbf{b})$ for each $s \in S, t \in T$. Using $O(\epsilon^{-2} \log \frac{1}{\delta} \log n)$ -sized sketches, this allows us to $(1 + \epsilon)$ -approximate $\|\text{CS}_s(\mathbf{a}) - \text{CS}_t(\mathbf{b})\|_2$ for each $s \in S$ and $t \in T$ with probability at least $1 - \delta'$. Since for all r, s we have that $\mathbf{a} - \text{CS}_s(\mathbf{b}) = \text{CS}_r(\mathbf{a}) - \text{CS}_{r+s}(\mathbf{b})$, these shifts suffice to $(1 + \epsilon)$ -approximate $\|\mathbf{a} - \text{CS}_u(\mathbf{b})\|_2$ for each $u \in \{1, \dots, n\}$.

Choosing $\delta' = \frac{\delta}{n}$, we have that each pair r, s is simultaneously a $(1 + \epsilon)$ -approximation with probability $\geq 1 - \delta$. We then find:

$$\Pr \left[\left| \min_{s \in S, t \in T} \|\text{CS}_s(\mathbf{a}) - \text{CS}_t(\mathbf{b})\|_2 - \text{CSD}(\mathbf{a}, \mathbf{b}) \right| \geq \epsilon \text{CSD}(\mathbf{a}, \mathbf{b}) \right] \leq \delta . \quad (6)$$

Theorem 4. *There exists a single pass algorithm using space $\tilde{O}(\epsilon^{-2} \sqrt{n})$ that returns a $(1 + \epsilon)$ approximation for $\text{CSD}(\mathbf{a}, \mathbf{b})$ with probability at least $1 - \delta$.*

5 Conclusion

We presented one-pass data stream algorithms for detecting periodic sequences and cyclic shifts, and for measuring the distance to the closest periodic sequence

or cyclic shift. Our principle goal was to minimize the space used, and all of our periodicity algorithms used $O(\text{polylog } n)$ space. Our algorithms used a range of techniques including bilinear sketches and combining a Fourier change of basis transform with a range of sketching techniques. This second technique is particularly powerful and we would be surprised if it didn't have applications that were still to be discovered (either via the Fourier basis or other bases). An important future direction is analyzing the structure of the sketches formed by combining the transform and sketch matrices: among other things, this could lead to more time-efficient algorithms. Another question is to generalize our results in Sects. 3.2 and 3.3 to estimate the period of signals that conclude with a partial repetition. This was not an issue with time-series data since there would always be a point near the end of the stream where there had been an exact number of repetitions. In the turnstile model the issue is more complicated, but we are hopeful that a more involved analysis of the Fourier approach may yield results.

Acknowledgements. We thank an anonymous reviewer for mentioning that ideas from the pattern matching result in Ergün et al. can be applied to cyclic shifts in the time-series model without requiring a second pass. We also thank Graham Cormode for suggesting a simplification to our cyclic shift algorithm.

References

1. Alon, N., Matias, Y., Szegedy, M.: The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.* 58(1), 137–147 (1999)
2. Cormode, G., Muthukrishnan, S.: An improved data stream summary: The count-min sketch and its applications. *J. Algorithms* 55, 58–75 (2005)
3. Czuma, A., Gasieniec, L.: On the complexity of determining the period of a string. In: Giancarlo, R., Sankoff, D. (eds.) *CPM 2000*. LNCS, vol. 1848, pp. 412–422. Springer, Heidelberg (2000)
4. Ergün, F., Jowhari, H., Saglam, M.: Periodicity in streams. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) *APPROX 2010*, LNCS, vol. 6302, pp. 545–559. Springer, Heidelberg (2010)
5. Ergün, F., Muthukrishnan, S., Sahinalp, S.C.: Periodicity testing with sublinear samples and space. *ACM Transactions on Algorithms* 6(2) (2010)
6. Gilbert, A.C., Guha, S., Indyk, P., Muthukrishnan, S., Strauss, M.: Near-optimal sparse fourier representations via sampling. In: *STOC*, pp. 152–161 (2002)
7. Hardy, G.H., Wright, E.M.: *An Introduction to The Theory of Numbers* (Fourth Edition). Oxford University Press, Oxford (1960)
8. Indyk, P.: Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM* 53(3), 307–323 (2006)
9. Indyk, P., Koudas, N., Muthukrishnan, S.: Identifying representative trends in massive time series data sets using sketches. In: *VLDB*, pp. 363–372 (2000)
10. Indyk, P., McGregor, A.: Declaring independence via the sketching of sketches. In: *SODA*, pp. 737–745 (2008)
11. Kane, D.M., Nelson, J., Woodruff, D.P.: On the exact space complexity of sketching and streaming small norms. In: *SODA*, pp. 1161–1178 (2010)

12. Kane, D.M., Nelson, J., Woodruff, D.P.: An optimal algorithm for the distinct elements problem. In: PODS, pp. 41–52 (2010)
13. Monemizadeh, M., Woodruff, D.P.: 1-pass relative-error L_p -sampling with applications. In: SODA (2010)
14. Muthukrishnan, S.: Data streams: Algorithms and applications. *Foundations and Trends in Theoretical Computer Science* 1(2) (2005)
15. Nisan, N.: Pseudorandom generators for space-bounded computation. *Combinatorica* 12, 449–461 (1992)
16. Porat, B., Porat, E.: Exact and approximate pattern matching in the streaming model. In: FOCS, pp. 315–323 (2009)
17. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* 26(5), 1484–1509 (1997)

An Approximation Algorithm for the Tree t -Spanner Problem on Unweighted Graphs via Generalized Chordal Graphs

Feodor F. Dragan¹ and Ekkehard Köhler²

¹ Algorithmic Research Laboratory, Department of Computer Science,
Kent State University, Kent, OH 44242, USA

dragan@cs.kent.edu

² Mathematisches Institut, Brandenburgische Technische Universität Cottbus,
D-03013 Cottbus, Germany
ekoehler@math.tu-cottbus.de

Abstract. A spanning tree T of a graph G is called a *tree t -spanner* of G if the distance between every pair of vertices in T is at most t times their distance in G . In this paper, we present an algorithm which constructs for an n -vertex m -edge unweighted graph G : (1) a tree $(2\lceil\log_2 n\rceil)$ -spanner in $O(m \log n)$ time, if G is a chordal graph; (2) a tree $(2\rho\lceil\log_2 n\rceil)$ -spanner in $O(mn \log^2 n)$ time or a tree $(12\rho\lceil\log_2 n\rceil)$ -spanner in $O(m \log n)$ time, if G is a graph admitting a Robertson-Seymour's tree-decomposition with bags of radius at most ρ in G ; and (3) a tree $(2\lceil t/2\rceil\lceil\log_2 n\rceil)$ -spanner in $O(mn \log^2 n)$ time or a tree $(6t\lceil\log_2 n\rceil)$ -spanner in $O(m \log n)$ time, if G is an arbitrary graph admitting a tree t -spanner. For the latter result we use a new necessary condition for a graph to have a tree t -spanner: if a graph G has a tree t -spanner, then G admits a Robertson-Seymour's tree-decomposition with bags of radius at most $\lceil t/2\rceil$ in G .

1 Introduction

Given a connected graph G and a spanning tree T of G , we say that T is a *tree t -spanner* of G if the distance between every pair of vertices in T is at most t times their distance in G . The parameter t is called the *stretch* (or *stretch factor*) of T . The TREE t -SPANNER problem asks, given a graph G and a positive number t , whether G admits a tree t -spanner. Note that the problem of finding a tree t -spanner of G minimizing t is known in literature also as the Minimum Max-Stretch spanning Tree problem (see, e.g., [14] and literature cited therein). This paper concerns the TREE t -SPANNER problem on unweighted graphs. The problem is known to be NP-complete even for planar graphs and chordal graphs (see [5, 8, 15]), and the paper presents an efficient algorithm which produces a tree t -spanner with $t \leq 2\log_2 n$ for every n -vertex chordal graph and a tree $(2\lceil t/2\rceil\lceil\log_2 n\rceil)$ -spanner for an arbitrary n -vertex graph admitting a tree t -spanner. To obtain the latter result, we show that every graph having a tree t -spanner admits a Robertson-Seymour's tree-decomposition with bags of radius at most $\lceil t/2\rceil$ in G . This tree-decomposition is a generalization of the well-known

notion of a clique-tree of a chordal graph, and allows us to extend our algorithm developed for chordal graphs to arbitrary graphs admitting tree t -spanners.

There are many applications of tree spanners in various areas. We refer to the survey paper of Peleg [21] for an overview on spanners and their applications.

Related work. Substantial work has been done on the TREE t -SPANNER problem on unweighted graphs. Cai and Cornil [8] have shown that, for a given graph G , the problem to decide whether G has a tree t -spanner is NP-complete for any fixed $t \geq 4$ and is linear time solvable for $t = 1, 2$ (the status of the case $t = 3$ is open for general graphs)¹. The NP-completeness result was further strengthened in [5] and [6], where Branstädt et al. showed that the problem remains NP-complete even for the class of chordal graphs (i.e., for graphs where each induced cycle has length 3) and every fixed $t \geq 4$, and for the class of chordal bipartite graphs (i.e., for bipartite graphs where each induced cycle has length 4) and every fixed $t \geq 5$.

The TREE t -SPANNER problem on planar graphs was studied in [15,23]. In [23], Peleg and Tendler presented a polynomial time algorithm for the minimum value of t for the TREE t -SPANNER problem on outerplanar graphs. In [15], Fekete and Kremer proved that the TREE t -SPANNER problem on planar graphs is NP-complete (when t is part of the input) and polynomial time solvable for $t = 3$. They also gave a polynomial time algorithm that for every fixed t decides for planar graphs with bounded face length whether there is a tree t -spanner. For fixed $t \geq 4$, the complexity of the TREE t -SPANNER problem on arbitrary planar graphs was left as an open problem in [15]. This open problem was recently resolved in [12], where it was shown that the TREE t -SPANNER problem is linear time solvable for every fixed constant t on the class of apex-minor-free graphs which includes all planar graphs and all graphs of bounded genus.

An $O(\log n)$ -approximation algorithm for the minimum value of t for the TREE t -SPANNER problem is due to Emek and Peleg [14], and until recently that was the only $O(\log n)$ -approximation algorithm available for the problem. Let G be an n -vertex m -edge unweighted graph and t^* be the minimum value such that a tree t^* -spanner exists for G . Emek and Peleg gave an algorithm which produces for every G a tree $(6t^* \lceil \log_2 n \rceil)$ -spanner in $O(mn \log^2 n)$ time. Furthermore, they established that unless $P = NP$, the problem cannot be approximated additively by any $o(n)$ term. Hardness of approximation is established also in [19], where it was shown that approximating the minimum value of t for the TREE t -SPANNER problem within factor better than 2 is NP-hard (see also [22] for an earlier result). Recently, another logarithmic approximation algorithm for the TREE t -SPANNER problem was announced in [3], but authors did not provide any details. A number of papers have studied the related but easier problem of finding a spanning tree with good *average* stretch factor (see [12,13] and papers cited therein).

Our contribution. In this paper, we present a new algorithm which constructs for an n -vertex m -edge unweighted graph G : (1) a tree $(2 \lfloor \log_2 n \rfloor)$ -spanner in $O(m \log n)$ time, if G is a chordal graph; (2) a tree $(2\rho \lfloor \log_2 n \rfloor)$ -spanner in

¹ When G is an unweighted graph, t can be assumed to be an integer.

$O(mn \log^2 n)$ time or a tree $(12\rho \lceil \log_2 n \rceil)$ -spanner in $O(m \log n)$ time, if G is a graph admitting a Robertson-Seymour's tree-decomposition with bags of radius at most ρ in G ; and (3) a tree $(2 \lceil t/2 \rceil \lceil \log_2 n \rceil)$ -spanner in $O(mn \log^2 n)$ time or a tree $(6t \lceil \log_2 n \rceil)$ -spanner in $O(m \log n)$ time, if G is an arbitrary graph admitting a tree t -spanner. For the latter result we employ a new necessary condition for a graph to have a tree t -spanner: if a graph G has a tree t -spanner, then G admits a Robertson-Seymour's tree-decomposition with bags of radius at most $\lceil t/2 \rceil$ and diameter at most t in G . The algorithm needs to know neither an appropriate Robertson-Seymour's tree-decomposition of G nor the true value of t . It works directly on an input graph G .

A high-level description of our method is similar to that of [14], although the details are very different. We find a "small radius" balanced disk-separator of a graph $G = (V, E)$, that is, a disk $D_r(v, G)$ of radius r and centered at vertex v such that removal of vertices of $D_r(v, G)$ from G leaves no connected component with more than $n/2$ vertices. We recursively build a spanning tree for each graph formed by a connected component G_i of $G[V \setminus D_r(v, G)]$ with one additional vertex v added to G_i to represent the disk $D_r(v, G)$ and its adjacency relation to G_i . The spanning trees generated by recursive invocations of the algorithm on each such graph are glued together at vertex v and then the vertex v of the resulting tree is substituted with a single source shortest path spanning tree of $D_r(v, G)$ to produce a spanning tree T of G . Analysis of the algorithm relies on an observation that the number of edges added to the unique path between vertices x and y in T , where xy is an edge of G , on each of $\lceil \log_2 n \rceil$ recursive levels is at most $2r$.

Comparing with the algorithm of Emek and Peleg ([14]), one variant of our algorithm has the same approximation ratio but a better run-time, other variant has the same run-time but a better constant term in the approximation ratio. Our algorithm and its analysis, in our opinion, are conceptually simpler due to a new necessary condition for a graph to have a tree t -spanner.

2 Preliminaries

All graphs occurring in this paper are connected, finite, unweighted, undirected, loopless and without multiple edges. We call $G = (V, E)$ an n -vertex m -edge graph if $|V| = n$ and $|E| = m$. A *clique* is a set of pairwise adjacent vertices of G . By $G[S]$ we denote a subgraph of G induced by vertices of $S \subseteq V$. Let also $G \setminus S$ be the graph $G[V \setminus S]$ (which is not necessarily connected). A set $S \subseteq V$ is called a *separator* of G if the graph $G[V \setminus S]$ has more than one connected component, and S is called a *balanced separator* of G if each connected component of $G[V \setminus S]$ has at most $|V|/2$ vertices. A set $C \subseteq V$ is called a *balanced clique-separator* of G if C is both a clique and a balanced separator of G . For a vertex v of G , the sets $N_G(v) = \{w \in V : vw \in E\}$ and $N_G[v] = N_G(v) \cup \{v\}$ are called the *open neighborhood* and the *closed neighborhood* of v , respectively.

In a graph G the *length* of a path from a vertex v to a vertex u is the number of edges in the path. The *distance* $d_G(u, v)$ between vertices u and v is the

length of a shortest path connecting u and v in G . The *diameter* in G of a set $S \subseteq V$ is $\max_{x,y \in S} d_G(x,y)$ and its *radius* in G is $\min_{x \in V} \max_{y \in S} d_G(x,y)$ (in some papers they are called the *weak diameter* and the *weak radius* to indicate that the distances are measured in G not in $G[S]$). The *disk* of G of radius k centered at vertex v is the set of all vertices at distance at most k to v : $D_k(v,G) = \{w \in V : d_G(v,w) \leq k\}$. A disk $D_k(v,G)$ is called a *balanced disk-separator* of G if the set $D_k(v,G)$ is a balanced separator of G .

Let G be a connected graph and t be a positive number. A spanning tree T of G is called a *tree t -spanner* of G if the distance between every pair of vertices in T is at most t times their distance in G , i.e., $d_T(x,y) \leq t d_G(x,y)$ for every pair of vertices x and y of G . It is easy to see that the tree t -spanners can equivalently be defined as follows.

Proposition 1. *Let G be a connected graph and t be a positive number. A spanning tree T of G is a tree t -spanner of G if and only if for every edge xy of G , $d_T(x,y) \leq t$ holds.*

This proposition implies that the stretch of a spanning tree of a graph G is always obtained on a pair of vertices that form an edge in G . Consequently, throughout this paper t can be considered as an integer which is greater than 1.

3 Tree Spanners of Chordal Graphs

As we have mentioned earlier the TREE t -SPANNER problem is NP-complete for every $t \geq 4$ even for the class of chordal graphs [5]. Recall that a graph G is called *chordal* if each induced cycle of G has length 3. In this section, we show that every chordal graph with n vertices admits a tree t -spanner with $t \leq 2 \log_2 n$. In the full version of the paper (see [26]), we show also that there are chordal graphs for which any tree t -spanner has to have $t \geq \log_2 \frac{n}{3} + 2$. All proofs omitted in this extended abstract can also be found in the full version.

We start with three lemmas that are crucial to our method. Let $G = (V, E)$ be an arbitrary connected graph with a clique-separator C , i.e., there is a clique C in G such that the removal of the vertices of C from G results in a graph with more than one connected component. Let G_1, \dots, G_k be those connected components of $G[V \setminus C]$. Denote by $S_i := \{x \in V(G_i) : d_G(x,C) = 1\}$ the neighborhood of C with respect to G_i . Let also G_i^+ be the graph obtained from component G_i by adding a vertex c_i (*representative* of C) and making it adjacent to all vertices of S_i , i.e., for a vertex $x \in V(G_i)$, $c_i x \in E(G_i^+)$ if and only if there is a vertex $x_C \in C$ with $xx_C \in E(G)$ (see Fig. 1). Clearly, given a connected m -edge graph G and a clique-separator C of G , the graphs G_1^+, \dots, G_k^+ can be constructed in total time $O(m)$. Note also that the total number of edges in graphs G_1^+, \dots, G_k^+ does not exceed the number of edges in G .

Denote by $G_{/e}$ the graph obtained from G by contracting its edge e . Recall that edge e contraction is an operation which removes e from G while simultaneously merging together the two vertices e previously connected. If a contraction results in multiple edges, we delete duplicates of an edge to stay within the

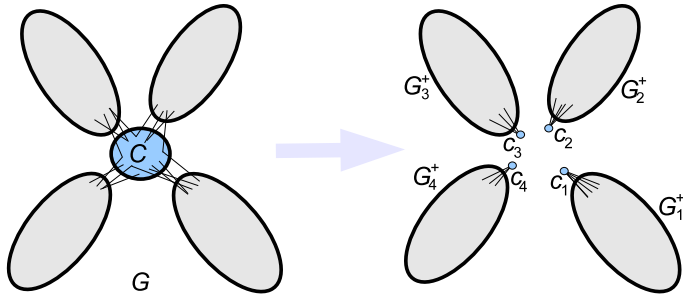


Fig. 1. A graph G with a clique-separator C and the corresponding graphs G_1^+, \dots, G_4^+ obtained from G

class of simple graphs. The operation may be performed on a set of edges by contracting each edge (in any order).

Lemma 1. *If a graph G is chordal then $G_{/e}$ is chordal as well, for any edge $e \in E(G)$. Consequently, if a graph G is chordal then G_i^+ is chordal as well, for each $i = 1, \dots, k$.*

Let T_i ($i = 1, \dots, k$) be a spanning tree of G_i^+ such that for any edge $xy \in E(G_i^+)$, $d_{T_i}(x, y) \leq \alpha$ holds, where α is some positive integer independent of i . We can form a spanning tree T of G from trees T_1, \dots, T_k and the vertices of the clique C in the following way. For each $i = 1, \dots, k$, rename vertex c_i in T_i to c . Glue trees T_1, \dots, T_k together at vertex c obtaining a tree T' (see Fig. 2). For the original clique C of G , pick an arbitrary vertex r_C of C and create a spanning star ST_C for C centered at r_C . Substitute vertex c in T' by that star ST_C . For each former edge xc of T' , create an edge xx_C in T where x_C is a vertex of C adjacent to x in G . We can show that for any edge $xy \in E(G)$, $d_T(x, y) \leq \alpha + 2$ holds. Evidently, the tree T of G can be constructed from trees T_1, \dots, T_k and the vertices of the clique C in $O(m)$ time.

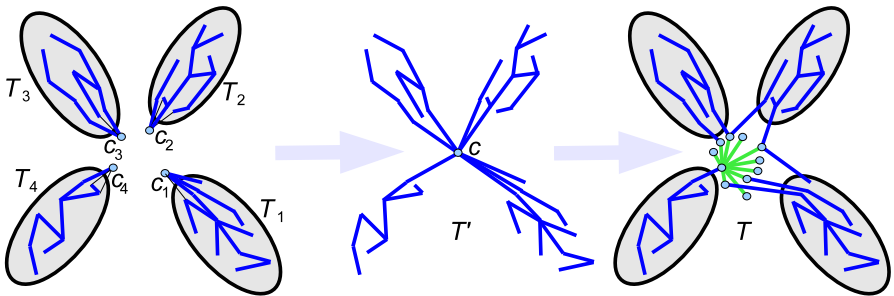


Fig. 2. Spanning trees T_1, \dots, T_4 of G_1^+, \dots, G_4^+ , resulting tree T' , and a corresponding spanning tree T of G

Lemma 2. *Let G be an arbitrary graph with a clique-separator C and G_1^+, \dots, G_k^+ be the graphs obtained from G as described above. Let also T_i ($i \in \{1, \dots, k\}$) be a spanning tree of the graph G_i^+ , and T be a spanning tree of G constructed from T_1, \dots, T_k and the clique C as described above. Assume also that there is a positive integer α such that, for each $i \in \{1, \dots, k\}$ and every edge $xy \in E(G_i^+)$, $d_{T_i}(x, y) \leq \alpha$ holds. Then, for every edge $xy \in E(G)$, $d_T(x, y) \leq \alpha + 2$ holds.*

Proof. Consider an arbitrary edge xy of G . If both x and y belong to C , then evidently $d_T(x, y) \leq 2 < \alpha + 2$. Assume now that xy is an edge of G_i for some $i \in \{1, \dots, k\}$. Then, xy is an edge of G_i^+ and therefore $d_{T_i}(x, y) \leq \alpha$. If the path P of T_i connecting x and y does not contain vertex c_i , then $d_T(x, y) = d_{T_i}(x, y) \leq \alpha$ must hold. If c_i is between x and y in T_i (i.e., $c_i \in P$), then the distance in T between x and y is at most $d_{T_i}(x, y) + 2$ (the path of T between x and y is obtained from P by substituting the vertex $c = c_i$ by a path of star ST_C with at most 2 edges). It remains to consider the case when $x \in C$ and $y \in V(G_i)$. By construction of G_i^+ , there must exist an edge c_iy in G_i^+ . We have $d_{T_i}(c_i, y) \leq \alpha$. Let z be the neighbor of c_i in the path of T_i connecting vertices y and c_i ($y = z$ is possible). Evidently, $z \in V(G_i)$. By construction, in T we must have an edge zz_c where z_c is a vertex of C adjacent to z in G . Vertices x and z_c both are in C and the distance in T between them is at most 2. Thus, $d_T(x, y) \leq d_T(z_c, y) + 2 = d_{T_i}(c_i, y) + 2 \leq \alpha + 2$. □

The third important ingredient to our method is the famous chordal balanced separator result of Gilbert, Rose, and Edenbrandt [18].

Lemma 3. [18] *Every connected chordal graph G with n vertices and m edges contains a balanced clique-separator which can be found in $O(m)$ time.*

Now let $G = (V, E)$ be a connected chordal graph with n vertices and m edges. Using Lemma 1 and Lemma 3, we can build a (rooted) *hierarchical-tree* $\mathcal{H}(G)$ for G , which can be constructed as follows. If G is a connected graph with at most 5 vertices or is a clique of size greater than 5, then $\mathcal{H}(G)$ is a one node tree with root node (G, nil) . Otherwise, find a balanced clique-separator C of G (which exists by Lemma 3 and which can be found in $O(m)$ time) and construct the associated graphs G_1^+, \dots, G_k^+ . For each graph G_i^+ , $i \in \{1, \dots, k\}$, which is chordal by Lemma 1, construct a hierarchical-tree $\mathcal{H}(G_i^+)$ recursively and build $\mathcal{H}(G)$ by taking the pair (G, C) to be the root and connecting the root of each tree $\mathcal{H}(G_i^+)$ as a child of (G, C) . The depth of this tree $\mathcal{H}(G)$ is the smallest integer k such that $\frac{n}{2^k} + \frac{1}{2^{k-1}} + \dots + \frac{1}{2} + 1 \leq 5$, that is, the depth is at most $\log_2 n - 1$.

To build a tree t -spanner T of G , we use the hierarchical-tree $\mathcal{H}(G)$ and a bottom-up construction. We know from Proposition 1 that a spanning tree T is a tree t -spanner of a graph G if and only if for any edge xy of G , $d_T(x, y) \leq t$ holds. For each leaf (L, nil) of $\mathcal{H}(G)$ (we know that graph L is a clique or a connected chordal graph with at most 5 vertices), we construct a tree 2-spanner T_L of L . It is easy to see that L admits such a tree 2-spanner. Hence, for any edge xy of L , we have $d_{T_L}(x, y) \leq 2$. Consider now an inner node (H, K) of

$\mathcal{H}(G)$, and assume that all its children H_1^+, \dots, H_l^+ in $\mathcal{H}(G)$ have tree α -spanners T_1, \dots, T_l for some positive integer α . Then, a tree $(\alpha + 2)$ -spanner of H can be constructed from T_1, \dots, T_l and clique K of H as described above (see Lemma 2 and paragraph before it). Since the depth of the hierarchical-tree $\mathcal{H}(G)$ is at most $\log_2 n - 1$ and all leaves of $\mathcal{H}(G)$ admit tree 2-spanners, applying Lemma 2 repeatedly, we will move from leaves to the root of $\mathcal{H}(G)$ and get a tree t -spanner T of G with t being no more than $2 \log_2 n$.

It is also easy to see that, given a chordal graph G with n vertices and m edges, a hierarchical-tree $\mathcal{H}(G)$ as well as a tree t -spanner T of G with $t \leq 2 \log_2 n$ can be constructed in $O(m \log n)$ total time since there are at most $O(\log n)$ levels in $\mathcal{H}(G)$ and one needs to do at most $O(m)$ operations per level. Thus, we have the following result for the class of chordal graphs.

Theorem 1. *Any connected chordal graph G with n vertices and m edges admits a tree $(2\lceil \log_2 n \rceil)$ -spanner constructible in $O(m \log n)$ time.*

4 Tree Spanners of Generalized Chordal Graphs

It is known that the class of chordal graphs can be characterized in terms of existence of so-called *clique-trees*. Let $\mathcal{C}(G)$ denote the family of maximal (by inclusion) cliques of a graph G . A *clique-tree* $\mathcal{CT}(G)$ of G has the maximal cliques of G as its nodes, and for every vertex v of G , the maximal cliques containing v form a subtree of $\mathcal{CT}(G)$.

Theorem 2. [7,17] *A graph is chordal if and only if it has a clique-tree.*

In their work on graph minors [25], Robertson and Seymour introduced the notion of tree-decomposition which generalizes the notion of clique-tree. A *tree-decomposition* of a graph G is a tree $\mathcal{T}(G)$ whose nodes, called *bags*, are subsets of $V(G)$ such that: (1) $\bigcup_{X \in V(\mathcal{T}(G))} X = V(G)$, (2) for each edge $vw \in E(G)$, there is a bag $X \in V(\mathcal{T}(G))$ such that $v, w \in X$, and (3) for each $v \in V(G)$ the set of bags $\{X : X \in V(\mathcal{T}(G)), v \in X\}$ forms a subtree $\mathcal{T}_v(G)$ of $\mathcal{T}(G)$.

Tree-decompositions were used in defining at least two graph parameters. The *tree-width* of a graph G is defined as minimum of $\max_{X \in V(\mathcal{T}(G))} |X| - 1$ over all tree-decompositions $\mathcal{T}(G)$ of G and is denoted by $\text{tw}(G)$ [25]. The *length* of a tree-decomposition $\mathcal{T}(G)$ of a graph G is $\max_{X \in V(\mathcal{T}(G))} \max_{u, v \in X} d_G(u, v)$, and the *tree-length* of G , denoted by $\text{tl}(G)$, is the minimum of the length, over all tree-decompositions of G [11]. These two graph parameters are not related to each other. Interestingly, the tree-length of a graph can be approximated in polynomial time within a constant factor [11] whereas such an approximation factor is unknown for the tree-width.

For the purpose of this paper, we introduce yet another graph parameter based on the notion of tree-decomposition. It is very similar to the notion of tree-length but is more appropriate for our discussions, and moreover it will lead to a better constant in our approximation ratio presented in Section 5 for the TREE t -SPANNER problem on general graphs.

Definition 1. The *breadth* of a tree-decomposition $\mathcal{T}(G)$ of a graph G is the minimum integer k such that for every $X \in V(\mathcal{T}(G))$ there is a vertex $v_X \in V(G)$ with $X \subseteq D_k(v_X, G)$ (i.e., each bag X has radius at most k in G). Note that vertex v_X does not need to belong to X . The *tree-breadth* of G , denoted by $\text{tb}(G)$, is the minimum of the breadth, over all tree-decompositions of G . We say that a family of graphs \mathcal{G} is of *bounded tree-breadth*, if there is a constant c such that for each graph G from \mathcal{G} , $\text{tb}(G) \leq c$.

Evidently, for any graph G , $1 \leq \text{tb}(G) \leq \text{tl}(G) \leq 2\text{tb}(G)$ holds. Hence, if one parameter is bounded by a constant for a graph G then the other parameter is bounded for G as well.

In what follows, we will show that any graph G with tree-breadth $\text{tb}(G) \leq \rho$ admits a tree $(2\rho \lceil \log_2 n \rceil)$ -spanner, thus generalizing the result for chordal graphs of Section 3 (if G is chordal then $\text{tl}(G) = \text{tb}(G) = 1$). It is interesting to note that the TREE t -SPANNER problem is NP-complete for graphs of bounded tree-breadth (even for chordal graphs for every fixed $t > 3$; see [5]), while it is polynomial time solvable for all graphs of bounded tree-width (see [24]).

First we present a balanced separator result.

Lemma 4. *Every graph G with n vertices, m edges and with tree-breadth at most ρ contains a vertex v such that $D_\rho(v, G)$ is a balanced disk-separator of G .*

Proof. The proof of this lemma follows from *acyclic hypergraph* theory. First we review some necessary definitions and an important result characterizing acyclic hypergraphs. Recall that a *hypergraph* H is a pair $H = (V, \mathcal{E})$ where V is a set of vertices and \mathcal{E} is a set of non-empty subsets of V called *hyperedges*. For these and other hypergraph notions see [4].

Let $H = (V, \mathcal{E})$ be a *hypergraph* with the vertex set V and the *hyperedge* set \mathcal{E} . For every vertex $v \in V$, let $\mathcal{E}(v) = \{e \in \mathcal{E} : v \in e\}$. The *2-section graph* $2SEC(H)$ of a hypergraph H has V as its vertex set and two distinct vertices are adjacent in $2SEC(H)$ if and only if they are contained in a common hyperedge of H . A hypergraph H is called *conformal* if every clique of $2SEC(H)$ is contained in a hyperedge $e \in \mathcal{E}$, and a hypergraph H is called *acyclic* if there is a tree T with node set \mathcal{E} such that for all vertices $v \in V$, $\mathcal{E}(v)$ induces a subtree T_v of T . It is a well-known fact (see, e.g., [4]) that a hypergraph H is acyclic if and only if H is conformal and $2SEC(H)$ of H is a chordal graph.

Let now G be a graph with $\text{tb}(G) = \rho$ and $\mathcal{T}(G)$ be its tree-decomposition of breadth ρ . Evidently, property (3) in the definition of tree-decomposition can be restated as follows: the hypergraph $H = (V(G), \{X : X \in V(\mathcal{T}(G))\})$ is an acyclic hypergraph. Since each edge of G is contained in at least one bag of $\mathcal{T}(G)$, the 2-section graph $G^* := 2SEC(H)$ of H is a chordal *supergraph* of the graph G (each edge of G is an edge of G^* , but G^* may have some extra edges between non-adjacent vertices of G contained in a common bag of $\mathcal{T}(G)$). By Lemma 3, the chordal graph G^* contains a balanced clique-separator $C \subseteq V(G)$. By conformality of H , C must be contained in a bag of $\mathcal{T}(G)$. Hence, there must exist a vertex $v \in V(G)$ with $C \subseteq D_\rho(v, G)$. As the removal of the vertices of C from G^* leaves no connected component in $G^*[V \setminus C]$ with more than $n/2$

vertices and since G^* is a supergraph of G , clearly, the removal of the vertices of $D_\rho(v, G)$ from G leaves no connected component in $G[V \setminus D_\rho(v, G)]$ with more than $n/2$ vertices. \square

We do not need to know a tree-decomposition $\mathcal{T}(G)$ of breadth ρ to find such a balanced disk-separator $D_\rho(v, G)$ of G . For a given graph G and an integer ρ , checking whether G has a tree-decomposition of breadth ρ could be a hard problem. For example, while graphs with tree-length 1 (as they are exactly the chordal graphs) can be recognized in linear time, the problem of determining whether a given graph has tree-length at most λ is NP-complete for every fixed $\lambda > 1$ (see [20]). Instead, we can use the following result.

Proposition 2. *For an arbitrary graph G with n vertices and m edges a balanced disk-separator $D_r(v, G)$ with minimum r can be found in $O(nm)$ time.*

Now let $G = (V, E)$ be an arbitrary connected n -vertex m -edge graph with a disk-separator $D_r(v, G)$. As in the case of chordal graphs, let G_1, \dots, G_k be the connected components of $G[V \setminus D_r(v, G)]$. Denote by $S_i := \{x \in V(G_i) : d_G(x, D_r(v, G)) = 1\}$ the neighborhood of $D_r(v, G)$ with respect to G_i . Let also G_i^+ be the graph obtained from component G_i by adding a vertex v_i (representative of $D_r(v, G)$) and making it adjacent to all vertices of S_i , i.e., for a vertex $x \in V(G_i)$, $v_i x \in E(G_i^+)$ if and only if there is a vertex $x_D \in D_r(v, G)$ with $xx_D \in E(G)$. Given graph G and its disk-separator $D_r(v, G)$, the graphs G_1^+, \dots, G_k^+ can be constructed in total time $O(m)$. Furthermore, the total number of edges in the graphs G_1^+, \dots, G_k^+ does not exceed the number of edges in G , and the total number of vertices in those graphs does not exceed the number of vertices in $G[V \setminus D_r(v, G)]$ plus k . Let again $G_{/e}$ be the graph obtained from G by contracting its edge e .

Lemma 5. *For any graph G and its edge e , $\text{tb}(G) \leq \rho$ implies $\text{tb}(G_{/e}) \leq \rho$. Consequently, for any graph G with $\text{tb}(G) \leq \rho$, $\text{tb}(G_i^+) \leq \rho$ holds for each i .*

As in Section 3, let T_i ($i = 1, \dots, k$) be a spanning tree of G_i^+ such that for any edge $xy \in E(G_i^+)$, $d_{T_i}(x, y) \leq \alpha$ holds, where α is some positive integer independent of i . For the disk $D_r(v, G)$ of G , construct a shortest path tree SPT_D rooted at vertex v (and spanning all and only the vertices of the disk). We can form a spanning tree T of G from trees T_1, \dots, T_k and SPT_D in the following way. For each $i = 1, \dots, k$, rename vertex v_i in T_i to v . Glue trees T_1, \dots, T_k together at vertex v obtaining a tree T' (consult with Fig. 2). Substitute vertex v in T' by the tree SPT_D . For each former edge xv of T' , create an edge xx_D in T where x_D is a vertex of $D_r(v, G)$ adjacent to x in G . We can show that for any edge $xy \in E(G)$, $d_T(x, y) \leq \alpha + 2r$ holds. Evidently, the tree T of G can be constructed from trees T_1, \dots, T_k and SPT_D in $O(m)$ time.

Lemma 6. *Let G be an arbitrary graph with a disk-separator $D_r(v, G)$ and G_1^+, \dots, G_k^+ be the graphs obtained from G as described above. Let also T_i ($i \in \{1, \dots, k\}$) be a spanning tree of the graph G_i^+ , and T be a spanning tree of G*

constructed from T_1, \dots, T_k and a shortest path tree SPT_D of the disk $D_r(v, G)$ as described above. Assume also that there is a positive integer α such that, for each $i \in \{1, \dots, k\}$ and every edge $xy \in E(G_i^+)$, $d_{T_i}(x, y) \leq \alpha$ holds. Then, for every edge $xy \in E(G)$, $d_T(x, y) \leq \alpha + 2r$ must hold.

Now we have all necessary ingredients to apply the technique used in Section 3 and show that each graph G admits a tree $(2\text{tb}(G)\lceil \log_2 n \rceil)$ -spanner.

Let $G = (V, E)$ be a connected n -vertex, m -edge graph and assume that $\text{tb}(G) \leq \rho$. Lemma 4 guaranties that G has a balanced disk-separator $D_r(v, G)$ with $r \leq \rho$. Proposition 2 says that such a balanced disk-separator $D_r(v, G)$ of G can be found in $O(nm)$ time by an algorithm that works directly on graph G and does not require the construction of a tree-decomposition of G of breadth $\leq \rho$. Using this and Lemma 5, we can build as before a (rooted) hierarchical-tree $\mathcal{H}(G)$ for G . Only now, the leaves of $\mathcal{H}(G)$ are connected graphs with at most 9 vertices. It is not hard to see that any leaf of $\mathcal{H}(G)$ has a tree t -spanner with $t \leq 4\rho$. Furthermore, a simple analysis shows that the depth of this tree $\mathcal{H}(G)$ is at most $\log_2 n - 2$.

To build a tree t -spanner T of G , we again use the hierarchical-tree $\mathcal{H}(G)$ and a bottom-up construction. Each leaf (L, nil) of $\mathcal{H}(G)$ has a tree (4ρ) -spanner. A tree t -spanner with minimum t of such a small graph L can be computed directly. Consider now an inner node $(H, D_r(v, G))$ of $\mathcal{H}(G)$ (where $D_r(v, G)$ is a balanced disk-separator of H), and assume that all its children H_1^+, \dots, H_l^+ in $\mathcal{H}(G)$ have tree α -spanners T_1, \dots, T_l for some positive integer α . Then, a tree $(\alpha + 2r)$ -spanner of H can be constructed from T_1, \dots, T_l and a shortest path tree SPT_D of the disk $D_r(v, G)$ as described above (see Lemma 6 and paragraph before it). Since the depth of the hierarchical-tree $\mathcal{H}(G)$ is at most $\log_2 n - 2$ and all leaves of $\mathcal{H}(G)$ admit tree (4ρ) -spanners, applying Lemma 6 repeatedly, we move from leaves to the root of $\mathcal{H}(G)$ and get a tree t -spanner T of G with t being no more than $2\rho \log_2 n$. It is also easy to see that, given a graph G with n vertices and m edges, a hierarchical-tree $\mathcal{H}(G)$ as well as a tree t -spanner T of G with $t \leq 2\text{tb}(G) \log_2 n$ can be constructed in $O(nm \log^2 n)$ total time. There are at most $O(\log n)$ levels in $\mathcal{H}(G)$, and one needs to do at most $O(nm \log n)$ operations per level since the total number of edges in the graphs of each level is at most m and the total number of vertices in those graphs can not exceed $O(n \log n)$.

Note that our algorithm does not need to know the value of $\text{tb}(G)$, neither it needs to know any appropriate Robertson-Seymour's tree-decomposition of G . It works directly on an input graph. To indicate this, we say that the algorithm constructs an appropriate tree spanner *from scratch*.

Thus, we have the following results.

Theorem 3. *There is an algorithm that for an arbitrary connected graph G with n vertices and m edges constructs a tree $(2\text{tb}(G)\lceil \log_2 n \rceil)$ -spanner of G in $O(nm \log^2 n)$ total time.*

Corollary 1. *Any connected n -vertex, m -edge graph G with $\text{tb}(G) \leq \rho$ admits a tree $(2\rho\lceil \log_2 n \rceil)$ -spanner constructible in $O(nm \log^2 n)$ time from scratch.*

Corollary 2. *Any connected n -vertex, m -edge graph G with $\text{tl}(G) \leq \lambda$ admits a tree $(2\lambda \lfloor \log_2 n \rfloor)$ -spanner constructible in $O(nm \log^2 n)$ time from scratch.*

There is another natural generalization of chordal graphs. A graph G is called k -chordal if its largest induced cycle has length at most k . Chordal graphs are exactly 3-chordal graphs. It was shown in [16] that every k -chordal graph has tree-length at most $k/2$. Thus, we have one more corollary.

Corollary 3. *Any connected n -vertex, m -edge k -chordal graph G admits a tree $(2 \lfloor k/2 \rfloor \lfloor \log_2 n \rfloor)$ -spanner constructible in $O(nm \log^2 n)$ time from scratch.*

5 Approximating Tree t -Spanners of General Graphs

In this section, we show that the results obtained for tree t -spanners of generalized chordal graphs lead to an approximation algorithm for the TREE t -SPANNER problem on general (unweighted) graphs. We show that every graph G admitting a tree t -spanner has tree-breadth at most $\lceil t/2 \rceil$. From this and Theorem 3 it follows that there is an algorithm which produces for every n -vertex and m -edge graph G a tree $(2 \lceil t/2 \rceil \lfloor \log_2 n \rfloor)$ -spanner in $O(nm \log^2 n)$ time, whenever G admits a tree t -spanner. The algorithm does not even need to know the true value of t .

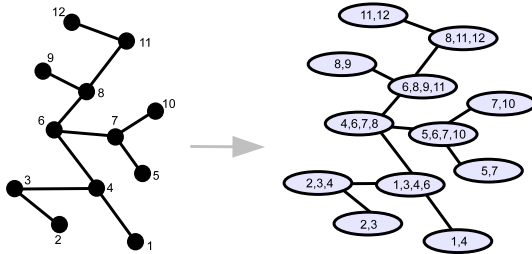


Fig. 3. From tree T to tree-decomposition \mathcal{T} with $t = 2$

Lemma 7. *If a graph G admits a tree t -spanner then $\text{tb}(G) \leq \lceil t/2 \rceil$.*

Proof. Let T be a tree t -spanner of G . We can transform this tree T to a tree-decomposition \mathcal{T} of G by expanding each vertex x in T to a bag X and putting all vertices of disk $D_{\lceil t/2 \rceil}(x, T)$ into that bag (note that the disk here is considered in T ; see Fig. 3 for an illustration). The edges of T and of \mathcal{T} are identical: XY is an edge in \mathcal{T} if and only if $xy \in E(T)$, where X is a bag that replaced vertex x in T and Y is a bag that replaced vertex y in T . Since $d_G(u, v) \leq d_T(u, v)$ for every pair of vertices u and v of G , we know that every bag $X := D_{\lceil t/2 \rceil}(x, T)$ is contained in a disk $D_{\lceil t/2 \rceil}(x, G)$ of G . It is easy to see that all three properties of tree-decomposition are fulfilled for \mathcal{T} .

Combining Lemma 7 with Theorem 3 we get our main result.

Theorem 4. *There is an algorithm that for an arbitrary connected graph G with n vertices and m edges constructs a tree $(2\lceil t/2 \rceil \lfloor \log_2 n \rfloor)$ -spanner in $O(nm \log^2 n)$ time, whenever G admits a tree t -spanner.*

The complexity of our algorithm is dominated by the complexity of finding a balanced disk-separator $D_r(v, G)$ of a graph G with minimum r . Proposition 2 says that for an n -vertex, m -edge graph such a balanced disk-separator can be found in $O(nm)$ time. In the full version of the paper, we show that a balanced disk-separator of a graph G with radius $r \leq 6 \cdot \text{tb}(G)$ can be found in linear $O(m)$ time. This immediately leads to the following result.

Theorem 5. *There is an algorithm that for an arbitrary connected graph G with n vertices and m edges constructs a tree $(6t \lfloor \log_2 n \rfloor)$ -spanner in $O(m \log n)$ time, whenever G admits a tree t -spanner.*

References

1. Abraham, I., Bartal, Y., Neiman, O.: Nearly Tight Low Stretch Spanning Trees. In: FOCS, pp. 781–790 (2008)
2. Alon, N., Karp, R.M., Peleg, D., West, D.B.: A Graph-Theoretic Game and Its Application to the k-Server Problem. *SIAM J. Comput.* 24, 78–100 (1995)
3. Bădoiu, M., Indyk, P., Sidiropoulos, A.: Approximation algorithms for embedding general metrics into trees. In: SODA 2007, pp. 512–521. SIAM, Philadelphia (2007)
4. Berge, C.: *Hypergraphs*. North-Holland, Amsterdam (1989)
5. Brandstädt, A., Dragan, F.F., Le, H.-O., Le, V.B.: on Chordal Graphs: Complexity and Algorithms. *Theoret. Comp. Science* 310, 329–354 (2004)
6. Brandstädt, A., Dragan, F.F., Le, H.-O., Le, V.B., Uehara, R.: Tree spanners for bipartite graphs and probe interval graphs. *Algorithmica* 47, 27–51 (2007)
7. Buneman, A.: A characterization of rigid circuit graphs. *Discrete Math.* 9, 205–212 (1974)
8. Cai, L., Corneil, D.G.: Tree spanners. *SIAM J. Discr. Math.* 8, 359–387 (1995)
9. Chepoi, V.D., Dragan, F.F., Newman, I., Rabinovich, Y., Vaxes, Y.: Constant Approximation Algorithms for Embedding Graph Metrics into Trees and Outerplanar Graphs. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX 2010, LNCS, vol. 6302, pp. 95–109. Springer, Heidelberg (2010)
10. Dourisboure, Y., Dragan, F.F., Gavoille, C., Yan, C.: Spanners for bounded tree-length graphs. *Theor. Comput. Sci.* 383, 34–44 (2007)
11. Dourisboure, Y., Gavoille, C.: Tree-decompositions with bags of small diameter. *Discrete Mathematics* 307, 2008–2029 (2007)
12. Dragan, F.F., Fomin, F., Golovach, P.: Spanners in sparse graphs. *J. Computer and System Sciences* (2010), doi: 10.1016/j.jcss.2010.10.002
13. Elkin, M., Emek, Y., Spielman, D.A., Teng, S.-H.: Lower-Stretch Spanning Trees. *SIAM J. Comput.* 38, 608–628 (2008)
14. Emek, Y., Peleg, D.: Approximating minimum max-stretch spanning trees on unweighted graphs. *SIAM J. Comput.* 38, 1761–1781 (2008)
15. Fekete, S.P., Kremer, J.: Tree spanners in planar graphs. *Discrete Appl. Math.* 108, 85–103 (2001)
16. Gavoille, C., Katz, M., Katz, N.A., Paul, C., Peleg, D.: Approximate Distance Labeling Schemes. In: Meyer auf der Heide, F. (ed.) ESA 2001. LNCS, vol. 2161, pp. 476–488. Springer, Heidelberg (2001)

17. Gavril, F.: The intersection graphs of subtrees in trees are exactly the chordal graphs. *J. Comb. Theory (B)* 16, 47–56 (1974)
18. Gilbert, J.R., Rose, D.J., Edenbrandt, A.: A separator theorem for chordal graphs. *SIAM J. Algebraic Discrete Methods* 5, 306–313 (1984)
19. Liebchen, C., Wünsch, G.: The zoo of tree spanner problems. *Discrete Appl. Math.* 156, 569–587 (2008)
20. Lokshtanov, D.: On the complexity of computing tree-length. *Discrete Appl. Math.* 158, 820–827 (2010)
21. Peleg, D.: Low Stretch Spanning Trees. In: Diks, K., Rytter, W. (eds.) *MFCS 2002*. LNCS, vol. 2420, pp. 68–80. Springer, Heidelberg (2002)
22. Peleg, D., Reshef, E.: Low complexity variants of the arrow distributed directory. *J. Comput. System Sci.* 63, 474–485 (2001)
23. Peleg, D., Tandler, D.: Low stretch spanning trees for planar graphs, Tech. Report MCS01-14, Weizmann Science Press of Israel, Israel (2001)
24. Makowsky, J.A., Rotics, U.: Optimal spanners in partial k -trees, manuscript
25. Robertson, N., Seymour, P.D.: Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms* 7, 309–322 (1986)
26. <http://www.cs.kent.edu/~dragan/papers/Approx-tree-spanner-FullVers.pdf>

Approximating the Closest Vector Problem Using an Approximate Shortest Vector Oracle

Chandan Dubey* and Thomas Holenstein

Institute for Theoretical Computer Science
ETH Zurich

chandan.dubey@inf.ethz.ch
thomas.holenstein@inf.ethz.ch

Abstract. We give a polynomial time Turing reduction from the $\gamma^2\sqrt{n}$ -approximate closest vector problem on a lattice of dimension n to a γ -approximate oracle for the shortest vector problem. This is an improvement over a reduction by Kannan, which achieved $\gamma^2 n^{\frac{3}{2}}$.

1 Introduction

A *lattice* is the set of all integer combinations of n linearly independent vectors $\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n$ in \mathbb{R}^m . These vectors are also referred to as a *basis* of the lattice. The *successive minima* $\lambda_i(\mathbb{L})$ (where $i = 1, \dots, n$) for the lattice \mathbb{L} are among the most fundamental parameters associated to a lattice. The value $\lambda_i(\mathbb{L})$ is defined as the smallest r such that a sphere of radius r centered around the origin contains at least i linearly independent lattice vectors. Lattices have been investigated by computer scientists for a few decades after the discovery of the LLL algorithm [14]. More recently, Ajtai [1] showed that lattice problems have a very desirable property for cryptography: they exhibit a worst-case to average-case reduction.

We now describe some of the most fundamental and widely studied lattice problems. Given a lattice \mathbb{L} , the γ -approximate shortest vector problem (γ -SVP for short) is the problem of finding a non-zero lattice vector of length at most $\gamma\lambda_1(\mathbb{L})$. Let the minimum distance of a point $\mathbf{t} \in \mathbb{R}^m$ from the lattice \mathbb{L} be denoted by $\mathbf{d}(\mathbf{t}, \mathbb{L})$. Given a lattice \mathbb{L} and a point $\mathbf{t} \in \mathbb{R}^m$, the γ -approximate closest vector problem or γ -CVP for short is the problem of finding a $\mathbf{v} \in \mathbb{L}$ such that $\|\mathbf{v} - \mathbf{t}\| \leq \gamma\mathbf{d}(\mathbf{t}, \mathbb{L})$.

Besides the search version just described, CVP and SVP also have a gap version. The problem $\text{GapCVP}_\gamma(\mathbf{B}, \mathbf{t})$ asks the distance of \mathbf{t} from the lattice $\mathbb{L}(\mathbf{B})$ within a factor of γ , and $\text{GapSVP}_\gamma(\mathbf{B})$ asks for $\lambda_1(\mathbf{B})$ within a factor of γ . This paper deals with the search version described above.

The problems CVP and SVP are quite well studied. The **Gap** versions of the problems are arguably easier than their **search** counterparts. We know that CVP

* Chandan Dubey is partially supported by the Swiss National Science Foundation (SNF), project no. 200021-132508.

and SVP can be solved exactly in deterministic $2^{O(n)}$ time [184]. In polynomial time they can be approximated within a factor of $2^{n(\log \log n)/\log n}$ using LLL [14] and subsequent improvements by Schnorr [21] and Micciancio et. al. [18] (for details, see the book by Micciancio and Goldwasser [9]). On the other hand, it is known that there exists $c > 0$, such that no polynomial time algorithm can approximate GapCVP and GapSVP within a factor of $n^{c/\log \log n}$, unless $\mathbf{P} = \mathbf{NP}$ or another unlikely scenario is true [710]. The security of hardness of cryptosystems following Ajtai’s seminal work [1] is based on the worst-case hardness of $\tilde{O}(n^2)$ -GapSVP [20,19,15]. In the hardness area, CVP is much more understood than SVP. For example, as opposed to CVP, until now all known NP-hardness proofs for SVP [2,17,13,10] are randomized. A way to prove deterministic hardness of SVP is to prove better reductions from CVP to SVP. This paper aims to study and improve the known relations between these two problems.

A very related result is from Kannan [11], who gave a way to solve \sqrt{n} -CVP using an exact SVP oracle. A generalization of his reduction was used to solve CVP within a factor of $(1 + \epsilon)$ by reducing it to sampling short vectors in the lattice [3]. The improvement from \sqrt{n} to $(1 + \epsilon)$ is achieved mainly because the reduction uses $2^{O(n)}$ time instead of polynomial. It is also known that a γ -CVP oracle can be used to solve γ -SVP [8].

In a survey [12], Kannan gave a different reduction from $\gamma^2 n^{\frac{3}{2}}$ -CVP to γ -SVP. A few words of comparison between our methods and the method used by Kannan [12]. Kannan uses the dual lattice (denoted by $\mathbf{B}^* = (\mathbf{B}^T)^{-1}$, where \mathbf{B}^T is the transpose of the matrix \mathbf{B}) and the transference bound $\lambda_1(\mathbf{B})\lambda_1(\mathbf{B}^*) \leq n$ to find a candidate close vector. Due to the fact that he applies the SVP oracle on both \mathbb{L} as well as \mathbb{L}^* , he loses an additional factor of n . Our method does not use the dual lattice.

Our contribution: We improve the result by Kannan [12], which shows that $\gamma^2 n^{3/2}$ -CVP can be solved using an oracle to solve γ -SVP, and solve $\gamma^2 \sqrt{n}$ -CVP using the same oracle.

For this, we essentially combine the earlier result by Kannan [11] with a reduction by Lyubashevsky and Micciancio [15], as we explain now in some detail.

Our starting point is the earlier reduction by Kannan, which solves \sqrt{n} -CVP using an exact SVP-oracle. In order to explain our ideas, we first shortly describe his reduction. Given a CVP-instance $\mathbf{B} \in \mathbb{Q}^{m \times n}$, $\mathbf{t} \in \mathbb{R}^m$, Kannan uses the SVP-oracle to find $\lambda_1(\mathbf{B})$. He then creates the new basis $\tilde{\mathbf{B}} = \begin{bmatrix} \mathbf{B} & \mathbf{t} \\ 0 & \alpha \end{bmatrix}$, where he picks α carefully somewhat smaller than $\lambda_1(\mathbf{B})$. Now, if $d(\mathbf{t}, \mathbf{B})$ is significantly smaller than $\lambda_1(\mathbf{B})$ (say, $\lambda_1(\mathbf{B})/3$), then the shortest vector in $\tilde{\mathbf{B}}$ is $\begin{bmatrix} \mathbf{t}^\dagger - \mathbf{t} \\ -\alpha \end{bmatrix}$, where \mathbf{t}^\dagger is the lattice vector closest to \mathbf{t} (i.e., the vector we are trying to find). On the other hand if $d(\mathbf{t}, \mathbf{B})$ is larger than $\lambda_1(\mathbf{B})/3$, then Kannan projects the instance in the direction orthogonal to the shortest vector of \mathbf{B} . This reduces the dimension by 1, and an approximation in the resulting instance can be used to get an

approximation in the original instance, because the projected approximation can be “lifted” to find some original lattice point which is not too far from \mathbf{t} .

We show that in case we only have an approximation oracle for SVP, we can argue as follows. First, if $\mathbf{d}(\mathbf{t}, \mathbf{B}) \leq \frac{\lambda_1(\mathbf{B})}{2\gamma}$, then we have an instance of a so called “Bounded Distance Decoding” problem. By a result of Lyubashevsky and Micciancio [15], this can be solved using the the oracle we assume. In case $\mathbf{d}(\mathbf{t}, \mathbf{B}) > \frac{\lambda_1(\mathbf{B})}{2\gamma}$ we can recurse in the same way as Kannan does. The approximation factor $\gamma^2\sqrt{n}$ comes from this case: lifting a projection after the recursion returns, incurs an error of roughly the half the length of the vector \mathbf{v} which was used to project. Since this \mathbf{v} can have length almost $\gamma\lambda_1(\mathbf{B})$, the length of \mathbf{v} can be almost a factor γ^2 larger than $\mathbf{d}(\mathbf{t}, \mathbf{B})$. The squares of these errors then add up as in Kannan’s reduction, which gives a total approximation factor of $\gamma^2\sqrt{n}$.

We remark that even though we do not know which of the two cases apply, we can simply run both, and then use the better result.

Finally, we would like to mention that to the best of our knowledge there is no published proof that in Kannan’s algorithm [11] the projected bases have a representation which is polynomial in the input size. We show that this is indeed the case. For this, it is essentially enough to use a lemma from [9] which states that the vectors in a Gram-Schmidt orthogonalization have this property.

2 Preliminaries

2.1 Notation

A lattice basis is a set of linearly independent vectors $\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{R}^m$. It is sometimes convenient to think of the basis as an $n \times m$ matrix \mathbf{B} , whose n columns are the vectors $\mathbf{b}_1, \dots, \mathbf{b}_n$. The lattice generated by the basis \mathbf{B} will be written as $\mathbb{L}(\mathbf{B})$ and is defined as $\mathbb{L}(\mathbf{B}) = \{\mathbf{B}x \mid x \in \mathbb{Z}^n\}$. The *span* of a basis \mathbf{B} , denoted as $\text{span}(\mathbf{B})$, is defined as $\{\mathbf{B}y \mid y \in \mathbb{R}^n\}$. We will assume that the lattice is over rationals, i.e., $\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{Q}^m$, and the entries are represented by the pair of numerator and denominator. An *elementary* vector $v \in \mathbb{L}(\mathbf{B})$ is a vector which cannot be written as a non-trivial multiple of another lattice vector.

A *shortest vector* of a lattice is a non-zero vector in the lattice whose ℓ_2 norm is minimal. The length of the shortest vector is $\lambda_1(\mathbf{B})$, where λ_1 is as defined in the introduction. For a vector $\mathbf{t} \in \mathbb{R}^m$, let $\mathbf{d}(\mathbf{t}, \mathbb{L}(\mathbf{B}))$ denote the distance of \mathbf{t} to the closest lattice point in \mathbf{B} . We use \mathbf{t}^\dagger to denote a (fixed) closest vector to \mathbf{t} in $\mathbb{L}(\mathbf{B})$.

For two vectors \mathbf{u} and \mathbf{v} in \mathbb{R}^m , $\mathbf{v}|_{\mathbf{u}}$ denotes the component of \mathbf{v} in the direction of \mathbf{u} i.e., $\mathbf{v}|_{\mathbf{u}} = \frac{\langle \mathbf{v}, \mathbf{u} \rangle}{\langle \mathbf{u}, \mathbf{u} \rangle} \mathbf{u}$. Also, the component of \mathbf{v} in the direction orthogonal to \mathbf{u} is denoted by $\mathbf{v}_{\perp \mathbf{u}}$ i.e., the vector $\mathbf{v} - \mathbf{v}|_{\mathbf{u}}$.

Consider a lattice $\mathbb{L}(\mathbf{B})$ and a vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ in the lattice. Then the projected lattice of $\mathbb{L}(\mathbf{B})$ perpendicular to \mathbf{v} is $\mathbb{L}(\mathbf{B}_{\perp \mathbf{v}}) := \{\mathbf{u}_{\perp \mathbf{v}} \mid \mathbf{u} \in \mathbb{L}(\mathbf{B})\}$. A basis of $\mathbb{L}(\mathbf{B}_{\perp \mathbf{b}_1})$ is given by the vectors $\{\mathbf{b}_{2\perp \mathbf{b}_1}, \dots, \mathbf{b}_{n\perp \mathbf{b}_1}\}$.

For an integer $k \in \mathbb{Z}^+$ we use $[k]$ to denote the set $\{1, \dots, k\}$.

2.2 Lattice Problems

In this paper we are concerned with the following approximation problems, which are parametrized by some $\gamma > 1$.

γ -SVP: Given a lattice basis \mathbf{B} , find a non-zero vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ such that $\|\mathbf{v}\| \leq \gamma \lambda_1(\mathbf{B})$.

γ -CVP: Given a lattice basis \mathbf{B} , and a vector $\mathbf{t} \in \mathbb{R}^m$ find a vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ such that $\|\mathbf{v} - \mathbf{t}\| \leq \gamma d(\mathbf{t}, \mathbf{B})$.

We also use the following promise problems, which are parameterized by some $\gamma > 0$.

γ -BDD: Given a lattice basis \mathbf{B} , and a vector $\mathbf{t} \in \mathbb{R}^m$ with the promise that $d(\mathbf{t}, \mathbb{L}(\mathbf{B})) \leq \gamma \lambda_1(\mathbf{B})$, find a vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ such that $\|\mathbf{v} - \mathbf{t}\| = d(\mathbf{t}, \mathbf{B})$.

γ -uSVP: Given a lattice basis \mathbf{B} with the promise that $\lambda_2(\mathbf{B}) \geq \gamma \lambda_1(\mathbf{B})$, find a non-zero vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ such that $\|\mathbf{v}\| = \lambda_1(\mathbf{B})$ (this makes sense only for $\gamma \geq 1$).

We assume that we have given a γ -SVP oracle, denoted by \mathfrak{O} . When given a set of linearly independent vectors $\mathbf{B} = \{\mathbf{b}_1, \mathbf{b}_2, \dots, \mathbf{b}_n\} \in \mathbb{Q}^{m \times n}$, $\mathfrak{O}(\mathbf{B})$ returns an elementary vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ which satisfies $0 < \|\mathbf{v}\| \leq \gamma \lambda_1(\mathbb{L}(\mathbf{B}))$ (if \mathbf{v} is not elementary then we can find out the multiple and recover the corresponding elementary vector).

3 Some Basic Tools

Given a basis \mathbf{B} and an elementary vector $\mathbf{v} \in \mathbb{L}(\mathbf{B})$, we can in polynomial time find a new basis of $\mathbb{L}(\mathbf{B})$ of the form $\{\mathbf{v}, \mathbf{b}'_2, \dots, \mathbf{b}'_n\}$. To do this we use the following lemma from Micciancio [16] (page 7, Lemma 1), which we specialized somewhat for our needs.

Lemma 1. *There is a polynomial time algorithm $\text{findbasis}(\mathbf{v}, \mathbf{B})$, which, on input an elementary vector \mathbf{v} of $\mathbb{L}(\mathbf{B})$ and a lattice basis $\mathbf{B} \in \mathbb{Q}^{m \times n}$ outputs $\tilde{\mathbf{B}} = (\tilde{\mathbf{b}}_2, \dots, \tilde{\mathbf{b}}_n)$ such that $\mathbb{L}(\mathbf{v}, \tilde{\mathbf{b}}_2, \dots, \tilde{\mathbf{b}}_n) = \mathbb{L}(\mathbf{B})$.*

Lemma 2. *Let $\mathbb{L}(\mathbf{B})$ be a lattice and $\mathbf{v} \in \mathbb{L}(\mathbf{B})$ be a vector in the lattice. If $\mathbb{L}(\mathbf{B}_{\perp \mathbf{v}})$ is the projected lattice of $\mathbb{L}(\mathbf{B})$ perpendicular to \mathbf{v} then $\lambda_i(\mathbf{B}_{\perp \mathbf{v}}) \leq \lambda_{i+1}(\mathbf{B})$, $i \in [n-1]$.*

Proof. Let \mathbf{v}_i be the vector of length $\lambda_i(\mathbf{B})$ such that $\{\mathbf{v}_1, \dots, \mathbf{v}_n\}$ are linearly independent. A set of such vectors exists [9]. If $(\mathbf{v}_i)_{\perp \mathbf{v}} = 0$ then $(\mathbf{v}_{i>1})_{\perp \mathbf{v}} \in \mathbb{L}(\mathbf{B}_{\perp \mathbf{v}})$ and $0 < \|(\mathbf{v}_i)_{\perp \mathbf{v}}\| \leq \|\mathbf{v}_i\|$, proving the lemma. If $(\mathbf{v}_1)_{\perp \mathbf{v}} \neq 0$ then $(\mathbf{v}_1)_{\perp \mathbf{v}} \in \mathbb{L}(\mathbf{B}_{\perp \mathbf{v}})$ and $0 < \|(\mathbf{v}_1)_{\perp \mathbf{v}}\| \leq \|\mathbf{v}_1\|$. We argue in a similar way with $(\mathbf{v}_2)_{\perp \mathbf{v}}$ to prove the lemma for $i > 1$. \square

We use the following reduction from due to Lyubashevsky and Micciancio [15].

Theorem 1. *For any $\gamma \geq 1$, there is a polynomial time oracle reduction from $\text{BDD}_{\frac{1}{2\gamma}}$ to uSVP_{γ} .*

For completeness, we sketch a proof of Theorem 1 in Appendix A.

4 Reducing CVP to SVP

We prove the following theorem:

Theorem 2. *Given a basis $\mathbf{B} \in \mathbb{Q}^{m \times n}$ and a vector $\mathbf{t} \in \mathbb{R}^m$, the problem $\gamma^2 \sqrt{n}$ -CVP is Turing reducible to the problem γ -SVP in time $\text{poly}(n, \log \gamma, \max_i \log \|\mathbf{b}_i\|)$.*

In this section we give the algorithm to prove our theorem, and show that once it terminates, it satisfies the requirements of the theorem. We will show that the algorithm runs in polynomial time in the next section.

The reduction takes as an input a basis $\mathbf{B} \in \mathbb{Q}^{m \times n}$ and a vector $\mathbf{t} \in \mathbb{R}^m$. Recall that the oracle \mathfrak{D} takes as input a basis over \mathbb{Q} and outputs an elementary vector which is a γ -approximation to the shortest vector. The reduction is given in Algorithm 1.

Algorithm 1. CVP(\mathbf{B}, \mathbf{t}) (input: $\mathbf{B} \in \mathbb{Q}^{m \times n}, \mathbf{t} \in \mathbb{Q}^m$)

```

1: if  $n = 1$  then
2:   Let  $\mathbf{b}_1$  be the only column of  $\mathbf{B}$ .
3:   return  $a\mathbf{b}_1$  with  $a \in \mathbb{Z}$  such that  $\|a\mathbf{b}_1 - \mathbf{t}\|$  is minimal.
4: else
5:    $\mathbf{z}_1 \leftarrow \frac{1}{2\gamma}$ -BDD( $\mathbf{B}, \mathbf{t}$ )           (Solve this with calls to  $\mathfrak{D}$  as in Theorem 1)
6:    $\mathbf{v} \leftarrow \mathfrak{D}(\mathbf{B})$ 
7:    $\{\mathbf{b}_2, \dots, \mathbf{b}_n\} \leftarrow \text{LLL}(\text{findbasis}(\mathbf{v}, \mathbf{B}))$ 
8:    $\forall i \in \{2, \dots, n\} : (\mathbf{b}'_i)_{\perp \mathbf{v}} \leftarrow \mathbf{b}_i - \mathbf{b}_i|_{\mathbf{v}}$ 
9:    $\mathbf{B}'_{\perp \mathbf{v}} \leftarrow \{(\mathbf{b}'_2)_{\perp \mathbf{v}}, \dots, (\mathbf{b}'_n)_{\perp \mathbf{v}}\}$ 
10:   $\mathbf{t}'_{\perp \mathbf{v}} \leftarrow \mathbf{t} - \mathbf{t}|_{\mathbf{v}}$ 
11:   $\mathbf{z}'_2 \leftarrow \text{CVP}(\mathbf{B}'_{\perp \mathbf{v}}, \mathbf{t}'_{\perp \mathbf{v}})$ 
12:  Find  $(a_2, \dots, a_n) \in \mathbb{Z}^{n-1}$  such that  $\mathbf{z}'_2 = \sum_{i=2}^n a_i (\mathbf{b}'_i)_{\perp \mathbf{v}}$ 
13:  Find  $a_1 \in \mathbb{Z}$  such that  $\mathbf{z}_2 = a_1 \mathbf{v} + \sum_{i=2}^n a_i \mathbf{b}_i$  is closest to  $\mathbf{t}$ 
14:  return the element of  $\{\mathbf{z}_1, \mathbf{z}_2\}$  which is closest to  $\mathbf{t}$ .
15: end if

```

In line 6, we can simulate an oracle for $\frac{1}{2\gamma}$ -BDD due to Theorem 1, given \mathfrak{D} . In line 7 we run the LLL algorithm on the basis returned by `findbasis`; this is an easy way to ensure that the representation of the basis does not grow too large (cf. the proof of Lemma 5). The optimization problem in line 13 is of course easy to solve: for example, we can find $a'_1 \in \mathbb{R}$ which minimizes the expression and then round a'_1 to the nearest integer.

Theorem 3. *The approximate CVP-solver (Algorithm 1) outputs a vector $\mathbf{z} \in \mathbb{L}(\mathbf{B})$ such that $\|\mathbf{z} - \mathbf{t}\| \leq \gamma^2 \sqrt{n} d(\mathbf{t}, \mathbf{B})$.*

Proof. We prove the theorem by induction on n . For the base case (i.e., $n = 1$) we find the closest vector to \mathbf{t} in a single vector basis. This can be done exactly by finding the correct multiple of the only basis vector that is closest to \mathbf{t} .

When $n > 1$, we see that each run of the algorithm finds two candidates \mathbf{z}_1 and \mathbf{z}_2 . We show that the shorter of the two is an approximation to the closest vector to \mathbf{t} in $\mathbb{L}(\mathbf{B})$ for which

$$\|\mathbf{z} - \mathbf{t}\| \leq \sqrt{n}\gamma^2 \mathbf{d}(\mathbf{t}, \mathbf{B}) \quad (1)$$

We divide the proof in two cases, depending on whether $\mathbf{d}(\mathbf{t}, \mathbf{B}) < \frac{\lambda_1(\mathbf{B})}{2\gamma}$. It is sufficient to show that in each case one of \mathbf{z}_1 or \mathbf{z}_2 satisfies Equation (1).

1. If $\mathbf{d}(\mathbf{t}, \mathbf{B}) < \frac{\lambda_1(\mathbf{B})}{2\gamma}$, the promise of $\frac{1}{2\gamma}$ -BDD is satisfied. Thus, \mathbf{z}_1 satisfies $\|\mathbf{z}_1 - \mathbf{t}\| \leq \mathbf{d}(\mathbf{t}, \mathbf{B})$.
2. If $\mathbf{d}(\mathbf{t}, \mathbf{B}) \geq \frac{\lambda_1(\mathbf{B})}{2\gamma}$ we proceed as in Kannan's proof to show that \mathbf{z}_2 satisfies Equation (1).

By the induction hypothesis, \mathbf{z}'_2 satisfies

$$\|\mathbf{z}'_2 - \mathbf{t}'_{\perp\mathbf{v}}\|^2 \leq (n-1)\gamma^4 \mathbf{d}^2(\mathbf{t}'_{\perp\mathbf{v}}, \mathbf{B}'_{\perp\mathbf{v}})$$

At this point, note first that $\mathbf{t} = \mathbf{t}'_{\perp\mathbf{v}} + \phi\mathbf{v}$ for some $\phi \in \mathbb{R}$. Since also $\sum_{i=2}^n a_i \mathbf{b}_i = \mathbf{z}'_2 + \eta\mathbf{v}$ for some $\eta \in \mathbb{R}$, we can write

$$\begin{aligned} \|\mathbf{z}_2 - \mathbf{t}\|^2 &= \|(a_1\mathbf{v} + \mathbf{z}'_2 + \eta\mathbf{v}) - (\mathbf{t}'_{\perp\mathbf{v}} + \phi\mathbf{v})\|^2 \\ &= \|(a_1 + \eta - \phi)\mathbf{v}\|^2 + \|\mathbf{z}'_2 - \mathbf{t}'_{\perp\mathbf{v}}\|^2 \end{aligned}$$

Since a_1 is chosen such that this expression is minimal we have $|a_1 + \eta - \phi| \leq \frac{1}{2}$, and so

$$\begin{aligned} \|\mathbf{z}_2 - \mathbf{t}\|^2 &\leq \|\mathbf{z}'_2 - \mathbf{t}'_{\perp\mathbf{v}}\|^2 + \frac{\|\mathbf{v}\|^2}{4} \leq \|\mathbf{z}'_2 - \mathbf{t}'_{\perp\mathbf{v}}\|^2 + \frac{\gamma^2 \lambda_1^2(\mathbf{B})}{4} \\ &\leq (n-1)\gamma^4 \mathbf{d}^2(\mathbf{t}'_{\perp\mathbf{v}}, \mathbb{L}(\mathbf{B}_{\perp\mathbf{v}})) + \frac{\gamma^2 4\gamma^2 \mathbf{d}^2(\mathbf{t}, \mathbf{B})}{4} \\ &\leq \gamma^4 n \mathbf{d}^2(\mathbf{t}, \mathbf{B}). \end{aligned}$$

The second last inequality follows from $\lambda_1^2(\mathbf{B}) \leq 4\gamma^2 \mathbf{d}^2(\mathbf{t}, \mathbf{B})$, which holds in this second case. To see the last inequality, note that $\mathbb{L}(\mathbf{B}_{\perp\mathbf{v}})$ is a projection of $\mathbb{L}(\mathbf{B})$ and $\mathbf{t}'_{\perp\mathbf{v}}$ is a projection of \mathbf{t} in the direction orthogonal to \mathbf{v} , and a projection cannot increase the length of a vector.

Thus, in both cases one of \mathbf{z}_1 and \mathbf{z}_2 satisfies the requirements, and so we get the result. \square

5 Analysis of Runtime

In this section, we show that Algorithm 1 runs in polynomial time. Observe first that in each recursive call the number of basis vector reduces by 1. Since all steps are obviously polynomial, it is enough to show that all the vectors generated during the run of the algorithm can be represented in polynomially

many bits in the input size of the top level of the algorithm. For this, we can assume that the original basis vectors $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ are integer vectors. This can be achieved by multiplying them with the product of their denominators. This operation does not increase the bit representation by more than a factor of $\log(mn)$. Assuming that the basis vectors are over integers, a lower bound on the input size can be given by $M = \max\{n, \log(\max_i \|\mathbf{b}_i\|)\}$.

Given a basis $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$, the Gram-Schmidt orthogonalization of \mathbf{B} is $\{\tilde{\mathbf{b}}_1, \dots, \tilde{\mathbf{b}}_n\}$, where $\tilde{\mathbf{b}}_i = \mathbf{b}_i - \sum_{j=1}^{i-1} \mathbf{b}_i|_{\tilde{\mathbf{b}}_j}$. We need the following Lemma from [9].

Lemma 3. [9] *Let $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\}$ be n linearly independent vectors. Define the vectors $\tilde{\mathbf{b}}_i = \mathbf{b}_i - \sum_{j=1}^{i-1} \mathbf{b}_i|_{\tilde{\mathbf{b}}_j}$. Then, the representation of any vector $\tilde{\mathbf{b}}_i$ as a vector of quotients of natural numbers takes at most $\text{poly}(M)$ bits for $M = \max\{n, \log(\max_i \|\mathbf{b}_i\|)\}$.*

Lemma 4. *Let $\mathbf{v}_i, i \in [n]$, be the vector \mathbf{v} generated in the i th level of the recursion in line 6 of Algorithm 1.*

There is a basis $\mathbf{x}_1, \dots, \mathbf{x}_n$ of \mathbf{B} such that the vectors \mathbf{v}_i are given by the Gram-Schmidt orthogonalization of $\mathbf{x}_1, \dots, \mathbf{x}_n$. Furthermore, $\mathbf{x}_1, \dots, \mathbf{x}_n$ as well as $\mathbf{v}_1, \dots, \mathbf{v}_n$ are polynomially representable in M .

Proof. We first find lattice vectors $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{L}(\mathbf{B})$ which satisfy

$$\mathbf{x}_i = \mathbf{v}_i + \sum_{j=1}^{i-1} \delta_j \mathbf{v}_j$$

for some $\delta_j \in [-\frac{1}{2}, \frac{1}{2}]$, and then show that these vectors satisfy the claim of the lemma.

To see that such vectors exist, let \mathbf{B}_j be the basis in the j th level of the recursion of Algorithm 1. Then, we note that given a vector in $\mathbb{L}(\mathbf{B}_j)$ one can find a lattice vector in $\mathbb{L}(\mathbf{B}_{j-1})$ at distance at most $\frac{\|\mathbf{v}_{j-1}\|}{2}$ in the direction of \mathbf{v}_{j-1} or $-\mathbf{v}_{j-1}$. We let \mathbf{x}_i be the vector obtained by doing such a lifting step repeatedly until we have a lattice vector in $\mathbb{L}(\mathbf{B})$.

The vectors $\mathbf{v}_1, \dots, \mathbf{v}_n$ are exactly the Gram-Schmidt orthogonalization of $\mathbf{x}_1, \dots, \mathbf{x}_n$, because

$$\mathbf{v}_i = \mathbf{x}_i - \mathbf{x}_i|_{\mathbf{v}_1} - \mathbf{x}_i|_{\mathbf{v}_2} - \dots - \mathbf{x}_i|_{\mathbf{v}_{i-1}},$$

and so the vectors \mathbf{x}_i must also form a basis of $\mathbb{L}(\mathbf{B})$.

Also, we have for all $i \in [n]$:

$$\begin{aligned} \|\mathbf{x}_i\|^2 &\leq \|\mathbf{v}_i\|^2 + \frac{\|\mathbf{v}_{i-1}\|^2}{4} + \dots + \frac{\|\mathbf{v}_1\|^2}{4} \\ &\leq \sum_{j=1}^i \|\mathbf{v}_j\|^2 \\ &\leq n\gamma^2 \lambda_n^2(\mathbf{B}) \end{aligned} \tag{From Lemma 2}$$

As $\mathbf{x}_1, \dots, \mathbf{x}_n$ are vectors in the integer lattice \mathbf{B} ; $\mathbf{x}_1, \dots, \mathbf{x}_n$ are polynomially representable in M (and $\log \gamma$, but we can assume $\gamma < 2^n$). Coupled with Lemma 3 this completes the proof. \square

Lemma 5. *All vectors which are generated in a run of Algorithm 7 have a representation of size $\text{poly}(M)$ for $M = \max\{n, \log(\max_i \|b_i\|)\}$, in case the entries are represented as quotients of natural numbers.*

Proof. The vectors \mathbf{v}_i which are generated in line 6 at different levels of recursion also have representation of size $\text{poly}(M)$ by Lemma 3. The basis \mathbf{B}_i is LLL reduced and hence it is representable in number of bits which is a fixed polynomial in the shortest vector [14] and hence also \mathbf{v}_i .

The remaining vectors are produced by oracles which run in polynomial time or are small linear combinations of other vectors. \square

We now give a proof of Theorem 2.

Proof. (Theorem 2) Given $\mathbf{B} \in \mathbb{Q}^{m \times n}$ and $\mathbf{t} \in \mathbb{R}^m$ we run Algorithm 1. From Lemma 3 the algorithm returns a vector \mathbf{z} which is a $\gamma^2 \sqrt{n}$ -approximation to the closest vector. Also, from Lemma 5, all vectors in the algorithm have polynomial size representation, and so the algorithm runs in time $\text{poly}(\log \gamma, M)$. \square

Acknowledgements. We thank Divesh Aggarwal and Robin Künzler for numerous helpful discussions, and Daniele Micciancio for useful comments on an earlier version of this paper. We want to thank the anonymous referees for helpful comments. In particular, we thank the referee who pointed out the relevance of Theorem 1 to our work and helped us simplify the proof to its current form.

References

1. Ajtai, M.: Generating hard instances of lattice problems. In: STOC, pp. 99–108 (1996)
2. Ajtai, M.: The shortest vector problem in ℓ_2 is **NP**-hard for randomized reductions. In: STOC, pp. 10–19 (1998)
3. Ajtai, M., Kumar, R., Sivakumar, D.: Sampling Short Lattice Vectors and the Closest Lattice Vector Problem. In: CCC, pp. 53–57 (2002)
4. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: STOC, pp. 266–275 (1998)
5. Blömer, J., Seifert, J.-P.: The complexity of computing short linearly independent vectors and short bases in a lattice. In: STOC, pp. 711–720 (1999)
6. Banaszczyk, W.: New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen* 296, 625–635 (1993)
7. Dinur, I., Kindler, G., Raz, R., Safra, S.: Approximating CVP to within almost-polynomial factors is **NP**-hard. *Combinatorica* 23(2), 205–243 (2003)
8. Goldreich, O., Micciancio, D., Safra, S., Seifert, J.-P.: Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Information Processing Letters* 71(2), 55–61 (1999)
9. Goldwasser, S., Micciancio, D.: *Complexity of lattice problems*. Springer, Heidelberg (2002)

10. Haviv, I., Regev, O.: Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In: STOC (2007)
11. Kannan, R.: Minkowski's convex body theorem and integer programming. *Math. Oper. Res.* 12, 415–440 (1987)
12. Kannan, R.: Algorithmic geometry of numbers. *Annual Review of Computer Science* 2, 231–267 (1987)
13. Khot, S.: Hardness of approximating the shortest vector problem in lattices. *JACM* 52(5), 789–808 (2005)
14. Lenstra, A.K., Lenstra Jr., H.W., Lovasz, L.: Factoring polynomials with rational coefficients. *Mathematische Annalen* 261, 513–534 (1982)
15. Lyubashevsky, V., Micciancio, D.: On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 577–594. Springer, Heidelberg (2009)
16. Micciancio, D.: Efficient reductions among lattice problems. In: SODA, pp. 84–93 (2008)
17. Micciancio, D.: The shortest vector problem is **NP**-hard to approximate within some constant. *SIAM Journal on Computing* 30(6), 2008–2035 (2001)
18. Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In: STOC, pp. 351–358 (2010)
19. Peikert, C.: Public-key cryptosystems from the worst-case shortest vector problem. In: STOC (2009)
20. Regev, O.: New lattice-based cryptographic constructions. *J. ACM* 51(6), 899–942 (2004)
21. Schnorr, C.-P.: A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science* 53(2-3), 201–224 (1987)

A Solving BDD Using a uSVP-oracle

In this appendix we sketch the reduction from $\text{BDD}_{1/2\gamma}$ to uSVP_γ from [15] for completeness. We will assume that $d(\mathbf{t}, \mathbb{L}(\mathbf{B}))$ is known – it is shown in [15] how to avoid this assumption.

Proof. (Theorem 1) Let (\mathbf{B}, \mathbf{t}) be an instance of $\text{BDD}_{\frac{1}{2\gamma}}$ and let $\alpha = d(\mathbf{t}, \mathbb{L}(\mathbf{B})) \leq \frac{\lambda_1(\mathbf{B})}{2\gamma}$. For simplicity we assume that we know α (see [15] for bypassing this). Our goal is to find a vector $\mathbf{t}^\dagger \in \mathbb{L}(\mathbf{B})$ such that $d(\mathbf{t}^\dagger, \mathbf{t}) = \alpha$. We define the new basis

$$\tilde{\mathbf{B}} = \begin{pmatrix} \mathbf{B} & \mathbf{t} \\ \mathbf{0} & \alpha \end{pmatrix}. \quad (2)$$

We will show that in $\tilde{\mathbf{B}}$ the vector $\mathbf{v} := \begin{bmatrix} \mathbf{t}^\dagger - \mathbf{t} \\ -\alpha \end{bmatrix}$ is a γ -unique shortest vector. It is clear that we can recover \mathbf{t}^\dagger , the solution to the BDD problem, when given \mathbf{v} . The length of \mathbf{v} is $\sqrt{2}\alpha$, and so it is enough to show that all other vectors in $\mathbb{L}(\tilde{\mathbf{B}})$, which are not a multiple of \mathbf{v} have length at least $\sqrt{2}\gamma\alpha$. Let us (for the sake of contradiction) assume that there is a vector \mathbf{v}_2 of length at most $\|\mathbf{v}_2\| < \sqrt{2}\gamma\alpha$ which is not a multiple of the vector \mathbf{v} above. We can write \mathbf{v}_2 as

$\mathbf{v}_2 = \begin{bmatrix} \mathbf{u} - a\mathbf{t} \\ -a\alpha \end{bmatrix}$, where $\mathbf{u} \in \mathbb{L}(\mathbf{B})$ and $a \in \mathbb{Z}$. Since \mathbf{v}_2 is not a multiple of \mathbf{v} , it must be that $\mathbf{u} - a\mathbf{t}^\dagger \in \mathbb{L}(\mathbf{B})$ is a non-zero lattice vector. Now, using the triangle inequality, we get

$$\begin{aligned} \|\mathbf{u} - a\mathbf{t}^\dagger\| &\leq \|\mathbf{u} - a\mathbf{t}\| + a\|\mathbf{t} - \mathbf{t}^\dagger\| \\ &= \sqrt{\|\mathbf{v}_2\|^2 - a^2\alpha^2} + a\alpha \\ &< \sqrt{2\alpha^2\gamma^2 - a^2\alpha^2} + a\alpha \\ &\leq 2\alpha\gamma \leq \lambda_1(\mathbf{B}), \end{aligned} \quad (\text{Maximized when } a = \gamma)$$

which is a contradiction. □

Opaque Sets

Adrian Dumitrescu^{1,*}, Minghui Jiang^{2,**}, and János Pach^{3,***}

¹ University of Wisconsin–Milwaukee

dumitres@uwm.edu

² Utah State University

mjiang@cc.usu.edu

³ Ecole Polytechnique Fédérale de Lausanne and City College of New York

pach@cims.nyu.edu

Abstract. The problem of finding “small” sets that meet every straight-line which intersects a given convex region was initiated by Mazurkiewicz in 1916. We call such a set an *opaque set* or a *barrier* for that region. We consider the problem of computing the shortest barrier for a given convex polygon with n vertices. No exact algorithm is currently known even for the simplest instances such as a square or an equilateral triangle. For general barriers, we present a $O(n)$ time approximation algorithm with ratio $\frac{1}{2} + \frac{2+\sqrt{2}}{\pi} = 1.5867\dots$. For connected barriers, we can achieve the approximation ratio $\frac{\pi+5}{\pi+2} = 1.5834\dots$ again in $O(n)$ time. We also show that if the barrier is restricted to the interior and the boundary of the input polygon, then the problem admits a fully polynomial-time approximation scheme for the connected case and a quadratic-time exact algorithm for the single-arc case. These are the first approximation algorithms obtained for this problem.

1 Introduction

The problem of finding small sets that block every line passing through a unit square was first considered by Mazurkiewicz in 1916 [27]; see also [3, 18]. Let C be a convex body in the plane. Following Bagemihl [3], we call a set B an *opaque set* or a *barrier* for C , if it meets all lines that intersect C . A *rectifiable curve* (or arc) is a curve with finite length. A barrier may consist of one or more rectifiable arcs. It does not need to be connected and its portions may lie anywhere in the plane, including the exterior of C ; see [3], [5].

What is the length of the shortest barrier for a given convex body C ? In spite of considerable efforts, the answer to this question is not known even for the simplest instances of C , such as a square, a disk, or an equilateral triangle; see [6], [7, Problem A30], [10], [12], [13], [16, Section 8.11]. The three-dimensional analogue of this problem was raised by Martin Gardner [17]; see also [2, 5]. Some

* Supported in part by NSF grants CCF-0444188 and DMS-1001667.

** Supported in part by NSF grant DBI-0743670.

*** Research partially supported by NSF grant CCF-08-302-72, Swiss NSF grant 200021-125287/1, Hungarian OTKA grant 68398, and by NSA.

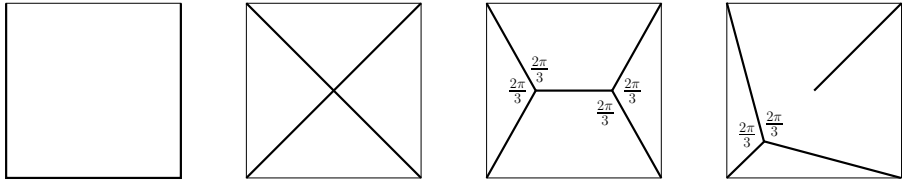


Fig. 1. Four barriers for the unit square. From left to right: 1: single-arc; 2–3: connected; 4: disconnected. The first three from the left have lengths 3, $2\sqrt{2} = 2.8284\dots$, and $1 + \sqrt{3} = 2.7320\dots$. Right: The diagonal segment $[(1/2, 1/2), (1, 1)]$ together with three segments connecting the corners $(0, 1)$, $(0, 0)$, $(1, 0)$ to the point $(\frac{1}{2} - \frac{\sqrt{3}}{6}, \frac{1}{2} - \frac{\sqrt{3}}{6})$ yield a barrier of length $\sqrt{2} + \frac{\sqrt{6}}{2} = 2.639\dots$

entertaining variants of the problem appeared in different forms [20,23,24], for instance: What should a swimmer at sea do in a thick fog if he knows that he is within a mile of a straight shoreline? The shortest known solution resembles the shortest known single-arc barrier for a disk of radius one mile; see [7, Problem A30].

A barrier blocks any line of sight across the region C or detects any ray that passes through it. Motivated by potential applications in guarding and surveillance, the problem of short barriers has been studied by several research communities. Recently, it circulated in internal publications at the Lawrence Livermore National Laboratory. The shortest barrier known for the square is illustrated in Figure 1(right). It is conjectured to be optimal. The best lower bound we know is 2, established by Jones [19].

Related work. The type of curve barriers considered may vary: the most restricted are barriers made from single continuous arcs, then connected barriers, and lastly, arbitrary (possibly disconnected) barriers. For the unit square, the shortest known in these three categories have lengths 3, $1 + \sqrt{3} = 2.7320\dots$ and $\sqrt{2} + \frac{\sqrt{6}}{2} = 2.6389\dots$, respectively. They are depicted in Figure 1. Interestingly, it has been shown by Kawohl [21] that the barrier in Figure 1(right) is optimal in the class of curves with at most two components (there seems to be an additional implicit assumption that the barrier is restricted to the interior of the square). For the unit disk, the shortest known barrier consists of three arcs. See also [12,16].

If instead of curve barriers, we want to find *discrete* barriers consisting of as few points as possible with the property that every line intersecting C gets closer than $\varepsilon > 0$ to at least one of them in some fixed norm, we arrive at a problem raised by László Fejes Tóth [14,15]. The problem has been later coined suggestively as the “point goalie problem” [31]. For instance, if C is an axis-parallel unit square, and we consider the *maximum norm*, the problem was studied by Bárány and Füredi [4], Kern and Wanka [22], Valtr [35], and Richardson and Shepp [31]. Makai and Pach [26] considered another variant of the question, in which we have a larger class of functions to block.

The problem of short barriers has attracted many other researchers and has been studied at length; see also [6,11,25]. Obtaining lower bounds for many of

these problems appears to be notoriously hard. For instance in the point goalie problem for the unit disk (with the Euclidean norm), while the trivial lower bound is $1/\varepsilon$, as given by the opaqueness condition in any one direction, the best lower bound known is only $1.001/\varepsilon$ as established in [31] via a complicated proof.

Our Results. Even we have so little control on the shape or length of optimal barriers, for any convex polygon, barriers whose lengths are somewhat longer can be computed efficiently. Let P be a given convex polygon with n vertices.

1. A (possibly disconnected) segment barrier for P , whose length is at most $\frac{1}{2} + \frac{2+\sqrt{2}}{\pi} = 1.5867\dots$ times the optimal, can be computed in $O(n)$ time.
2. A connected polygonal barrier whose length is at most $\frac{\pi+5}{\pi+2} = 1.5834\dots$ times the optimal can be also computed in $O(n)$ time.
3. For interior single-arc barriers we present an algorithm that finds an optimal barrier in $O(n^2)$ time.
4. For interior connected barriers we present an algorithm that finds a barrier whose length is at most $(1 + \varepsilon)$ times the optimal in polynomial time.

It might be worth mentioning to avoid any confusion: the approximation ratios are for each barrier class, that is, the length of the barrier computed is compared to the optimal length in the corresponding class; and of course these optimal lengths might differ. For instance the connected barrier computed by the approximation algorithm with ratio $\frac{\pi+5}{\pi+2} = 1.5834\dots$ is *not* necessarily shorter than the (possibly disconnected) barrier computed by the approximation algorithm with the larger ratio $\frac{1}{2} + \frac{2+\sqrt{2}}{\pi} = 1.5867\dots$

2 Preliminaries

Definitions and notations. For a polygonal curve γ , let $|\gamma|$ denote the length (or weight) of γ . Similarly, if Γ is a set of polygonal curves, let $|\Gamma|$ denote the total length of the curves in Γ . As usual, when there is no danger of confusion, we also denote by $|A|$ the cardinality of a set A . We call a barrier consisting of segments (or polygonal lines) a *segment barrier*. In order to be able to speak of the *length* $\ell(B)$ of a barrier B , we restrict our attention to barriers that can be obtained as the union of finitely many simple rectifiable curves. We first show (Lemma [1](#)) that the shortest segment barrier is not much longer than the shortest rectifiable one. Due to space limitations we omit the proof of Lemma [1](#).

Lemma 1. *Let B be a barrier of length $\ell(B) < \infty$ for a convex body C in the plane. Then, for any $\varepsilon > 0$, there exists a segment barrier B_ε for C , consisting of finitely many straight-line segments, such that $\ell(B_\varepsilon) \leq \ell(B) + \varepsilon$.*

Denote by $\text{per}(C)$ the perimeter of a convex body C in the plane. The following lemma providing a lower bound on the length of an optimal barrier for C in terms of $\text{per}(C)$, is used in the analysis of our approximation algorithms. Its proof is folklore; see e.g. [\[13\]](#).

Lemma 2. *Let C be a convex body in the plane and let B be a barrier for C . Then the length of B is at least $\frac{1}{2} \cdot \text{per}(C)$.*

Proof. By Lemma 1, we can assume w.l.o.g. that B is a segment barrier. Let $B = \{s_1, \dots, s_n\}$ consist of n segments of lengths $\ell_i = |s_i|$, where $L = |B| = \sum_{i=1}^n \ell_i$. Let $\alpha_i \in [0, \pi)$ be the angle made by s_i with the x -axis. For each direction $\alpha \in [0, \pi)$, the blocking (opaqueness) condition for a convex body C can be written as

$$\sum_{i=1}^n \ell_i |\cos(\alpha - \alpha_i)| \geq W(\alpha), \tag{1}$$

where $W(\alpha)$ is the width of C in direction α . By integrating this inequality over the interval $[0, \pi]$, one gets:

$$\sum_{i=1}^n \ell_i \int_0^\pi |\cos(\alpha - \alpha_i)| \, d\alpha \geq \int_0^\pi W(\alpha) \, d\alpha. \tag{2}$$

According to Cauchy’s surface area formula [28, pp. 283–284], for any planar convex body C , we have

$$\int_0^\pi W(\alpha) \, d\alpha = \text{per}(C). \tag{3}$$

Since

$$\int_0^\pi |\cos(\alpha - \alpha_i)| \, d\alpha = 2,$$

we get

$$2L = \sum_{i=1}^n 2\ell_i \geq \text{per}(C) \Rightarrow L \geq \frac{1}{2} \cdot \text{per}(C), \tag{4}$$

as required. □

For instance, for the square, $\text{per}(C) = 4$, and Lemma 2 immediately gives $L \geq 2$, the lower bound of Jones [19].

A key fact in the analysis of the approximation algorithm is the following lemma. This inequality is implicit in [36]; another proof can be found in [9].

Lemma 3. *Let P be a convex polygon. Then the minimum-perimeter rectangle R containing P satisfies $\text{per}(R) \leq \frac{4}{\pi} \text{per}(P)$.*

Let P be a convex polygon with n vertices. Let $\text{OPT}_{\text{arb}}(P)$, $\text{OPT}_{\text{conn}}(P)$ and $\text{OPT}_{\text{arc}}(P)$ denote optimal barrier lengths of the types arbitrary, connected, and single-arc. Let us observe the following inequalities:

$$\text{OPT}_{\text{arb}}(P) \leq \text{OPT}_{\text{conn}}(P) \leq \text{OPT}_{\text{arc}}(P). \tag{5}$$

We first deal with connected barriers, and then with arbitrary (i.e., possibly disconnected) barriers.

3 Connected Barriers

Theorem 1. *Given a convex polygon P with n vertices, a connected polygonal barrier whose length is at most $\frac{\pi+5}{\pi+2} = 1.5834\dots$ times longer than the optimal can be computed in $O(n)$ time.*

Proof. Consider the following algorithm **A1** that computes a connected barrier consisting of a single-arc; refer to Figure 2. First compute a parallel strip of minimum width enclosing P . Assume w.l.o.g. that the strip is bounded by the two horizontal lines ℓ_1 and ℓ_2 . Second, compute a minimal orthogonal (i.e., vertical) strip enclosing P , bounded by the two vertical lines ℓ_3 and ℓ_4 . Let a, b, c, d, e, f be the six segments on ℓ_3 and ℓ_4 as shown in the figure; here b and e are the two (possibly degenerate) segments on the boundary of P . Let P_1 be the polygonal path (on P 's boundary) between the lower vertices of b and e . Let P_2 be the polygonal path (on P 's boundary) between the top vertices of b and e . Consider the following two barriers for P : B_1 consists of the polygonal path P_1 extended upward at both ends until they reach ℓ_2 . B_2 consists of the polygonal path P_2 extended downward at both ends until they reach ℓ_1 . The algorithm returns the shorter of the two.

Let $p, w,$ and $r,$ respectively, be the perimeter, the width, and the in-radius of P . Clearly

$$|P_1| + |P_2| + |b| + |e| = p.$$

We have the following equalities:

$$\begin{aligned} |B_1| &= |a| + |b| + |P_1| + |e| + |f|, \\ |B_2| &= |c| + |b| + |P_2| + |e| + |d|. \end{aligned}$$

By adding them up we get

$$|B_1| + |B_2| = |P_1| + |P_2| + |b| + |e| + 2w = p + 2w.$$

Hence

$$\min\{|B_1|, |B_2|\} \leq p/2 + w.$$

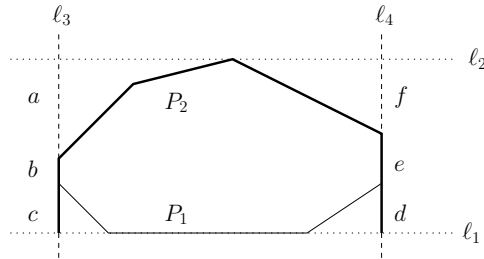


Fig. 2. The approximation algorithm **A1** returns B_2 (in bold lines)

By Blaschke’s Theorem (see e.g. [32]), every planar convex body of width w contains a disk of radius $w/3$. Thus $r \geq w/3$. According to a result of Eggleston [10], the optimal connected barrier for a disk of radius r has length $(\pi+2)r$. It follows that the optimal connected barrier for P has length at least $(\pi+2)w/3$. By Lemma 2, $p/2$ is another lower bound on the optimal solution. Thus the approximation ratio of the algorithm **A1** is at most

$$\begin{aligned} \frac{p/2 + w}{\max\{(\pi + 2)w/3, p/2\}} &= \min \left\{ \frac{p/2 + w}{(\pi + 2)w/3}, \frac{p/2 + w}{p/2} \right\} \\ &= \min \left\{ \frac{3}{2(\pi + 2)} \cdot \frac{p}{w} + \frac{3}{\pi + 2}, 1 + 2 \cdot \frac{w}{p} \right\}. \end{aligned}$$

The equation

$$\frac{3x}{2(\pi + 2)} + \frac{3}{\pi + 2} = 1 + \frac{2}{x}$$

has one positive real root $x_0 = \frac{2(\pi+2)}{3}$. Consequently, the approximation ratio of the algorithm **A1** is at most $1 + \frac{3}{\frac{2(\pi+2)}{3}} = \frac{\pi+5}{\pi+2} = 1.5834\dots$. The algorithm takes $O(n)$ time, since computing the width of P takes $O(n)$ time; see [29,34]. \square

4 Single-arc Barriers

Since **A1** computes a single-arc barrier, and we have $\text{OPT}_{\text{conn}}(P) \leq \text{OPT}_{\text{arc}}(P)$, we immediately get an approximation algorithm with the same ratio $1.5834\dots$ for computing single-arc barriers. One may ask whether this single arc barrier computed by **A1** is optimal (in the class of single arc barriers). We show that this is not the case:

Consider a Reuleaux triangle T of (constant) width 1, with three vertices a, b, c . Now slightly shave the two corners of T at b and c to obtain a convex body T' of (minimum) width $1 - \varepsilon$ along bc . The algorithm **A1** would return a curve of length close to $\pi/2 + 1 = 2.57\dots$, while the optimal curve has length at most $2\pi/3 + 2(1 - \sqrt{3}/2) = 2\pi/3 + 2 - \sqrt{3} = 2.36\dots$. This example shows a lower bound of $1.088\dots$ on the approximation ratio of the algorithm **A1**. Moreover, we believe that the approximation ratio of **A1** is much closer to this lower bound than to $1.5834\dots$

We next present an improved version **B1** of our algorithm **A1** that computes the shortest single-arc barrier of the form shown in Figure 2; see below for details.

Let P be a convex polygon with n sides, and let ℓ be a line tangent to the polygon, i.e., $P \cap \ell$ consists of a vertex of P or a side of P . For simplicity assume that ℓ is the x -axis, and P lies in the closed halfplane $y \geq 0$ above ℓ . Let $T = (\ell_1, \ell_2)$ be a minimal vertical strip enclosing P . Let $p_1 \in \ell_1 \cap P$ and $p_2 \in \ell_2 \cap P$, be the two points of P of minimum y -coordinates on the two vertical lines defining the strip. Let $q_1 \in \ell_1$ and $q_2 \in \ell_2$ be the projections of p_1 and p_2 , respectively, on ℓ . Let $\text{arc}(p_1, p_2) \subset \partial(\text{conv}(P))$ be the polygonal arc connecting p_1 and p_2 on the top boundary of P . The U -curve corresponding to P and ℓ , denoted $U(P, \ell)$ is the polygonal curve obtained by concatenating $q_1p_1,$

$\text{arc}(p_1, p_2)$, and p_2q_2 , in this order. Obviously, for any line ℓ , the curve $U(P, \ell)$ is a single-arc barrier for P . Let $U_{\min}(P)$ be the U -curve of minimum length over all directions $\alpha \in [0, \pi)$ (i.e., lines ℓ of direction α).

We next show that given P , the curve $U_{\min}(P)$ can be computed in $O(n)$ time. The algorithm **B1** is very simple: instead of rotating a line ℓ around P , we fix ℓ to be horizontal, and rotate P over ℓ by one full rotation (of angle 2π). We only compute the lengths of the U -curves corresponding to lines ℓ, ℓ_1, ℓ_2 , supporting one edge of the polygon. The U -curve of minimum length among these is output. There are at most $3n$ such discrete angles (directions), and the length of a U -curve for one such angle can be computed in constant time from the length of the U -curve for the previous angle. The algorithm is similar to the classic rotating calipers algorithm of Toussaint [34], and it takes $O(n)$ time by the previous observation.

To justify its correctness, it suffices to show that if each of the lines ℓ, ℓ_1, ℓ_2 is incident to only one vertex of P , then the corresponding U -curve is not minimal. Due to space limitations we omit the proof of Lemma 4.

Lemma 4. *Let P be a convex polygon tangent to a line ℓ at a vertex $v \in P$ only, and tangent to ℓ_1 and ℓ_2 at vertices p_1 and p_2 only. Then the corresponding U -curve $U(P, \ell)$ is not minimal.*

We thus conclude this section with the following result.

Theorem 2. *Given a convex polygon P with n vertices, the single-arc barrier (polygonal curve) $U_{\min}(P)$ can be computed in $O(n)$ time.*

Obviously, the single-arc barrier computed by **B1** is not longer than that computed by **A1**, so the approximation ratio of the algorithm **B1** is also bounded by $\frac{\pi+5}{\pi+2} = 1.5834\dots$ One may ask again whether this single arc barrier computed by **B1** is optimal (in the class of single arc barriers). We can show again that this is not the case (details omitted here).

5 Arbitrary Barriers

Theorem 3. *Given a convex polygon P with n vertices, a (possibly disconnected) segment barrier for P , whose length is at most $\frac{1}{2} + \frac{2+\sqrt{2}}{\pi} = 1.5867\dots$ times longer than the optimal can be computed in $O(n)$ time.*

Proof. Consider the following algorithm **A2** which computes a (generally disconnected) barrier. First compute a minimum-perimeter rectangle R containing P ; refer to Figure 3. Let $a, b, c, d, e, f, g, h, i, j, k, l$ be the 12 segments on the boundary of R as shown in the figure; here b, e, h and k are (possibly degenerate) segments on the boundary of P contained in the left, bottom, right and top side of R . Let $P_i, i = 1, 2, 3, 4$ be the four polygonal paths on P 's boundary, connecting these four segments as shown in the figure.

Consider four barriers for P , denoted B_i , for $i = 1, 2, 3, 4$. B_i consists of the polygonal path P_i extended at both ends on the corresponding rectangle sides,

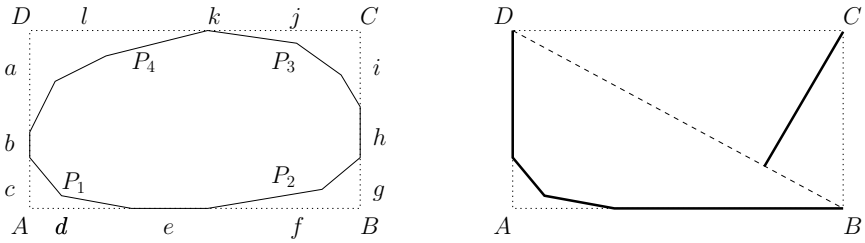


Fig. 3. The approximation algorithm **A2**

and the height from the opposite rectangle vertex in the complementary right angled triangle; see Figure 3(right). The algorithm returns the shortest of the four barriers. Let h_A, h_B, h_C, h_D denote the four heights. We have $|h_A| = |h_B| = |h_C| = |h_D|$ and the following other length equalities:

$$\begin{aligned} |B_1| &= |a| + |b| + |P_1| + |e| + |f| + |h_A|, \\ |B_2| &= |d| + |e| + |P_2| + |h| + |i| + |h_B|, \\ |B_3| &= |g| + |h| + |P_3| + |k| + |l| + |h_C|, \\ |B_4| &= |j| + |k| + |P_4| + |b| + |c| + |h_D|. \end{aligned}$$

By adding them up we get

$$\begin{aligned} \sum_{i=1}^4 |B_i| &= (|b| + |e| + |h| + |k| + \sum_{i=1}^4 |P_i|) + (|a| + \dots + |k|) \\ &\quad + (|h_A| + |h_B| + |h_C| + |h_D|) = \text{per}(P) + \text{per}(R) + 4|h_A|. \end{aligned} \tag{6}$$

Expressing the rectangle area in two different ways yields $|h_A| = \frac{xy}{\sqrt{x^2+y^2}}$, where x and y are the lengths of the two sides of R . By Lemma 3 we have

$$\text{per}(R) = 2(x + y) \leq \frac{4}{\pi} \text{per}(P).$$

Under this constraint, $|h_A|$ is maximized for $x = y = \frac{\text{per}(P)}{\pi}$, namely

$$|h_A| \leq \frac{\text{per}(P)}{\pi\sqrt{2}} \Rightarrow 4|h_A| \leq \frac{2\sqrt{2}}{\pi} \text{per}(P).$$

Hence from (6) we deduce that

$$\min_i |B_i| \leq \frac{1}{4} \left(1 + \frac{4}{\pi} + \frac{2\sqrt{2}}{\pi} \right) \text{per}(P).$$

Recall that $\text{per}(P)/2$ is a lower bound on the weight of an optimal solution. The ratio between the length of the solution and the lower bound on the optimal solution is

$$\frac{\pi + 4 + 2\sqrt{2}}{2\pi} = \frac{1}{2} + \frac{2 + \sqrt{2}}{\pi} = 1.5867\dots$$

Consequently, the approximation ratio of the algorithm **A2** is $\frac{1}{2} + \frac{2+\sqrt{2}}{\pi} = 1.5867\dots$. The algorithm takes $O(n)$ time, since computing the minimum-perimeter rectangle containing P takes $O(n)$ time with the standard technique of rotating calipers [29,34]. This completes the proof of Theorem 3. \square

6 Interior-Restricted Versus Unrestricted Barriers

In certain instances, it is infeasible to construct barriers guarding a specific domain outside the domain (which presumably belongs to someone else). We call such barriers constrained to the interior and the boundary of the domain, *interior-restricted*, or just *interior*, and all others *unrestricted*. For example, all four barriers for the unit square illustrated in Figure 1 are interior barriers.

In the late 1980s, Akman [1] soon followed by Dublish [8] had reported algorithms for computing a minimum interior-restricted barrier of a given convex polygon (they refer to such a barrier as an *opaque minimal forest* of the polygon). Both algorithms however have been shown to be incorrect by Shermer [33] in 1991. He also proposed (conjectured) a new exact algorithm instead, but apparently, so far no one succeeded to prove its correctness. To the best of our knowledge, the computational complexity of computing a shortest barrier (either interior-restricted or unrestricted) for a given convex polygon remains open.

Next we show that a minimum *connected interior* barrier for a convex polygon can be computed efficiently:

Theorem 4. *Given a convex polygon P , a minimum Steiner tree of the vertices of P forms a minimum connected interior barrier for P . Consequently, there is a fully polynomial-time approximation scheme for finding a minimum connected interior barrier for a convex polygon.*

Proof. Let B be an optimal barrier. For each vertex $v \in P$, consider a line ℓ_v tangent to P at v , such that $P \cap \ell_v = \{v\}$. Since B lies in P , ℓ_v can be only blocked by v , so $v \in B$. Now since B is connected and includes all vertices of P , its length is at least that of a minimum Steiner tree of P , as claimed. Recall that the minimum Steiner tree problem for n points in the plane in convex position admits a fully polynomial-time approximation scheme that achieves an approximation ratio of $1 + \epsilon$ and runs in time $O(n^6/\epsilon^4)$ for any $\epsilon > 0$ [30]. \square

A minimum *single-arc interior* barrier for a convex polygon can be also computed efficiently. As it turns out, this problem is equivalent to that of finding a shortest traveling salesman path (i.e., Hamiltonian path) for the n vertices of the polygon.

Theorem 5. *Given a convex polygon P , a minimum Hamiltonian path of the vertices of P forms a minimum single-arc interior barrier for P . Consequently, there is an $O(n^2)$ -time exact algorithm for finding a minimum single-arc interior barrier for a convex polygon with n vertices.*

Proof. The same argument as in the proof of Theorem 4 shows that any interior barrier for P must include all vertices of P . By the triangle inequality, the optimal

single-arc barrier visits each vertex exactly once. Thus a minimum Hamiltonian path of the vertices forms a minimum single-arc interior barrier.

We now present a dynamic programming algorithm for finding a minimum Hamiltonian path of the vertices of a convex polygon. Let $\{v_0, \dots, v_{n-1}\}$ be the n vertices of the convex polygon in counter-clockwise order; for convenience, the indices are modulo n , e.g., $v_n = v_0$. Denote by $\text{dist}(i, j)$ the Euclidean distance between the two vertices v_i and v_j . For the subset of vertices from v_i to v_j counter-clockwise along the polygon, denote by $S(i, j)$ the minimum length of a Hamiltonian path starting at v_i , and denote by $T(i, j)$ the minimum length of a Hamiltonian path starting at v_j . Note that a minimum Hamiltonian path must not intersect itself. Thus the two tables S and T can be computed by dynamic programming with the base cases

$$S(i, i + 1) = T(i, i + 1) = \text{dist}(i, i + 1)$$

and with the recurrences

$$\begin{aligned} S(i, j) &= \min\{\text{dist}(i, i + 1) + S(i + 1, j), \text{dist}(i, j) + T(i + 1, j)\}, \\ T(i, j) &= \min\{\text{dist}(j, j - 1) + T(i, j - 1), \text{dist}(j, i) + S(i, j - 1)\}. \end{aligned}$$

Then the minimum length of a Hamiltonian path on the n vertices is

$$\min_i \min\{\text{dist}(i, i + 1) + S(i + 1, i - 1), \text{dist}(i, i - 1) + T(i + 1, i - 1)\}.$$

The running time of the algorithm is clearly $O(n^2)$. □

Remark. Observe that the unit square contains a disk of radius $1/2$. According to the result of Eggleston mentioned earlier [10], the optimal (not necessarily interior-restricted) connected barrier for a disk of radius r has length $(\pi + 2)r$. This optimal barrier is a single curve consisting of half the disk perimeter and two segments of length equal to the disk radius. It follows that the optimal (not necessarily interior-restricted) connected barrier for the unit square has length at least $(\pi + 2)/2 = \pi/2 + 1 = 2.5707\dots$. Compare this with the current best construction (illustrated in Figure 1, third from the left) of length $1 + \sqrt{3} = 2.7320\dots$. Note that this third construction in Figure 1 gives the optimal connected interior barrier for the square because of Theorem 4. Further note that the first construction in Figure 1 gives the optimal single-arc interior barrier because of Theorem 5.

7 Concluding Remarks

Interesting questions remain open regarding the structure of optimal barriers and the computational complexity of computing such barriers. For instance:

- (1) Does there exist an absolute constant $c \geq 0$ (perhaps zero) such that the following holds? The shortest barrier for any convex polygon with n vertices is a segment barrier consisting of at most $n + c$ segments.

- (2) Is there a polynomial-time algorithm for computing a shortest barrier for a given convex polygon with n vertices?
- (3) Can one give a characterization of the class of convex polygons whose optimal barriers are interior?

In connection with question (2) above, let us notice that the problem of deciding whether a given segment barrier B is an opaque set for a given convex polygon is solvable in polynomial time. Due to space limitations the proof of Theorem 6 is omitted.

Theorem 6. *Given a convex polygon P with n vertices, and a segment barrier B with k segments, there is a polynomial-time algorithm for deciding whether B is an opaque set for P .*

We have presented several approximation and exact algorithms for computing shortest barriers of various kinds, for a given convex polygon. The two approximation algorithms with ratios close to 1.58 probably cannot be improved substantially without either increasing their computational complexity or finding a better lower bound on the optimal solution than that given by Lemma 2. The question of finding a better lower bound is particularly intriguing, since even for the simplest polygons, such as a square, we don't possess any better tool. While much research up to date focused on upper or lower bounds for specific example shapes, obtaining a polynomial time approximation scheme (in the class of arbitrary barriers) for an arbitrary convex polygon is perhaps not out of reach.

References

1. Akman, V.: An algorithm for determining an opaque minimal forest of a convex polygon. *Inform. Process. Lett.* 24, 193–198 (1987)
2. Asimov, D., Gerver, J.L.: Minimum opaque manifolds. *Geom. Dedicata* 133, 67–82 (2008)
3. Bagemihl, F.: Some opaque subsets of a square. *Michigan Math. J.* 6, 99–103 (1959)
4. Bárány, I., Füredi, Z.: Covering all secants of a square. In: Fejes Tóth, G. (ed.) *Intuitive Geometry*. *Colloq. Math. Soc. János Bolyai*, vol. 48, pp. 19–27 (1987)
5. Brakke, K.A.: The opaque cube problem. *Amer. Math. Monthly* 99(9), 866–871 (1992)
6. Croft, H.T.: Curves intersecting certain sets of great-circles on the sphere. *J. London Math. Soc.* 1(2), 461–469 (1969)
7. Croft, H.T., Falconer, K.J., Guy, R.K.: *Unsolved Problems in Geometry*. Springer, New York (1991)
8. Dublish, P.: An $O(n^3)$ algorithm for finding the minimal opaque forest of a convex polygon. *Inform. Process. Lett.* 29(5), 275–276 (1988)
9. Dumitrescu, A., Jiang, M.: Minimum-perimeter intersecting polygons. In: López-Ortiz, A. (ed.) *LATIN 2010*. LNCS, vol. 6034, pp. 433–445. Springer, Heidelberg (2010)
10. Eggleston, H.G.: The maximal in-radius of the convex cover of a plane connected set of given length. *Proc. London Math. Soc.* 45(3), 456–478 (1982)
11. Erdős, P., Pach, J.: On a problem of L. Fejes Tóth. *Discrete Math.* 30(2), 103–109 (1980)

12. Faber, V., Mycielski, J.: The shortest curve that meets all the lines that meet a convex body. *Amer. Math. Monthly* 93, 796–801 (1986)
13. Faber, V., Mycielski, J., Pedersen, P.: On the shortest curve that meets all the lines which meet a circle. *Ann. Polon. Math.* 44, 249–266 (1984)
14. Fejes Tóth, L.: Exploring a planet. *Amer. Math. Monthly* 80, 1043–1044 (1973)
15. Fejes Tóth, L.: Remarks on a dual of Tarski’s plank problem. *Mat. Lapok.* 25, 13–20 (1974)
16. Finch, S.R.: *Mathematical Constants*. Cambridge University Press, Cambridge (2003)
17. Gardner, M.: The opaque cube problem. *Cubism for Fun* 23, 15 (1990)
18. Gupta, H.M.S., Mazumdar, N.C.B.: A note on certain plane sets of points. *Bull. Calcutta Math. Soc.* 47, 199–201 (1955)
19. Jones, R.E.D.: Opaque sets of degree α . *Amer. Math. Monthly* 71, 535–537 (1964)
20. Joris, H.: Le chasseur perdu dans le foret: une problème de géométrie plane. *Elem. der Mathematik* 35, 1–14 (1980)
21. Kawohl, B.: Some nonconvex shape optimization problems. In: Cellina, A., Ornelas, A. (eds.) *Optimal Shape Design*. LNM, vol. 1740. Springer, Heidelberg (2000)
22. Kern, W., Wanka, A.: On a problem about covering lines by squares. *Discrete Comput. Geom.* 5, 77–82 (1990)
23. Klötzler, R.: Universale Rettungskurven I. *Zeitschrite für Analysis und ihre Anwendungen* 5, 27–38 (1986)
24. Klötzler, R., Pickenhain, S.: Universale Rettungskurven II. *Zeitschrite für Analysis und ihre Anwendungen* 6, 363–369 (1987)
25. Makai Jr., E.: On a dual of Tarski’s plank problem. *Discrete Geometrie* 2, 127–132 (1980)
26. Makai Jr., E., Pach, J.: Controlling function classes and covering Euclidean space. *Stud. Sci. Math. Hung.* 18, 435–459 (1983)
27. Mazurkiewicz, S.: Sur un ensemble fermé, punctiforme, qui rencontre toute droite passant par un certain domaine. *Prace Mat.-Fiz.* 27, 11–16 (1916)
28. Pach, J., Agarwal, P.K.: *Combinatorial Geometry*. John Wiley, New York (1995)
29. Preparata, F., Shamos, M.I.: *Computational Geometry*. Springer, New York (1985)
30. Provan, J.S.: Convexity and the Steiner tree problem. *Networks* 18, 55–72 (1988)
31. Richardson, T., Shepp, L.: The “point” goalie problem. *Discrete Comput. Geom.* 20, 649–669 (2003)
32. Scott, P.R.: A family of inequalities for convex sets. *Bull. Austral. Math. Soc.* 20, 237–245 (1979)
33. Shermer, T.: A counterexample to the algorithms for determining opaque minimal forests. *Inform. Process. Lett.* 40, 41–42 (1991)
34. Toussaint, G.: Solving geometric problems with the rotating calipers. In: *Proc. Mediterranean Electrotechnical Conf., MELECON 1983*, Athens (1983)
35. Valtr, P.: Unit squares intersecting all secants of a square. *Discrete Comput. Geom.* 11, 235–239 (1994)
36. Welzl, E.: The smallest rectangle enclosing a closed curve of length π , manuscript (1993), <http://www.inf.ethz.ch/personal/emo/SmallPieces.html>

Exploring and Triangulating a Region by a Swarm of Robots

Sándor P. Fekete¹, Tom Kamphans^{1,*}, Alexander Kröller¹,
Joseph S.B. Mitchell^{2,**}, and Christiane Schmidt¹

¹ Algorithms Group, Braunschweig Institute of Technology, Germany
{s.fekete,t.kamphans,a.kroeller,c.schmidt}@tu-bs.de

² Department of Applied Mathematics and Statistics, Stony Brook University,
jsbm@ams.stonybrook.edu

Abstract. We consider online and offline problems related to exploring and surveying a region by a swarm of robots with limited communication range. The minimum relay triangulation problem (MRTP) asks for placing a minimum number of robots, such that their communication graph is a triangulated cover of the region. The maximum area triangulation problem (MATP) aims at finding a placement of n robots such that their communication graph contains a root and forms a triangulated cover of a maximum possible amount of area. Both problems are geometric versions of natural graph optimization problems.

The offline version of both problems share a decision problem, which we prove to be NP-hard. For the online version of the MRTP, we give a lower bound of $6/5$ for the competitive ratio, and a strategy that achieves a ratio of 3; for different offline versions, we describe polynomial-time approximation schemes. For the MATP we show that no competitive ratio exists for the online problem, and give polynomial-time approximation schemes for offline versions.

1 Introduction

Exploration and Guarding. Many geometric problems of searching, exploring or guarding are motivated by questions from robot navigation. What strategies should be used for an autonomous robot when dealing with known or unknown environments?

A typical scenario considers an unknown polygonal region P that needs to be fully inspected by one or several robots; in a guarding problem, a (typically known) region needs to be fully covered from a set of guarding positions.

Triangulation is another canonical geometric problem that plays an important role in many contexts. It is an underlying task for many computer graphics approaches and the basis for a huge variety of problems in polygons, e.g., the

* Supported by the ICT Programme of the European Union under contract number ICT-2008-215270 (FRONTS).

** Partially supported by the National Science Foundation (CCF-0729019, CCF-1018388), Metron Aviation, and NASA Ames.

computation of shortest watchman routes in simple polygons. Other contexts include mesh generation, divide-and-conquer algorithms, and even guarding problems: as Fisk [12] showed, a simple argument based on triangulations can be used to show that $\lfloor \frac{n}{3} \rfloor$ guards always suffice to guard any simple polygon with n vertices. Hence, triangulation has been intensively studied. Moreover, classical surveying relies on triangulation, which makes it possible to compute geographic coordinates with high precision by simple trigonometry.

In this paper, we present a number of geometric problems and solutions motivated by exploration and guarding of a region by a large swarm of robots. We consider a static sensor network that needs to react to different scenarios by adding further mobile sensors, e.g. sensor nodes attached to mobile robots. We have experimented with actual robot swarms, consisting of Roombas controlled by iSense sensor nodes, which in turn communicate using IEEE 802.15.4 radios. Localization based on signal strength turned out to be completely infeasible indoors, due to ranging errors well exceeding a factor of 10. However, we are convinced that we can steer a robot through a triangle, making it leave through a designated side. This is done by a simple strategy that tries to increase the signal strength to exactly two vertices. If we have a triangulated environment, we can use the connectivity information as a rough localization map, and navigate robots by switching from triangle to triangle.

We consider online problems (in which the region is unknown) and offline problems (in which the region is known). Another distinction arises from minimizing the number of relays necessary to completely cover and triangulate a region (Minimum Relay Triangulation Problem (MRTP)), or by maximizing the covered subregion for a given number of robots (Maximum Area Triangulation Problem (MATP)). We use the terms *robots* and *relays* synonymously; see Section 2 for more precise details.

For the MRTP we ask for complete coverage of the polygon. Hence, relays must be located at all vertices and the polygon has to be fully triangulated. The knowledge of necessary positions makes an actual placement easier. On the other hand, in combination with the edge lengths restriction, it complicates an NP-hardness proof.

Related Work. Hoffmann et al. [13] presented a 26.5-competitive strategy for the online exploration of simple polygons with unlimited vision. Icking et al. [15] and Fekete et al. [11] considered exploration with limited and time-discrete vision, respectively. Exploration with both limited and time-discrete vision is presented by Fekete et al. [10]. Placing stationary guards was first considered by Chvátal [6], see also O'Rourke [19].

Classical triangulation problems (see, e.g., [19]) ask for a triangulation of all vertices of a polygon, but allow arbitrary length of the edges in the triangulation. This differs from our problem, in which edge lengths are bounded by communication length. Triangulations with shape constraints for the triangles and the use of Steiner points are considered in mesh generation, see for example the survey by Bern and Eppstein [3].

The problem of placing a minimum number of relays with limited communication range in order to achieve a connected network (a generalization of the classical Steiner tree problem) has been considered by Efrat et al. [8], who gave a number of approximation results for the offline problem (a 3.11-approximation for the one-tier version and a PTAS for the two-tier version, that does allow for the use of relays only, of this problem); see the survey [7] for related problems.

For robot swarms, Hsiang et al. [14] consider the problem of dispersing a swarm of simple robots in a cellular environment, minimizing the time until every cell is occupied by a robot. For the case of a single door, Hsiang et al. present algorithms with optimal makespan. For k doors a $\Theta(\log(k + 1))$ -competitive algorithm is given. A similar problem from a more practical view was solved by McLurkin and Smith [17].

Instead of considering simple robots for certain tasks, another approach is to consider the minimal necessary capabilities that allow for a certain task, Suri et al. [20] and Brunner et al. [5] classified different robot models along these lines.

Some work has been done on *budget problems*, optimization problems with a hard limit on the total cost. For example, Blum et al. [4] presented the problem of finding a path in a graph with edge costs and vertex rewards, that maximizes the collected reward while keeping the cost below a fixed limit and give a constant factor approximation. See also Averbuch et al. [2].

Our Results are as follows.

- We show that the offline versions of MRTP and MATP are NP-hard.
- For the online MRTP, we give a lower bound of 6/5 for the competitive ratio.
- We give an online strategy for the MRTP with a competitive ratio of 3.
- For an offline version of the MRTP, we give a polynomial-time approximation scheme (PTAS).
- For the online MATP, we show that no strategy can achieve a constant competitive ratio.
- For an offline version of the MATP, we give a polynomial-time approximation scheme (PTAS).

It should be noted that the results for the offline versions provide approximation schemes for *vertex-based* and *area-based* cost functions, whereas classical approximation schemes for geometric optimization problems (such as the ones developed in [11, 18]) focus on *length-based* cost functions.

The rest of this paper is organized as follows. Section 2 presents basic definitions and preliminaries. Section 3 sketches the hardness proof for the offline problems. Section 4 considers the online MRTP, while Section 5 gives a polynomial time approximation scheme (PTAS) for the offline version. Section 6 shows that no constant competitive ratio for the online MATP exists. We conclude in Section 7. For lack of space a description of the PTAS for the OMATP will be given in the full version of this paper.

2 Notation and Preliminaries

We are given a polygon (an n -gon) P . Every robot in the swarm has a (circular) communication range r . Within this range, perception of and communication with other robots is possible. For the ease of description we assume that r is equal to 1 (and scale the polygon accordingly).

In the offline *Minimum Relay Triangulation Problem* (MRTP), we are given the n -gon P and a point $z \in P$, and the goal is to compute a set, R (with $z \in R$), of relays within P such that there exists a (Steiner) triangulation of P whose vertex set is exactly the set R and whose edges are each of length at most 1. Necessarily, R contains the set V of n vertices of P . The objective is to minimize the number of relays. We let R_{OPT} denote an optimal (minimum-cardinality) set of relays and let T_{OPT} denote a corresponding optimal triangulation of P using an optimal set of relays; with slight abuse of notation, we also use R_{OPT} to denote the cardinality of the set. For convenience, we refer to a triangulation whose edges are each of length at most 1 as a *unit-triangulation*. The triangulation must not contain edges crossing the boundary of P , reflecting the impossibility of communicating through walls. Thus, the triangulation contains all vertices of P , plus intermediate points. The latter are needed as edges in the triangulation must not have a length exceeding 1.

In the offline *Maximum Area Triangulation Problem* (MATP), we are given the n -gon P and a point $z \in P$, and a budget, k , of relays. The goal is to compute a set, R (with $z \in R$), of $k = |R|$ relays within P such that there exists a connected (Steiner) unit-triangulation within P covering the maximum possible area. Let R_{OPT} denote an optimal set of relays, T_{OPT} the optimal triangulation, and A_{OPT} the total area of T_{OPT} . In some situations, two natural assumptions, rooted in the robots finite size, come into play: the region may be assumed to be free of bottlenecks that are too narrow for robots, and we may already have a discrete set of candidate relay locations.

For the online versions (OMRTP and OMATP), the polygon P is unknown. Each relay may move through the area, and has to decide on a new location for a vertex of the triangulation while still within reach of other relays. Once it has stopped, it becomes part of the static triangulation, allowing other relays to extend the exploration and the triangulation. This is motivated by our application, where it is desirable to partially fix the triangulation as it is constructed, to begin location services in this area even if the polygon is not fully explored yet. This is a crucial property if we assume a huge area that is explored over long times. More precisely, also for the OMRTP we are given the n -gon P and a point $z \in P$, and the goal is to compute a set, R , of relays within P such that there exists a (Steiner) triangulation of P whose vertex set is exactly the set R and whose edges are each of length at most 1. The relays move into the polygon, starting from z . A relay extending the yet established subset $R' \subset R$ must stay within a distance of 1 of at least one relay $p \in R$. Once it fixed its position it will not move again. No non-triangle edges are allowed in the final construction. For the OMRTP we let R_{OPT} denote the number of relays used by the optimum, for the OMATP A_{OPT} denotes the area covered by the optimum.

3 NP-Hardness

Theorem 1. *The Minimum Relay Triangulation Problem (MRTP) is NP-hard, even without a discrete set of candidate locations.*

Proof Sketch. A complete proof is omitted for lack of space. The proof is based on a reduction of the NP-hard problem **Planar 3SAT**, a special case of 3SAT in which the variable-clause incidence graph H is planar. In particular, there is a rectilinear configuration of the set of variables in a straight line, with the clauses above and below them; see [16]. This layout is represented by a polygon. The different components (for the clauses, the edges and the variables) can be triangulated using two different configurations, corresponding to truth settings for the variables that may or may not satisfy the respective clause. See Figure 1(a) for an example of a clause gadget, and (b) for a variable gadget: the edge corridors of three variables end in the triangular gadget. The boundary is shown in bold black, all other lines are used only to highlight certain distances. A truth setting satisfying the clause is indicated by black squares, a setting not satisfying the clause as black framed gray squares. In case at least one variable satisfies the clause, 3 interior relays are sufficient for the clause component. If all variables do not satisfy the clause, 4 interior relays are required for the clause component.

A considerable number of further technical issues need to be resolved to complete the overall proof. Among them are detailed constructions for corridors connecting the gadgets, which implement the logical structure of the gadgets, while still allowing careful book-keeping for all the involved angles and distances. A full account of these details is given in the full paper.

Using the same construction as in Theorem 1, we can further conclude:

Theorem 2. *The Maximum Area Triangulation Problem (MATP) is NP-hard, even without a discrete set of candidate locations.*

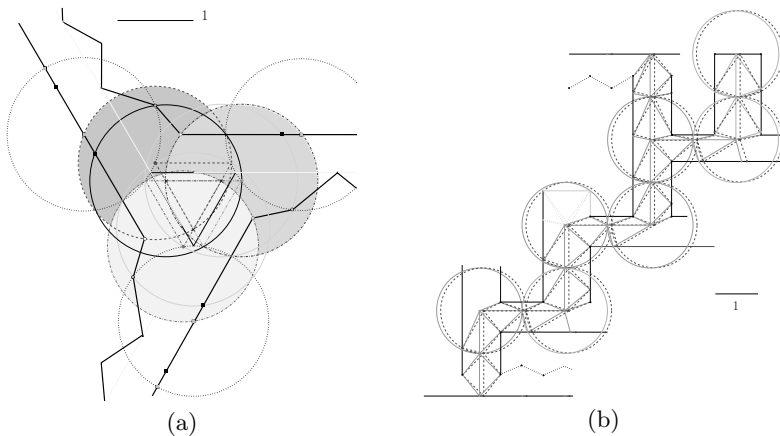


Fig. 1. Polygonal gadgets for a clause (a) and a variable (b). Circles have radius 1

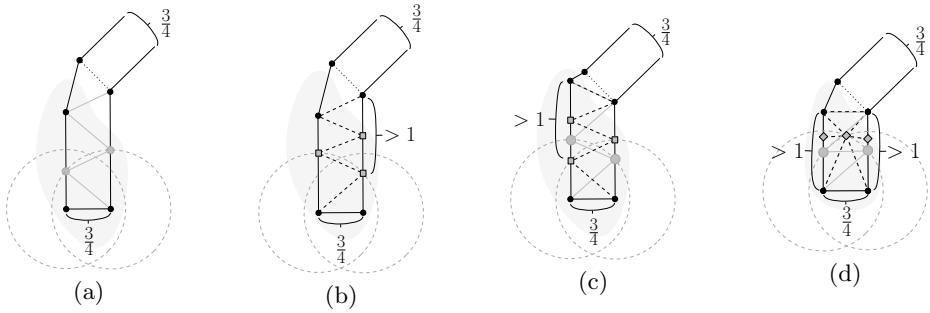


Fig. 2. A lower bound for the OMRTP. Black dots indicate polygon vertices, i.e., mandatory relays; grey disks indicate an optimal solution, while grey squares indicate relays placed by an online strategy.

4 Online Minimum Relay Triangulation

We give a lower bound of $6/5$ and present a 3-competitive online strategy for the OMRTP, improving both values we gave in the informal workshop paper [9].

Lower Bound. For the lower bound we use a polygonal corridor of width $3/4$. For a complete triangulation, relays must be placed at the vertices, i.e., the position of the first two relays is fixed.

In case the algorithm places the next relay on the right boundary, the polygonal corridor will turn out to look like in Figure 2(a). We need to determine the number of relays up to the two relays connected by the dotted edge (in the area indicated by the light gray shape in Figure 2), those build the two fixed relays of the next polygonal pieces. The optimum needs 5 relays. The distance of the relay placed by the algorithm on the right boundary to the next vertex is larger than 1, thus, the algorithm uses 6 relays; see Figure 2(b). In case the algorithm locates the next relay on the left boundary, the polygonal corridor turns out to look like in Figure 2(c). If, on the other hand, the algorithm places the next relay in the center, the polygonal corridor turns out to look like in Figure 2(d). In both cases with an optimum of 5, and an online solution of 6.

The construction we presented results in the next component connected in 45° to the right. Constructions for a connection within a 45° angle to the left are done analogously (mirrored constructions)—resulting in reflected components. The additional relays ensure that a final corridor of $3/4$ is achieved again. Thus, we can iterate the construction. We alternate between the components shown in Figure 2 and the reflected components to avoid a self-overlapping polygon. We conclude

Theorem 3. *No deterministic algorithm for the online minimum relay triangulation problem can be better than $\frac{6}{5}$ -competitive.*

Online Triangulation. In the following, we describe our algorithm for the online minimum relay triangulation problem. Our construction is based on two

components that are glued together into one triangulation: (i) following the boundary of P and (ii) triangulating the interior.

For (i) we place relays within distance 1 along the boundary and on vertices; interior boundaries are dealt with in a similar manner once they are encountered. Let b_{ALG} be the number of relays used in this step, and b_{OPT} the number of relays placed on the boundary by an optimal solution. As any triangulation needs to establish edges along all edges of the polygon P , and the maximum distance of relays is $r = 1$, we conclude:

Lemma 1. $b_{\text{ALG}} \leq b_{\text{OPT}}$.

For (ii), triangulating the interior of P , we build an overlay with an arbitrarily oriented triangular unit grid from our starting point. Whenever we cannot continue this grid but are able to place a relay with distance 1 to all existing interior relays, we do so (and resume the grid construction when possible). Let i_{ALG} be the number of relays used in this step.

Lemma 2. *For an optimal solution for the MRTP with b_{OPT} relays located on the boundary and i_{OPT} located in the interior, the number Δ_{OPT} of triangles in an optimal triangulation satisfies $\Delta_{\text{OPT}} = 2 \cdot i_{\text{OPT}} + b_{\text{OPT}} - 2$.*

The proof relies on accounting for interior angles. Comparing grid relays and optimal triangles, we conclude (see proof in the full version of this paper)

Lemma 3. $i_{\text{ALG}} \leq \Delta_{\text{OPT}}$.

Having positioned $b_{\text{ALG}} + i_{\text{ALG}}$ relays, we are left with the task of keeping the explored region triangulated. Whenever we encounter an untriangulated cell bounded by a number of connections between positioned relays, we use additional relays; let their total number be c_{ALG} . We claim that:

Lemma 4. *In total, $c_{\text{ALG}} \leq b_{\text{OPT}}$ additional relays suffice to ensure an overall triangulation.*

Proof. A full proof is omitted for lack of space. As interior relays of degree 0 and 1 can be triangulated without causing further cost, we consider an edge between two relays ($\{r_1, r_2\}$); see Figure 3. We then give a case distinction of possible relay locations: we distinguish several placements of relays, depending on the location of edges and on p_2 and p_3 being included in the triangular grid. Altogether, every relay on the boundary gets charged at most once, concluding the proof. □

This implies the following theorem.

Theorem 4. *There is a 3-competitive strategy for the online minimum relay triangulation problem in polygons (even with holes).*

The proof is based on the previous lemmas; details are contained in the full version of the paper.

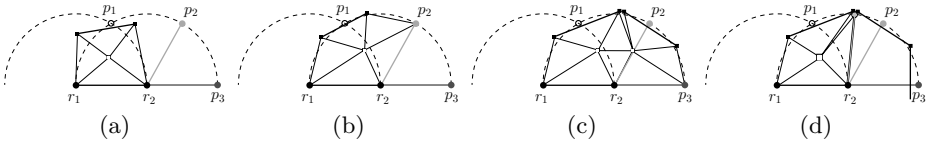


Fig. 3. We consider the black edge $\{r_1, r_2\}$ of a triangulation. The circle position p_1 of the grid is not included. Boundary relays are denoted by black squares, relays placed in phase (ii) as black circles and relays placed to glue the triangulations by black-and-white squares. The resulting triangulations are indicated by fine black lines.

5 Offline Minimum Relay Triangulation

We assume that the relays are restricted to a discrete set, \mathcal{C} , of candidate locations. In particular, we assume that $V \subset \mathcal{C}$ and that, for a fixed (small) parameter $\beta > 0$, \mathcal{C} includes the set of grid points within P , with spacing β , whose points lie at positions $(i\beta, j\beta)$ in the plane, and that \mathcal{C} includes the points where lines of the form $x = i\beta$ and $y = j\beta$ intersect edges of P .

Further, in order to address the most realistic form of the problem, we assume that P is δ -accessible, for some fixed $0 < \delta < 1$: The δ -medial axis is topologically equivalent to the medial axis, and each point of P is within geodesic distance $O(\delta)$ of some disk of radius δ within P . Here, the δ -medial axis is the locus of all centers of medial disks (i.e., disks within P that are in contact with the boundary of P at two or more points) that are of radius δ or greater. Domains P that are δ -accessible can be fully accessed by robots of some fixed size δ : any path within P for a point has a homotopically equivalent path for a disk of radius δ , for all of its length except possible length $O(\delta)$ at each of its ends.

Let $BB(P)$ denote the axis-aligned bounding box of P . Without loss of generality, we can assume that $BB(P)$ has bottom left corner at $(0,0)$; let (x_{max}, y_{max}) denote the upper right corner of $BB(P)$. We let X denote the set of x -coordinates of V , together with the multiples of β , $i\beta$, for $i = 1, 2, \dots, \lfloor x_{max}/\beta \rfloor$; we define the set Y of y -coordinates similarly. An axis-parallel line ℓ is a *cut* if it is defined by the candidate coordinates X (for vertical lines) or Y (for horizontal lines). An axis-aligned rectangle, $\rho \subseteq BB(P)$, is (X, Y) -respecting if its left/right sides are defined by x -coordinates of X and its top/bottom sides are defined by y -coordinates of Y .

Let T denote an arbitrary triangulation of P . The m -span, $\sigma_m(\ell, \rho, T)$, of ℓ with respect to rectangle ρ and triangulation T is defined as follows. Assume that ℓ is vertical; the definition for horizontal cuts is similar. If $\ell \cap \rho$ intersects at most $2m$ edges of T , then the m -span is empty ($\sigma_m(\ell, \rho, T) = \emptyset$). Otherwise, let a be the topmost point in the m th intersection between ℓ and edges of T , from the top of ρ , along $\ell \cap \rho$. (Each intersection between ℓ and edges of T is either a single point, where ℓ crosses an edge, or is an edge e of T , in the case that ℓ contains the (vertical) edge e of T .) Similarly, let b be the bottommost point in the m th intersection between ℓ and edges of T , from the bottom of ρ , along $\ell \cap \rho$. Then, the m -span is defined to be the segment ab : $\sigma_m(\ell, \rho, T) = ab$;

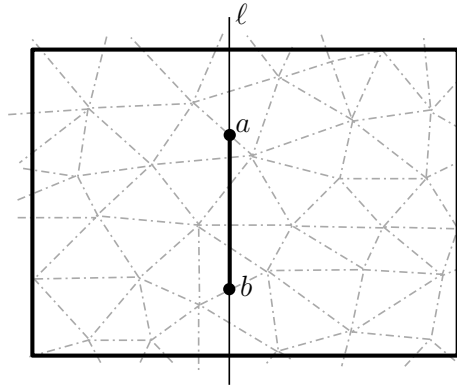


Fig. 4. The m -span (in bold), $\sigma_m(\ell, \rho, T)$, ($m = 2$) of ℓ with respect to rectangle ρ and triangulation T

see Figure 4. Note that $\sigma_m(\ell, \rho, T) \cap P$ is a union of potentially many ($O(n)$) vertical segments along ℓ , all but two of which are vertical chords of P .

We describe a PTAS for the MRTP problem by developing the m -guillotine method [18], with several key new ideas needed to address the triangulation problem. We fix an $\epsilon > 0$ and let $m = \lceil 1/\epsilon \rceil$. Our method employs a structure theorem, which shows that we can transform an arbitrary (Steiner) unit-triangulation T of P into a (Steiner) unit-triangulation T_G that is “ m -guillotine”, with the number of vertices of T_G at most $(1 + \epsilon)$ times the number of vertices of T . The m -guillotine structure is a special recursive structure that allows a dynamic programming algorithm to optimize over all m -guillotine unit-triangulations with vertices in \mathcal{C} . Since our algorithm will find an m -guillotine unit-triangulation of P having a minimum number of vertices, and our structure theorem shows that any unit-triangulation (in particular, an optimal unit-triangulation) can be transformed into an m -guillotine unit-triangulation having approximately the same number of vertices (within factor $(1 + \epsilon)$), it follows that the unit-triangulation found by our algorithm yields a PTAS for determining R_{OPT} . We say that a (Steiner) triangulation T of P is m -guillotine if the bounding box $BB(P)$ can be recursively partitioned into (X, Y) -respecting rectangles by “ m -perfect cuts”. (At the base of the recursion are rectangles of dimensions $O(\delta)$, for which a brute-force enumeration of a constant number of cases in the dynamic programming algorithm will suffice.) A cut ℓ is m -perfect with respect to a rectangle ρ if its intersection with the triangulation has the following special structure: (i) ℓ intersects ρ , and (ii) the m -span of ℓ with respect to ρ is either empty or, if nonempty, is *canonically partitioned by the triangulation T* , in the following sense. Assume that the cut ℓ is vertical; the horizontal case is handled similarly. Let pq be one segment of $\sigma_m(\ell, \rho, T) \cap P$, with p (on an edge of T) vertically above q (also on an edge of T). Then, we say that the m -span is *canonically partitioned by T* if each component segment pq of the set $\sigma_m(\ell, \rho, T) \cap P$ that has length $|pq| \geq 2\delta$ is a union of $k = \lceil |pq|/\delta \rceil$ vertical edges of T , each of length exactly $|pq|/k$ (which is at most δ). We refer to the sequence of edges along the

m -span (each component pq) as a *Steiner bridge*. Those segments pq of length $|pq| < 2\delta$ define *small pockets* of P – such a segment bounds a simple polygon (since p and q must be on the same connected component of the boundary ∂P , and there can be no hole in the pocket, in order that the δ -medial axis have the same topology as the medial axis, as we assume in δ -accessibility), and this simple polygon has geodesic diameter $O(\delta)$ (again by δ -accessibility, since each point of P is within distance $O(\delta)$ of a disk of radius δ within P). Our main structure theorem for the PTAS is:

Theorem 5. *Let P be a δ -accessible polygonal domain with n vertices. For any fixed $\epsilon > 0$, let $m = \lceil 1/\epsilon \rceil$. Let T be a Steiner unit-triangulation of P whose $t = |T|$ vertices lie in the set \mathcal{C} of candidates. Then, there exists an m -guillotine (Steiner) unit-triangulation, T_G , of P with $t_G \leq (1 + \epsilon)t$ vertices in the set \mathcal{C} .*

The following lemma (whose proof is in the full paper) is utilized in the proof of the structure theorem.

Lemma 5. *Let ab be the m -span, $\sigma_m(\ell, \rho, T)$, of a cut ℓ through rectangle ρ and unit-triangulation T of P . Then, we can transform T to a new unit-triangulation T' that is canonically partitioned along ab by adding $O(|ab|/\delta)$ new Steiner points at candidate locations in \mathcal{C} .*

The Algorithm. The main algorithm is based on dynamic programming to compute a minimum-vertex m -guillotine (Steiner) unit-triangulation. A subproblem is specified by an (X, Y) -respecting rectangle ρ , and various boundary information specifying how the unit-triangulation of P within ρ must interface with the unit-triangulation outside of ρ . This boundary information includes up to $2m$ edges (each of length at most 1) per side of ρ ; since these edges have endpoints that lie on the grid of candidate Steiner points, \mathcal{C} , we know that there are only a polynomial number of possibilities for these edges. Importantly, the m -span on each side of ρ is partitioned into a *canonical* set of edges, which is determined solely by the location of the cuts bounding ρ , and their interactions with the (fixed) geometry of P . This means that the interface specification, between subproblems, is *succinct* (specifiable with a constant, $O(m)$, of data), as it must be for a polynomial-time dynamic program.

Theorem 6. *Let P be a multiply connected polygonal domain with n vertices. Assume that P is δ -accessible, for some fixed $0 < \delta < 1$. Then, for any fixed $\epsilon > 0$, there is an algorithm, with running time polynomial in n , that computes a unit-triangulation, T_G , of P having at most $(1 + \epsilon)R_{\text{OPT}}$ vertices.*

6 Online Maximum Area Triangulation

Theorem 7. *There is no competitive algorithm for the Online Maximum Area Triangulation Problem.*

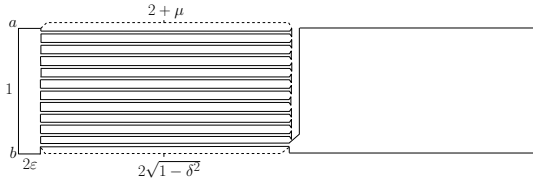


Fig. 5. An example for the polygon construction

Proof Sketch. A full proof is omitted for lack of space. For ℓ given relays we construct a polygon with $x = \lceil \frac{\ell-2}{4} \rceil$ narrow corridors, $x - 1$ of which end in a small structure and one in a large polygonal piece that allows for the placement for ℓ unit triangles. Every online algorithm for the OMATP will use all relays in the corridors, while the offline optimum needs a few relays on the way to the large polygonal piece and then places unit triangles only. Hence, every online algorithm for the OMATP covers less than $\frac{8}{\sqrt{3}}\varepsilon$ of the area that the optimal offline algorithm OPT covers for any given ε (using k relays); see Figure 5 for the general idea of the construction.

7 Conclusions

In this paper we have presented a number of online and offline results for natural problems motivated by exploration and triangulation of a region by a swarm of robots. A variety of open problems and issues remain.

Can we further improve the upper and lower bounds on the competitive factor? We believe that the final answer should be a factor of 2. On the other hand, the lower bound of $6/5$ applies to *any* algorithm; it may be possible to bring this closer to 2 by using corridor pieces of varying width. For an online strategy that separately considers boundary and interior, such as our algorithm, we believe that 2 is best possible.

As discussed above, the OMATP does not allow any strategy with a constant competitive factor, as some robots need to commit to a location before further exploration is possible. It may be interesting to consider variants in which robots may be allowed to continue exploration in a connected fashion before being required to settle down. However, this changes the basic nature of the problem, and will be treated elsewhere.

References

1. Arora, S.: Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *Journal of the ACM* 45(5), 753–782 (1998)
2. Awerbuch, B., Azar, Y., Blum, A., Vempala, S.: New approximation guarantees for minimum-weight k -trees and prize-collecting salesmen. *SIAM J. Comput.* 28, 254–262 (1998)
3. Bern, M., Eppstein, D.: Mesh generation and optimal triangulation. In: Du, D.-Z., Hwang, F.K. (eds.) *Computing in Euclidean Geometry*. Lecture Notes Series on Computing, vol. 1, pp. 23–90. World Scientific, Singapore (1992)

4. Blum, A., Chawla, S., Karger, D.R., Lane, T., Meyerson, A., Minkoff, M.: Approximation algorithms for orienteering and discounted-reward TSP. *SIAM J. Comput.* 37, 653–670 (2007)
5. Brunner, J., Mihalák, M., Suri, S., Vicari, E., Widmayer, P.: Simple Robots in Polygonal Environments: A Hierarchy. In: *Algorithmic Aspects of Wireless Sensor Networks*, pp. 111–124. Springer, Heidelberg (2008)
6. Chvátal, V.: A combinatorial theorem in plane geometry. *J. Combin. Theory Ser. B* 18, 39–41 (1975)
7. Degener, B., Fekete, S.P., Kempkes, B., auf der Heide, F.M.: A survey on relay placement with runtime and approximation guarantees. *Computer Science Review* 5(1), 57–68 (2011)
8. Efrat, A., Fekete, S.P., Gaddehosur, P.R., Mitchell, J.S., Polishchuk, V., Suomela, J.: Improved approximation algorithms for relay placement. In: *Proc. 16th Annu. Europ. Sympos. Algor.*, pp. 356–367. Springer, Heidelberg (2008)
9. Fekete, S.P., Kamphans, T., Kröller, A., Schmidt, C.: Robot swarms for exploration and triangulation of unknown environments. In: *Proceedings of the 25th European Workshop on Computational Geometry*, pp. 153–156 (2010)
10. Fekete, S.P., Mitchell, J.S.B., Schmidt, C.: Minimum covering with travel cost. In: *Proc. 20th Int. Symp. Algor. Comput.*, pp. 393–402. Springer, Heidelberg (2009)
11. Fekete, S.P., Schmidt, C.: Polygon exploration with time-discrete vision. *Comput. Geom.* 43(2), 148–168 (2010)
12. Fisk, S.: A short proof of Chvátal’s watchman theorem. *Journal of Combinatorial Theory, Series B* 24(3), 374 (1978)
13. Hoffmann, F., Icking, C., Klein, R., Kriegel, K.: The polygon exploration problem. *SIAM J. Comput.* 31, 577–600 (2001)
14. Hsiang, T.-R., Arkin, E.M., Bender, M.A., Fekete, S.P., Mitchell, J.S.B.: Algorithms for rapidly dispersing robot swarms in unknown environments. In: *Proc. 5th Workshop Algorithmic Found. Robot*, pp. 77–94 (2002)
15. Icking, C., Kamphans, T., Klein, R., Langetepe, E.: Exploring simple grid polygons. In: *11th Int. Comput. Combin. Conf*, pp. 524–533. Springer, Heidelberg (2005)
16. Knuth, D.E., Raghunathan, A.: The problem of compatible representatives. *SIAM J. Discrete Math.* 5(3), 422–427 (1992)
17. McLurkin, J., Smith, J.: Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots (2004)
18. Mitchell, J.S.B.: Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k -MST, and related problems. *SIAM J. Comput.* 28, 1298–1309 (1999)
19. O’Rourke, J.: *Art Gallery Theorems and Algorithms*. The Internat. Series of Monographs on Computer Science. Oxford University Press, Oxford (1987)
20. Suri, S., Vicari, E., Widmayer, P.: Simple robots with minimal sensing: From local visibility to global geometry. *International Journal of Robotics Research* 27, 1055–1067 (2008)

Improved Competitive Ratios for Submodular Secretary Problems

(Extended Abstract)

Moran Feldman*, Joseph (Seffi) Naor**, and Roy Schwartz

Computer Science Dept., Technion, Haifa, Israel
{moranfe,naor,schwartz}@cs.technion.ac.il

Abstract. The Classical Secretary Problem was introduced during the 60's of the 20th century, nobody is sure exactly when. Since its introduction, many variants of the problem have been proposed and researched. In the classical secretary problem, and many of its variant, the input (which is a set of secretaries, or elements) arrives in a random order. In this paper we apply to the secretary problem a simple observation which states that the random order of the input can be generated by independently choosing a random continuous arrival time for each secretary. Surprisingly, this simple observation enables us to improve the competitive ratio of several known and studied variants of the secretary problem. In addition, in some cases the proofs we provide assuming random arrival times are shorter and simpler in comparison to existing proofs. In this work we consider three variants of the secretary problem, all of which have the same objective of maximizing the value of the chosen set of secretaries given a monotone submodular function f . In the first variant we are allowed to hire a set of secretaries only if it is an independent set of a given partition matroid. The second variant allows us to choose any set of up to k secretaries. In the last and third variant, we can hire any set of secretaries satisfying a given knapsack constraint.

1 Introduction

In the (classical) secretary problem (CS), a set of n secretaries arrives in a random order for an interview. Each secretary is associated with a distinct non-negative value which is revealed upon arrival, and the objective of the interviewer is to choose the best secretary (the one having maximum value). The interviewer must decide after the interview whether to choose the candidate or not. This decision is irrevocable and cannot be altered later. The goal is to maximize the probability of choosing the best secretary. It is known that the optimal algorithm for CS

* Moran Feldman is a recipient of the Google Europe Fellowship in Market Algorithms, and this research is supported in part by this Google Fellowship.

** This work was partly supported by ISF grant 1366/07.

¹ The above definition of CS is slightly different from the standard one, though both definitions are equivalent. In the standard definition, the secretaries have ranks instead of values, and the relative rank of each secretary is revealed upon arrival.

is to reject the first n/e secretaries, and then choose the first secretary that is better than any of the first n/e secretaries. This algorithm succeeds in finding the best secretary with probability of e^{-1} [10, 24].

Throughout the years, many variants of CS have been considered. In this work we focus on variants where a subset of the secretaries can be chosen and the goal is to maximize some function of this chosen subset. Such variants of CS have attracted much attention (see, e.g., [3–5, 17, 19]). Some examples of these variants include: requiring the subset of chosen secretaries to form an independent set in a given matroid, limiting the size of the subset to at most k , and requiring the chosen secretaries to satisfy some knapsack constraints. All these variants have been studied with both linear and submodular objectives. More details on previous results can be found in Section 1.2.

In this work we use a simple observation which has been employed in problems other than CS. The observation states that the random order in which the secretaries arrive can be modeled by assigning each secretary an independent uniform random variable in the range $[0, 1)$. This continuous random variable determines the time in which the secretary arrives. Obviously, this modeling is equivalent to a random arrival order.² Though this modeling of the arrival times as continuous random variables is very simple, it has several advantages when applied to variants of the secretary problem, on which we elaborate now. First, it enables us to achieve better competitive ratios for several variants of CS. Second, the proofs of the performance guarantees of the algorithms are much simpler. The latter can be exemplified by the following simple problem.

Assume one wishes to partition the arriving secretaries into two sets where each secretary independently and uniformly chooses one of the sets, and all secretaries of one set arrive before all secretaries of the other set. An obvious difficulty is that the position of a secretary in the random arrival order depends on the positions of all other secretaries. For example, if a set S contains many secretaries that have arrived early, then a secretary outside of S is likely to arrive late, since many of the early positions are already taken by members of S . This difficulty complicates both the algorithms and their analysis. To get around this dependence [3, 19], for example, partition the secretaries into two sets: one containing the first m secretaries and the other containing the last $n-m$ secretaries. The value of m is binomially distributed $\text{Bin}(n, 1/2)$. It can be shown that this partition, together with the randomness of the input, guarantees that every secretary is uniformly and independently assigned to one of the two sets. Such an elaborate argument is needed to create the desired partitioning because of the dependencies between positions of secretaries in the arrival order.

In contrast, using the modeling of the arrival times as continuous random variables, creating a subset of secretaries where each secretary independently belongs to it with probability $1/2$ is simple. One just has to choose all secretaries that arrive before time $t = 1/2$. This simplifies the above argument, designed for a random arrival order, considerably. This simple example shows how continuous

² Given a random arrival order, we can sample n independent uniformly random arrival times, sort them and assign them sequentially to the secretaries upon arrival.

arrival times can be used to simplify both the algorithm and its analysis. In the variants of CS considered in this paper, this simplification enables us to obtain improved competitive ratios.

For some variants, the algorithms presented in this paper can be viewed as the continuous “counterparts” of known algorithms (see the uniform matroid and knapsack variants), but this is not always the case. For the partition matroid variant, the algorithm we define in this work using continuous arrival times uses different techniques than previous known algorithms for this case.

It is important to note that the continuous time “counterparts” of algorithms designed using a random arrival order are *not* equivalent to the original algorithms. For example, the optimal algorithm for CS assuming continuous random arrival times is to inspect secretaries up to time e^{-1} , and hire the first secretary after this time which is better than any previously seen secretary (see Appendix A for an analysis of this algorithm using continuous arrival times). Observe that this algorithm inspects n/e secretaries in *expectation*, while the classical algorithm inspects that number of secretaries *exactly*. This subtle difference is what enables us to improve and simplify previous results.

Formally, all problems considered in this paper are online problems in which the input consists of a set of secretaries (elements) arriving in a random order. Consider an algorithm \mathcal{A} for such a problem, and denote by opt an optimal offline algorithm for the problem. Let I be an instance of the problem, and let $\mathcal{A}(I)$ and $\text{opt}(I)$ be the values of the outputs of \mathcal{A} and opt , respectively, given I . We say that \mathcal{A} is α -competitive (or has an α -competitive ratio) if $\inf_I \frac{\mathbb{E}[\mathcal{A}(I)]}{\text{opt}(I)} \geq \alpha$, where the expectation is over the random arrival order of the secretaries of I and the randomness of \mathcal{A} (unless \mathcal{A} is deterministic). The competitive ratio is a standard measure for the quality of an online algorithm.

1.1 Our Results

In this paper we consider variants of CS where the objective function is normalized, monotone and submodular.³ There are three variants for which we provide improved competitive ratios. The first is the *submodular partition matroid secretary* problem (SPMS) in which the secretaries are partitioned into subsets, and at most one secretary from each subset can be chosen. The second is the *submodular cardinality secretary* problem (SCS) in which up to k secretaries can be chosen. The third and last variant is the *submodular knapsack secretary* problem (SKS), in which each secretary also has a cost (which is revealed upon arrival), and any subset of secretaries is feasible as long as the total cost of the subset does not exceed a given budget.

For SPMS we present a competitive ratio of $(1 - \ln 2)/2 \approx 0.153$, which improves on the current best result of $\Omega(1)$ by Gupta et al. [17]. We note that the exact competitive ratio given by [17] is not stated explicitly, however, by

³ Given a groundset \mathcal{S} , a function $f : 2^{\mathcal{S}} \rightarrow \mathbb{R}$ is called *submodular* if for every $A, B \subseteq \mathcal{S}$, $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$. Additionally, f is called *normalized* if $f(\emptyset) = 0$ and *monotone* if for every $A \subseteq B \subseteq \mathcal{S}$, $f(A) \leq f(B)$.

inspecting their result carefully it seems that the competitive ratio they achieve is at most $71/1280000$. We note that for SPMS the algorithm we provide is not a continuous time “counterpart” of the algorithm of [17]. This demonstrates the fact that modeling the arrival times as continuous random variables helps in designing and analyzing algorithms for submodular variants of CS.

For SCS we present a competitive ratio of $(e - 1)/(e^2 + e) \approx 0.170$, and the current best result for this problem is due to Bateni et al. [5] who provided a $(1 - e^{-1})/7 \approx 0.0903$ competitive ratio. There are two points to notice when comparing our result and that of [5]. First, [5] did not optimize the competitive ratio analysis of their algorithm. In fact, their algorithm provides better ratios then stated, however, it does not seem that their competitive ratio reaches 0.17. Hence, in this paper we still obtain improved competitive ratios, though the improvement is smaller than stated above. Second, the algorithm presented in this paper for SCS can be seen as a continuous time “counterpart” of [5]’s algorithm. However, our analysis is simpler than the analysis presented in [5], and also enables us to provide improved competitive ratios.

For SKS we provide a competitive ratio of $(20e)^{-1} \approx 0.0184$. The current best result is due to Bateni et al. [5] which provide a ratio of $\Omega(1)$. The exact competitive ratio is not stated in [5], but careful inspection of their algorithm shows that it is at most $96^{-1} \approx 0.0104$. Notice that the best known competitive ratio for the *linear* version of SKS is only $10e^{-1} \approx 0.0368$ [3]. As before, the algorithm presented in this paper for SKS can be seen as a continuous time “counterpart” of [5]’s algorithm. However, our analysis is simpler than the analysis presented in [5], enabling us to provide improved competitive ratios. Table 1 summarizes the above results.

1.2 Related Work

Many variants of CS have been considered throughout the years and we shall mention here only those most relevant to this work. Babaioff et al. [4] considered the case where the chosen subset of secretaries needs to be an independent set of a given matroid, and the objective function f is linear. They provided a competitive ratio of $\Omega(\log^{-1} r)$ for this problem, where r is the rank of the matroid. For several specific matroids, better constant competitive ratios are

Table 1. Comparison of our results with the known results for the monotone submodular and linear variants of the problems we consider

Problem	Our Result	Previous Result	Best Result for Linear Variant
SPMS	0.153	0.0000555 [17]	0.368^1
SCS	0.170	0.0903 [5]	0.368 [3]
SKS	0.0184	0.0104 [5]	0.0368 [3]

¹ For linear objective functions one can apply the algorithm for the classical secretary problem to each subset of secretaries independently.

known [4, 9, 18, 20]. The special variant of SCS where the objective function is linear has also been studied. Two incomparable competitive ratios were obtained by Babaioff et al. [3] and Kleinberg [19], achieving competitive ratios of e^{-1} and $1 - O(1/\sqrt{k})$, respectively. An interesting variant of SCS with a linear objective function gives each of the k possible slots of secretaries a different weight. The value of the objective function in this case is the sum of the products of a slots' weights with the values of the secretaries assigned to them. Babaioff et al. [2] provide a competitive ratio of $1/4$ for this special variant. Additional variants of CS can be found in [1, 6, 13, 14, 16].

Another rich field of study is that of submodular optimization, namely optimization problems in which the objective function is submodular. In recent years many new results in this field have been achieved. The most basic problem, in this field, is that of unconstrained maximization of a nonmonotone submodular function [12, 15]. Other works consider the maximization of a nonmonotone submodular function under various combinatorial constraints [17, 22, 25]. The maximization of a monotone submodular function under various combinatorial constraints (such as a matroid, the intersection of several matroids and knapsack constraints) has also been widely studied [7, 8, 21, 23, 25].

Thus, it comes as no surprise that recent works have combined the secretary problem with submodular optimization. Gupta et al. [17] were the first to consider this combination. For the variant where the goal is to maximize a submodular function of the chosen subset of secretaries under the constraint that this subset is independent in a given matroid, Gupta et al. [17] provide a competitive ratio of $\Omega(\log^{-1} r)$ (where r is the rank of the matroid). If the constraint is that the chosen subset of secretaries belongs to the intersection of ℓ matroids, Bateni et al. [5] provide a competitive ratio of $\Omega(\ell^{-1} \log^{-2} r)$. If the objective function is submodular and *monotone*, the special case of a partition matroid is exactly SPMS, and the special case of a uniform matroid is exactly SCS. As mentioned before, for SPMS, Gupta et al. [17] provide a competitive ratio of $\Omega(1)$ for SPMS which is at most $71/1280000$ (the exact constant is not explicitly stated in their work). They also get a similar competitive ratio for a variant of SPMS with a non-monotone submodular objective function. For SCS, Gupta et al. [17] provide a competitive ratio of $\Omega(1)$ which is at most $1/1417$ (again, the exact constant is not explicitly stated in their work), and a bit more complex $\Omega(1)$ ratio for a variant of SCS with a non-monotone submodular objective function. Both ratios were improved by Bateni et al. [5] who provided a competitive ratio of $(1 - e^{-1})/7 \approx 0.0903$ for SCS, and a $e^{-2}/8 \approx 0.0169$ -competitive algorithm for its non-monotone variant. For SKS the current best result is due to Bateni et al. [5] who provide a competitive ratio of at most $96^{-1} \approx 0.0104$ (as before, the exact constant is not explicitly stated in their work). Another extension considered by [5] is a generalization of SKS where every secretary has a ℓ dimensional cost, and the total cost of the secretaries in each dimension should not exceed the budget of this dimension (i.e., the hired secretaries should obey ℓ knapsack constraints). For this problem [5] gives an $\Omega(\ell^{-1})$ competitive algorithm.

Organization. Section 2 contains formal definitions and several technical lemmata. Sections 3, 4 and 5 provide the improved competitive ratio for SPMS, SCS and SKS, respectively.

2 Preliminaries

An instance of a constrained secretary problem consists of three components \mathcal{S}, \mathcal{I} and f .

- \mathcal{S} is a set of n secretaries arriving in a random order⁴
- $\mathcal{I} \subseteq 2^{\mathcal{S}}$ is a collection of independent sets of secretaries. The sets in \mathcal{I} are known in advance in some settings (e.g., SCS), and are revealed over time in other settings (e.g., SKS). However, at any given time, we know all independent sets containing only secretaries that already arrived.
- $f : 2^{\mathcal{S}} \rightarrow \mathbb{R}$ is a function over the set of secretaries accessed using an oracle which given a set S of secretaries that have already arrived, returns $f(S)$.

The goal is to maintain an independent set R of secretaries, and maximize the final value of $f(R)$ (i.e., its value after all secretaries have arrived). Upon arrival of a secretary s , the algorithm has to either add it to R (assuming $R \cup \{s\} \in \mathcal{I}$), or reject it. Either way, the decision is irrevocable. Given a submodular function $f : \mathcal{S} \rightarrow \mathbb{R}^+$, the discrete derivative of f with respect to s is $f_s(R) = f(R \cup \{s\}) - f(R)$. We use this shorthand throughout the paper.

Most algorithms for secretary problems with linear objective functions require every secretary to have a distinct value. This requirement does not make sense for submodular objective functions, and therefore, we work around it by introducing a total order over the secretaries, which is a standard practice (see, e.g., [9]). Formally, we assume the existence of an arbitrary fixed order Z over the secretaries. If such an order does not exist, it can be mimicked by starting with an empty ordering, and placing every secretary at a random place in this ordering upon arrival. The resulting order is independent of the arrival order of the secretaries, and therefore, can be used instead of a fixed order. Let s_1, s_2 be two secretaries, and let S be a set of secretaries. Using order Z we define $s_1 \succ_S s_2$ to denote “ $f_{s_1}(S) > f_{s_2}(S)$, or $f_{s_1}(S) = f_{s_2}(S)$ and s_1 precedes s_2 in Z ”. Notice that \succ_S is defined using f and Z . Whenever we use \succ_S , we assume f is understood from context and Z is the order defined above.

Remark: The probability that two secretaries arrive at the same time is 0, thus we ignore this event.

The following theorem is used occasionally in our proofs. Similar theorems appear in [11, 12].

⁴ If the input is given as a random permutation, the size n of \mathcal{S} is assumed to be known. Note that in order to generate the random arrival times of the secretaries and assign them upon arrival, n has to be known in advance. On the other hand, if the arrival times are part of the input, the algorithms in this work need not know n .

Theorem 1. *Given a normalized monotone submodular function $f : 2^S \rightarrow \mathbb{R}^+$, a set A and a random set A' containing every element of A with probability at least p (not necessarily independently). Then, $\mathbb{E}[f(A')] \geq p \cdot f(A)$.*

Proof. Order the elements of A in an arbitrary order $\{a_1, a_2, \dots, a_{|A|}\}$. Let X_i be an indicator for the event $\{a_i \in A'\}$, and let $A_i = \{a_1, a_2, \dots, a_i\}$. Then,

$$\begin{aligned} \mathbb{E}[f(A')] &= \mathbb{E} \left[\sum_{i=1}^{|A|} X_i \cdot f_{a_i}(A' \cap A_{i-1}) \right] = \\ & \sum_{i=1}^{|A|} \Pr[X_i = 1] \cdot \mathbb{E}[f_{a_i}(A' \cap A_{i-1}) | X_i = 1] \geq \sum_{i=1}^{|A|} p \cdot f_{a_i}(A_{i-1}) = p \cdot f(A) \end{aligned}$$

where the inequality follows from the submodularity of f .

3 $(1 - \ln 2)/2 \approx 0.153$ -Competitive Algorithm for SPMS

The Submodular Partition Matroid Secretary Problem (SPMS) is a secretary problem with a normalized monotone and submodular objective function f . The collection \mathcal{I} of independent sets is determined by $G_1 \times \dots \times G_k$ (where the G_i 's are a partition of \mathcal{S}). This variant corresponds to the scenario where the goal is to hire secretaries of different types, one of each type. For every secretary s , the index of the set G_i containing s is revealed when s arrives.

When designing an algorithm for SPMS, we want to select the best secretary from every set G_i . If f was linear, we could apply the algorithm for the classical secretary problem to each G_i separately. The following algorithm is based on a similar idea.

SPMS Algorithm(f, k):

1. Initialize $R \leftarrow \emptyset$.
2. Observe the secretaries arriving till time t^a
3. After time t , for every secretary s arriving, let G_i be the set of s . Accept s into R if:
 - (a) no previous secretary of G_i was accepted,
 - (b) and for every previously seen $s' \in G_i$, $s' \prec_R s$.
4. Return R .

^a t is a constant to be determined later.

In this subsection we prove the following theorem.

Theorem 2. *The above algorithm is a $(1 - \ln 2)/2 \approx 0.153$ -competitive algorithm for SPMS.*

The algorithm clearly maintains R as a feasible set of secretaries, hence, we only need to show that, in expectation, it finds a good set of secretaries.

Observation 3. *We can assume there is at least one secretary in every set G_i .*

Proof. The behavior of the algorithm is not effected by empty sets G_i .

3.1 Analysis of a Single G_i

In this subsection we focus on a single G_i . Let E_i be an event consisting of the arrival times of all secretaries in $\mathcal{S} - G_i$, we assume throughout this subsection that some fixed event E_i occurred. Let R_x be the set of secretaries from $\mathcal{S} - G_i$ collected by the algorithm up to time x assuming no secretary of G_i arrives (observe that R_x is not random because we fixed E_i). We define \hat{s}_x as the maximum secretary in G_i with respect to \succ_{R_x} . The analysis requires two additional event types: A_x is the event that \hat{s}_x arrives at time x , and B_x is the same event with the additional requirement that the algorithm collected \hat{s}_x .

Lemma 1. *For every $x > t$, $\Pr[B_x|A_x] \geq 1 - \ln x + \ln t$.*

Proof. Event A_x states that \hat{s}_x arrived at time x . If no other secretary of G_i is collected till time x , \hat{s}_x is collected by the definition of the algorithm. Hence, it is enough to bound the probability that no secretary of G_i is collected till time x , given A_x .

Observe that R_x takes at most $k - 1$ values for $x \in [0, 1)$. Hence, the range $[0, 1)$ can be divided into k intervals $\mathcal{I}_1, \dots, \mathcal{I}_k$ such that the set R_x is identical for all times within one interval. Divide the range $[0, x)$ into small steps of size Δy such that Δy divides t and x , and every step is entirely included in an interval (this is guaranteed to happen if Δy also divides the start time and the length of every interval \mathcal{I}_i). Since each step is entirely included in a single interval, for every time x in step j , $R_x = R_{(j-1) \cdot \Delta y}$.

A secretary cannot be collected in step j if $j \cdot \Delta y \leq t$. If this is not the case, a secretary is collected in step j if the maximum secretary of G_i in the range $[0, j \cdot \Delta y)$ with respect to $\succ_{R_{(j-1) \cdot \Delta y}}$ arrives at time $(j - 1) \cdot \Delta y$ or later. The probability that this happens is $\Delta y / (j \cdot \Delta y) = j^{-1}$. We can now use the union bound to upper bound the probability that any secretary is accepted in any of the steps before time x :

$$\Pr[B_x|A_x] \geq 1 - \sum_{j=t/\Delta y+1}^{x/\Delta y} j^{-1} \geq 1 - \int_{t/\Delta y}^{x/\Delta y} \frac{dj}{j} = 1 - [\ln j]_{t/\Delta y}^{x/\Delta y} = 1 - \ln x + \ln t .$$

Let s_i^* denote the single secretary of $G_i \cap OPT$, and let a_i be the secretary of G_i collected by the algorithm. If no secretary is collected from G_i , assume a_i is a dummy secretary of value 0 (i.e., f is oblivious to the existence of this dummy secretary in a set). We also define R_i to be the set R immediately before the algorithm collects a_i (if the algorithms collects no secretary of G_i , R_i is an arbitrary set).

Observation 4. *If B_x occurs for some x , $f_{a_i}(R_i) \geq f_{s_i^*}(R)$, where R is the set returned by the algorithm.*

Proof.

$$f_{a_i}(R_i) \stackrel{(1)}{=} f_{a_i}(R_x) \stackrel{(2)}{\geq} f_{s_i^*}(R_x) \stackrel{(3)}{\geq} f_{s_i^*}(R) .$$

Where (1) and (2) follow from the fact that B_x occurred, and therefore, a_i was collected at time x and $a_i = \hat{s}_x$. Also, B_x implies $R_x \subseteq R$, hence, the submodularity of f implies (3).

Let $B_i = \cup_{x \in (t,1)} B_x$, and let P_i be $\{s_i^*\}$ if B_i occurred, and \emptyset , otherwise.

Corollary 1. $f_{a_i}(R_i) \geq f(R \cup P_i) - f(R)$.

Proof. If $P_i = \emptyset$, the claim follows because $f_{a_i}(R_i)$ is nonnegative by the monotonicity of f . If $P_i = \{s_i^*\}$, we know that B_i occurred, and therefore, there must be an x for which B_x occurred also. The corollary now follows immediately from Observation 4.

Lemma 2. $\Pr[B_i] \geq 2 + \ln t - 2t$.

Proof. Observe that the events B_x are disjoint, hence, $\Pr[B_i]$ is the sum of the probabilities of the events B_x . Since B_x implies A_x , $\Pr[B_x] = \Pr[B_x|A_x] \cdot \Pr[A_x]$. The event A_x requires that the maximum secretary with respect to \succ_{R_x} arrives in time x . The probability that this secretary arrives in an interval of size Δx is Δx . Hence, the probability that it arrives in an infinitesimal interval of size dx is $\Pr[A_x] = dx$. Therefore,

$$\Pr[B_i] = \int_t^1 \Pr[B_x|A_x]dx \geq \int_t^1 (1 - \ln x + \ln t)dx = 2 + \ln t - 2t .$$

The last expression is maximized for $t = 0.5$, thus, we choose $t = 0.5$.

3.2 Analysis of the Entire Output

Throughout the previous subsection we assumed some fixed event E_i occurred. Hence, Corollary 1 and Lemma 2 were proven given this assumption, however, they are also true without it.

Lemma 3. *Corollary 1 and Lemma 2 also hold without fixing an event E_i .*

Proof. Corollary 1 states that for every fixed E_i , if B_i occurs then $f_{a_i}(R_i) \geq f(R \cup P_i) - f(R)$. Since some event E_i must occur (the secretaries of $\mathcal{S} - G_i$ must arrive at some times), this is also true without fixing some E_i .

Let us rephrase Lemma 2 to explicitly present the assumption that some fixed E_i occurred: $\Pr[B_i|E_i] \geq 1 - \ln 2$ (recall that we chose $t = 0.5$). Therefore,

$$\Pr[B_i] = \sum_{E_i} \Pr[E_i] \cdot \Pr[B_i|E_i] \geq (1 - \ln 2) \cdot \sum_{E_i} \Pr[E_i] = 1 - \ln 2 .$$

Let $P = \cup_{i=1}^k P_i$, and notice that $P \subseteq OPT$. The following lemma lower bounds the expected value of $f(P)$.

Lemma 4. $\mathbb{E}[f(P)] \geq (1 - \ln 2)f(OPT)$.

Proof. Every element $s_i^* \in OPT$ appears in P with probability $\Pr[B_i]$. By Lemma 2 and the value we chose for t , the last probability is at least $1 - \ln 2$. Hence, the lemma follows from Theorem 1.

We are now ready to prove Theorem 2.

Proof (Proof of Theorem 2). Observe the following.

$$\mathbb{E}[f(R)] = \mathbb{E}\left[\sum_{i=1}^k f_{a_i}(R_i)\right] \geq \mathbb{E}\left[\sum_{i=1}^k f(R \cup P_i) - f(R)\right] \geq \mathbb{E}[f(R \cup P)] - \mathbb{E}[f(R)].$$

Rearranging terms, we get: $\mathbb{E}[f(R)] \geq \frac{f(R \cup P)}{2} \geq \frac{f(P)}{2} \geq \frac{1 - \ln 2}{2} \cdot f(OPT)$.

4 The Submodular Cardinality Secretary Problem

The Submodular Cardinality Secretary Problem (SCS) is a secretary problem in which the objective function f is a normalized monotone submodular function, and we are allowed to hire up to k secretaries (the collection \mathcal{I} of independent sets contains every set of up to k secretaries).

Theorem 5. *There is a $(e - 1)/(e^2 + e) \approx 0.170$ -competitive algorithm for SCS.*

Due to space limitations, the proof of Theorem 5 is omitted from this extended abstract.

5 The Submodular Knapsack Secretary Problem

The Submodular Knapsack Secretary Problem (SKS) is a secretary problem in which the objective function f is a normalized monotone submodular function and every secretary s has a cost $c(s)$ (revealed upon arrival). A budget B is also given as part of the input, and the algorithm is allowed to hire secretaries as long as it does not exceed the budget. In other words, the collection \mathcal{I} of allowed sets contains every set of secretaries whose total cost is at most B .

Theorem 6. *There is a $1/(20e)$ -competitive algorithm for SKS.*

Due to space limitations, the proof of Theorem 6 is omitted from this extended abstract.

References

1. Ajtai, M., Megiddo, N., Waarts, O.: Improved algorithms and analysis for secretary problems and generalizations. *SIAM J. Discrete Math.* 14(1), 1–27 (2001)
2. Babaioff, M., Dinitz, M., Gupta, A., Immorlica, N., Talwar, K.: Secretary problems: Weights and discounts. In: 20th ACM-SIAM Symposium on Discrete Algorithms, pp. 1245–1254. Society for Industrial and Applied Mathematics, Philadelphia (2009)
3. Babaioff, M., Immorlica, N., Kempe, D., Kleinberg, R.: A knapsack secretary problem with applications. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) *RANDOM 2007 and APPROX 2007*. LNCS, vol. 4627, pp. 16–28. Springer, Heidelberg (2007)

4. Babaioff, M., Immorlica, N., Kleinberg, R.: Matroids, secretary problems, and on-line mechanisms. In: 18th ACM-SIAM Symposium on Discrete Algorithms, pp. 434–443. Society for Industrial and Applied Mathematics, Philadelphia (2007)
5. Bateni, M.H., Hajiaghayi, M.T., Zadimoghaddam, M.: Submodular secretary problem and extensions. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX 2010, LNCS, vol. 6302, pp. 39–52. Springer, Heidelberg (2010)
6. Buchbinder, N., Jain, K., Singh, M.: Secretary problems via linear programming. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 163–176. Springer, Heidelberg (2010)
7. Calinescu, G., Chekuri, C., Pál, M., Vondrák, J.: Maximizing a monotone submodular function subject to a matroid constraint. To appear in *SIAM J. Comput.*
8. Chekuri, C., Vondrák, J., Zenklusen, R.: Submodular function maximization via the multilinear relaxation and contention resolution schemes. In: 42nd ACM Symposium on Theory of Computer Science, pp. 783–792. ACM, New York (2011)
9. Dimitrov, N.B., Plaxton, C.G.: Competitive weighted matching in transversal matroids. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfssdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 397–408. Springer, Heidelberg (2008)
10. Dynkin, E.B.: The optimum choice of the instant for stopping a markov process. *Sov. Math. Dokl.* 4, 627–629 (1963)
11. Feige, U.: On maximizing welfare when utility functions are subadditive. *SIAM J. Comput.* 39(1), 122–142 (2009)
12. Feige, U., Mirrokni, V.S., Vondrák, J.: Maximizing non-monotone submodular functions. In: 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 461–471. IEEE Computer Society, Washington DC (2007)
13. Ferguson, T.S.: Who solved the secretary problem? *Statistical Science* 4(3), 282–289 (1989)
14. Freeman, P.R.: The secretary problem and its extensions: A review. *International Statistical Review* 51(2), 189–206 (1983)
15. Gharan, S.O., Vondrák, J.: Submodular maximization by simulated annealing. In: 22nd ACM-SIAM Symposium on Discrete Algorithms, pp. 1096–1116 (2011)
16. Gilbert, J., Mosteller, F.: Recognizing the maximum of a sequence. In: Fienberg, S., Hoaglin, D. (eds.) *Selected Papers of Frederick Mosteller*. Springer Series in Statistics, pp. 355–398. Springer, New York (2006)
17. Gupta, A., Roth, A., Schoenebeck, G., Talwar, K.: Constrained non-monotone submodular maximization: Offline and secretary algorithms. In: Saberi, A. (ed.) WINE 2010. LNCS, vol. 6484, pp. 246–257. Springer, Heidelberg (2010)
18. Im, S., Wang, Y.: Secretary problems: Laminar matroid and interval scheduling. In: 22nd ACM-SIAM Symposium on Discrete Algorithms, pp. 1096–1116 (2011)
19. Kleinberg, R.: A multiple-choice secretary algorithm with applications to online auctions. In: 16th ACM-SIAM Symposium on Discrete Algorithms, pp. 630–631. Society for Industrial and Applied Mathematics, Philadelphia (2005)
20. Korula, N., Pál, M.: Algorithms for secretary problems on graphs and hypergraphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikolettseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5556, pp. 508–520. Springer, Heidelberg (2009)
21. Kulik, A., Shachnai, H., Tamir, T.: Maximizing submodular set functions subject to multiple linear constraints. In: 20th ACM-SIAM Symposium on Discrete Algorithms, pp. 545–554. Society for Industrial and Applied Mathematics, Philadelphia (2009)

22. Lee, J., Mirrokni, V.S., Nagarajan, V., Sviridenko, M.: Maximizing non-monotone submodular functions under matroid or knapsack constraints. *SIAM J. Discrete Mathematics* 23(4), 2053–2078 (2010)

23. Lee, J., Sviridenko, M., Vondrák, J.: Submodular maximization over multiple matroids via generalized exchange properties. In: 12th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pp. 244–257. Springer, Heidelberg (2009)

24. Lindley, D.V.: Dynamic programming and decision theory. *Applied Statistics* 10, 39–51 (1961)

25. Vondrák, J.: Symmetry and approximability of submodular maximization problems. In: 50th Annual IEEE Symposium on Foundations of Computer Science, pp. 651–670. IEEE Computer Society, Washington DC (2009)

A Example - The Classical Secretary Problem

The Classical Secretary Problem (CS) is a secretary problem with the set \mathcal{I} of independent sets consisting of all singletons. We demonstrate the usefulness of the continuous model by analyzing an algorithm for this problem.

CS Algorithm(f):

1. Observe the secretaries arriving till time $t = e^{-1}$, and let L be the set of secretaries arriving until this time.
2. Let \hat{s} be the maximum secretary in L with respect to \succ_{\emptyset} ^a
3. After time t , accept the first secretary s such that $f(s) \succ_{\emptyset} f(\hat{s})$.

^a If no secretary arrives till time t , we assume $s \succ_{\emptyset} \hat{s}$ for every secretary s .

Theorem 7. *The above algorithm for CS is e^{-1} -competitive.*

Proof. Let s^* be the secretary of the optimal solution (breaking ties in favor of the earlier secretary according to \succ_{\emptyset}). Given that s^* arrives at some time $x \in (t, 1)$, s^* is accepted if one of the following conditions hold:

- No secretary arrives before time x .
- The best secretary arriving in the time range $[0, x)$ arrives before time t .

Since the secretaries are independent, with probability at least t/x , at least one of these conditions holds. The probability that s^* arrives in an interval of size ℓ is ℓ . Hence, the probability it arrives in an infinitesimal interval of size dx is dx . Therefore, by the law of total probability, the probability that the above algorithm accept s^* is at least

$$\int_t^1 \frac{t}{x} dx = t[\ln x]_t^1 = -t \ln t = e^{-1}.$$

Locating Depots for Capacitated Vehicle Routing

Inge Li Gørtz^{1,*} and Viswanath Nagarajan²

¹ Technical University of Denmark
ilg@imm.dtu.dk

² IBM T.J. Watson Research Center
viswanath@us.ibm.com

Abstract. We study a location-routing problem in the context of capacitated vehicle routing. The input to k -LocVRP is a set of demand locations in a metric space and a fleet of k vehicles each of capacity Q . The objective is to *locate* k depots, one for each vehicle, and *compute routes* for the vehicles so that all demands are satisfied and the total cost is minimized. Our main result is a constant-factor approximation algorithm for k -LocVRP. To achieve this result, we reduce k -LocVRP to the following generalization of k median, which might be of independent interest. Given a metric (V, d) , bound k and parameter $\rho \in \mathbb{R}_+$, the goal in the k median forest problem is to find $S \subseteq V$ with $|S| = k$ minimizing:

$$\sum_{u \in V} d(u, S) + \rho \cdot d(\text{MST}(V/S)),$$

where $d(u, S) = \min_{w \in S} d(u, w)$ and $\text{MST}(V/S)$ is a minimum spanning tree in the graph obtained by contracting S to a single vertex. We give a $(3 + \epsilon)$ -approximation algorithm for k median forest, which leads to a $(12 + \epsilon)$ -approximation algorithm for k -LocVRP, for any constant $\epsilon > 0$. The algorithm for k median forest is t -swap local search, and we prove that it has locality gap $3 + \frac{2}{t}$; this generalizes the corresponding result for k median [3].

Finally we consider the k median forest problem when there is a different (unrelated) cost function c for the MST part, i.e. the objective is $\sum_{u \in V} d(u, S) + c(\text{MST}(V/S))$. We show that the locality gap for this problem is unbounded even under multi-swaps, which contrasts with the $c = d$ case. Nevertheless, we obtain a constant-factor approximation algorithm, using an LP based approach along the lines of [12].

1 Introduction

In typical facility location problems, one wishes to locate centers and connect clients directly to centers at minimum cost. On the other hand, the goal in vehicle routing problems (VRPs) is to compute routes for vehicles originating from a given set of depots. Location routing problems represent an integrated

* Supported by the Danish Council for Independent Research | Natural Sciences.

approach, where we wish to make combined decisions on facility location and vehicle routing. This is a widely researched area in operations research, see eg. surveys [13,14,5,16,17]. Most of these papers deal with exact methods or heuristics, without any performance guarantees. In this paper we present an approximation algorithm for a location routing problem in context of capacitated vehicle routing.

Capacitated vehicle routing (CVRP) is an extensively studied vehicle routing problem [19] which involves distributing identical items to a set of demand locations. Formally we are given a metric space (V, d) on vertices V with distance function $d : V \times V \rightarrow \mathbb{R}_+$ that is symmetric and satisfies triangle inequality. Each vertex $u \in V$ demands q_u units of the item. We have available a fleet of k vehicles, each having capacity Q and located at specified depots. The goal is to distribute items using the k vehicles at minimum total cost. There are two versions of CVRP depending on whether or not the demand at a vertex may be satisfied over multiple visits. We focus on the *unsplit delivery* version in the paper, while noting that this also implies the result under split-deliveries.

We consider the question “where should one locate the k depots so that the resulting vehicle routing solution has minimum cost?” This is called *k-location capacitated vehicle routing* (k -LocVRP). The k -LocVRP problem bears obvious similarity to the well-known *k median* problem, where the goal is to choose k centers to minimize the sum of distances of each vertex to its closest center. The difference is that our problem also takes the routing aspect into account. Not surprisingly, our algorithm for k -LocVRP builds on approximation algorithms for the k median problem.

In obtaining an algorithm for k -LocVRP we introduce the *k median forest* problem, which might be of independent interest. The objective here is a combination of k -median and minimum spanning tree. Given metric (V, d) , bound k and parameter $\rho \in \mathbb{R}_+$, the goal is to find $S \subseteq V$ with $|S| = k$ minimizing $\sum_{u \in V} d(u, S) + \rho \cdot d(\text{MST}(V/S))$. Here $d(u, S) = \min_{w \in S} d(u, w)$ is the minimum distance between u and an S -vertex; $\text{MST}(V/S)$ is a minimum spanning tree in the graph obtained by contracting S to a single vertex. Note that when $\rho = 0$ we have the k -median objective, and ρ being very large reduces to the *k-tree* problem where the goal is to choose k centers S to minimize $d(\text{MST}(V/S))$. (Observe that the k -tree problem can be solved optimally using the greedy algorithm for MST and stopping when there are k components.) On the other hand, the k -median, k -tree, and k median forest objectives are incomparable in general. In the full version of the paper we give an instance where near-optimal solutions to these three objectives are mutually far apart.

Our Results. The main result is the following.

Theorem 1. *There is a $(12 + \epsilon)$ -approximation algorithm for k -LocVRP, for any constant $\epsilon > 0$.*

Our algorithm first reduces k -LocVRP to k median forest, at the loss of a approximation factor of four. This step is fairly straightforward and makes use of

known lower-bounds [9] for the CVRP problem. We present this reduction in Section 2. In Section 3 we prove the following result which implies Thm 1.

Theorem 2. *There is a $(3 + \epsilon)$ -approximation algorithm for k median forest, for any constant $\epsilon > 0$.*

This is the technically most interesting part of the paper. The algorithm is straightforward: perform local search using multi-swaps. It is well known that (single swap) local search is optimal for the minimum spanning tree problem. Moreover, Arya et al. [3] showed that t -swap local search achieves exactly a $(3 + \frac{2}{t})$ -approximation ratio for the k -median objective (this proof was later simplified by Gupta and Tangwongsan [8]). Thus one can hope that local search performs well for k median forest, which is a combination of both MST and k -median objectives. However, the local moves used in proving the quality of local optima are different for the MST and k -median objectives. Our proof shows we can *simultaneously* bound both MST and k -median objectives using a common set of local moves. In fact we prove that the locality gap for k median forest under t -swaps is also $(3 + \frac{2}{t})$. To bound the k -median part of the objective due to these swaps, we use the result from [8]. The interesting part of the proof is in bounding the change in the MST cost due to these swaps—this makes use of non-trivial exchange properties of spanning trees (that are easier to describe in a matroid context [13]) and additional properties of the potential swaps from [8].

Finally we consider the *non-uniform k median forest* problem. This is an extension of k median forest where there is a different cost function c for the MST part in the objective. Given vertices V with two metrics d and c , and bound k , the goal is to find $S \subseteq V$ with $|S| = k$ minimizing $\sum_{u \in V} d(u, S) + c(\text{MST}(V/S))$. Here $\text{MST}(V/S)$ is a minimum spanning tree in the graph obtained by contracting S to a single vertex, *under metric c* . It is natural to consider the local search algorithm in this setting as well, since local search achieves good approximations for both k -median and MST. However, the locality gap of non-uniform k median forest is unbounded even if we allow multiple swaps (see full version [7]). In light of this, Thm 2 appears a bit surprising. Still, we show that a different LP-based approach yields:

Theorem 3. *There is a 16-approximation algorithm for non-uniform k median forest.*

This algorithm follows closely that for the matroid median problem [12]. We consider the natural LP relaxation and round it in two phases. The first phase sparsifies the solution (using ideas from [6]) and allows us to reformulate a new LP-relaxation using fewer variables. The second phase solves the new LP-relaxation, which we show to be integral. Due to lack of space the proof of Thm 3 is omitted. It can be found in the full version of the paper [7].

Related Work. The basic capacitated vehicle routing problem involves a single fixed depot. There are two versions of CVRP: *split delivery* where the demand of a vertex may be satisfied over multiple visits; and *unsplit delivery* where the

demand at a vertex must be satisfied in a single visit (in this case we also assume $\max_{u \in V} q_u \leq Q$). Observe that the optimal value under split-delivery is at most that under unsplit-delivery. The best known approximation guarantee for split-delivery is $\alpha + 1$ [9,2] and for unsplit-delivery is $\alpha + 2$ [1], where α denotes the best approximation ratio for the Traveling Salesman Problem. We make use of the following known lower bounds for CVRP with single depot r : the minimum TSP tour on all demand locations, and $\frac{2}{Q} \sum_{u \in V} d(r, u) \cdot q_u$. Similar constant factor approximation algorithms [15] are known for the CVRP with multiple (fixed) depots.

The k median problem is a widely studied location problem and has many constant factor approximation algorithms. Starting with the LP-rounding algorithm of [6], the primal-dual approach was used in [11], and also local search [3]. A simpler analysis of the local search algorithm was given in [8]; we make use of this in our proof for the k median forest problem. Several variants of k median have also been studied. One that is relevant to us is the matroid median problem [12], where the set of open centers are constrained to be independent in some matroid; our approximation algorithm for the non-uniform k median forest problem is based on this approach.

Recently [10] studied (among other problems) a facility-location variant of CVRP: there are opening costs for depots and the goal is to open a set of depots and find vehicle routes so as to minimize the sum of opening and routing costs. The k -LocVRP problem in this paper can be thought of as the k -median variant of [10]. In [10] the authors give a 4.38-approximation algorithm for facility-location CVRP. Following a similar approach one can obtain a bicriteria approximation algorithm for k -LocVRP, where $2k$ depots are opened. However more work is needed to obtain a true approximation, and this is precisely where we need an algorithm for the k median forest problem.

2 Reducing k -LocVRP to k Median Forest

Here we show that the k -LocVRP problem can be reduced to k median forest at the loss of a constant approximation factor. This makes use of known lower bounds for CVRP [9,15,10].

For any subset $S \subseteq V$, let $\text{Flow}(S) := \frac{2}{Q} \sum_{u \in V} q_u \cdot d(u, S)$, and let $\text{Tree}(S) = d(\text{MST}(V/S))$ be the length of the minimum spanning tree in the metric obtained by contracting S . The following theorem is implicit in previous work [9,15,10]; this uses a natural MST splitting algorithm.

Theorem 4 ([10]). *Given any instance of CVRP on metric (V, d) with demands $\{q_u\}_{u \in V}$, vehicle capacity Q and depots $S \subseteq V$,*

- *The optimal value of split-delivery CVRP is at least $\max\{\text{Flow}(S), \text{Tree}(S)\}$.*
- *There is a polynomial time algorithm that computes an unsplit-delivery solution of length at most $2 \cdot \text{Flow}(S) + 2 \cdot \text{Tree}(S)$.*

Based on this it is clear that the optimal value of the CVRP instance given depot positions S is roughly given by $\text{Flow}(S) + \text{Tree}(S)$, which is similar to the

k median forest objective. The following lemma formalizes this reduction. We will assume an algorithm for the k median forest problem with vertex-weights $\{q_u : u \in V\}$, where the objective becomes $\sum_{u \in V} q_u \cdot d(u, S) + \rho \cdot d(\text{MST}(V/S))$.

Lemma 1. *If there is a β -approximation algorithm for k median forest then there is a 4β -approximation algorithm for k -LocVRP.*

Proof. Let Opt denote the optimal value of the k -LocVRP instance. Using the lower bound in Thm 4,

$$\text{Opt} \geq \min_{S:|S|=k} \max\{\text{Flow}(S), \text{Tree}(S)\} \geq \min_{S:|S|=k} [\epsilon \cdot \text{Flow}(S) + (1 - \epsilon) \cdot \text{Tree}(S)],$$

where $\epsilon \in [0, 1]$ is any value; this will be fixed later. Consider the instance of k median forest on metric (V, d) , vertex weights $\{q_u\}_{u \in V}$ and parameter $\rho = \frac{1-\epsilon}{\epsilon} \cdot \frac{Q}{2}$. For any $S \subseteq V$ the objective is:

$$\begin{aligned} \sum_{u \in V} q_u \cdot d(u, S) + \rho \cdot d(\text{MST}(V/S)) &= \frac{Q}{2} \cdot \text{Flow}(S) + \rho \cdot \text{Tree}(S) \\ &= \frac{Q}{2\epsilon} \cdot [\epsilon \cdot \text{Flow}(S) + (1 - \epsilon) \cdot \text{Tree}(S)]. \end{aligned}$$

Thus the optimal value of the k median forest instance is at most $\frac{Q}{2\epsilon} \cdot \text{Opt}$. Let S_{alg} denote the solution found by the β -approximation algorithm for k median forest. It follows that $|S_{alg}| = k$ and:

$$\epsilon \cdot \text{Flow}(S_{alg}) + (1 - \epsilon) \cdot \text{Tree}(S_{alg}) \leq \beta \cdot \text{Opt} \tag{1}$$

For the k -LocVRP instance, we locate the depots at S_{alg} . Using Thm 4, the cost of the resulting vehicle routing solution is at most $2 \cdot \text{Flow}(S_{alg}) + 2 \cdot \text{Tree}(S_{alg}) = 4 \cdot [\epsilon \cdot \text{Flow}(S_{alg}) + (1 - \epsilon) \cdot \text{Tree}(S_{alg})]$ where we set $\epsilon = 1/2$. From Inequality (1) it follows that our algorithm is a 4β -approximation algorithm for k -LocVRP. \square

We remark that this reduction already gives us a constant factor *bicriteria* approximation algorithm for k -LocVRP as follows. Let S_{med} denote an approximate solution to k -median on metric (V, d) with vertex-weights $\{q_u : u \in V\}$, which can be obtained by directly using a k -median algorithm [3]. Let S_{mst} denote the optimal solution to $\min_{S:|S| \leq k} d(\text{MST}(V/S))$, which can be obtained using the greedy MST algorithm. We output $S_{bi} = S_{med} \cup S_{mst}$ as a solution to k -LocVRP, along with the vehicle routes obtained from Thm 4 applied to S_{bi} . Note that $|S_{bi}| \leq 2k$, so we open at most $2k$ depots. Moreover, if S^* denotes the location of depots in the optimal solution to k -LocVRP then:

- $\text{Flow}(S_{med}) \leq (3 + \delta) \cdot \text{Flow}(S^*)$ since we used a $(3 + \delta)$ -approximation algorithm for k -median [3], for any constant $\delta > 0$.
- $\text{Tree}(S_{mst}) \leq \text{Tree}(S^*)$ since S_{mst} is an optimal solution to the MST part of the objective.

Clearly $\text{Flow}(S_{bi}) \leq \text{Flow}(S_{med})$ and $\text{Tree}(S_{bi}) \leq \text{Tree}(S_{mst})$, so:

$$\frac{1}{2} \cdot \text{Flow}(S_{bi}) + \frac{1}{2} \cdot \text{Tree}(S_{bi}) \leq \frac{3 + \delta}{2} \cdot [\text{Flow}(S^*) + \text{Tree}(S^*)] \leq (3 + \delta) \cdot \text{Opt}$$

Using Thm 4 the cost of the CVRP solution with depots S_{bi} is at most $4(3 + \delta) \cdot \text{Opt}$. So this gives a $(12 + \delta, 2)$ bicriteria approximation algorithm for k -LocVRP, where $\delta > 0$ is any fixed constant. We note that this approach combined with algorithms for facility-location and Steiner tree immediately gives a constant factor approximation for the facility location CVRP considered in [10]. The algorithm in that paper [10] has to do some more work in order to get a sharper constant. For k -LocVRP this approach clearly does not give any true approximation ratio, and for this purpose we give an algorithm for k median forest.

3 Multi-swap Local Search for Median Forest

The input to k median forest consists of a metric (V, d) , vertex-weights $\{q_u\}_{u \in V}$ and bound k . The goal is to find $S \subseteq V$ with $|S| = k$ minimizing:

$$\Phi(S) = \sum_{u \in V} q_u \cdot d(u, S) + d(\text{MST}(V/S)),$$

where $d(u, S) = \min_{w \in S} d(u, w)$ and $\text{MST}(V/S)$ is a minimum spanning tree in the graph obtained by contracting S to a single vertex. Note that this is slightly more general than the definition in Section 1 (which is the special case when $q_u = 1/\rho$ for all $u \in V$).

We analyze the natural t -swap local search for this problem, for any constant t . Starting at an arbitrary solution $L \subseteq V$ consisting of k centers, do the following until no improvement is possible: if there exists $D \subseteq L$ and $A \subseteq V \setminus L$ with $|D| = |A| \leq t$ and $\Phi((L \setminus D) \cup A) < \Phi(L)$ then $L \leftarrow (L \setminus D) \cup A$. Clearly each local step can be performed in $n^{O(t)}$ time which is polynomial for fixed t . The number of iterations to reach a local optimum may be super-polynomial; however this can be made polynomial by the standard method [3] of performing a local move only if the cost Φ reduces by some $1 + \frac{1}{\text{poly}(n)}$ factor. Here we omit this (minor) detail and bound the local optimum under the swaps as defined above. We prove that the locality gap of this procedure for k -median forest is at most $3 + \frac{2}{t}$. This is also tight since a matching lower bound of $3 + \frac{2}{t}$ is already known, even in the case of k -median [3]. Somewhat surprisingly, it suffices to consider exactly the same set of swaps from [8] to establish our result, although [8] did not take into account any MST contribution.

Let $F \subseteq V$ denote the local optimum solution (under t -swaps) and $F^* \subseteq V$ the global optimum. Note that $|F| = |F^*| = k$. Define map $\eta : F^* \rightarrow F$ as $\eta(w) = \arg \min_{v \in F} d(w, v)$ for all $w \in F^*$, i.e., for each optimum center $w \in F^*$, $\eta(w)$ is w 's closest center in F . For any $S \subseteq V$, let $\text{Med}(S) := \sum_{u \in V} q_u \cdot d(u, S)$, and $\text{Tree}(S) = d(\text{MST}(V/S))$ be as defined in Section 2; so $\Phi(S) = \text{Med}(S) + \text{Tree}(S)$. For any $D \subseteq F$ and $A \subseteq V \setminus F$ with $|D| = |A| \leq t$ we refer to the swap $F - D + A$ as a “ (D, A) swap”. We use the following swap construction from [8] for the k -median problem.

Theorem 5 ([8]). *For any $F, F^* \subseteq V$ with $|F| = |F^*| = k$, there are partitions $\{F_i\}_{i=1}^\ell$ of F and $\{F_i^*\}_{i=1}^\ell$ of F^* such that $\forall i \in [\ell], |F_i| = |F_i^*|$. Furthermore, for each $i \in [\ell]$, there is a unique $c_i \in F_i$ such that $\eta(w) = c_i$ for all $w \in F_i^*$ and $\eta^{-1}(v) = \emptyset$ for all $v \in F_i \setminus \{c_i\}$. Define set \mathcal{S} of t -swaps with multipliers $\{\alpha(s) : s \in \mathcal{S}\}$ as:*

- For any $i \in [\ell]$, if $|F_i| \leq t$ then swap $(F_i, F_i^*) \in \mathcal{S}$ with $\alpha(F_i, F_i^*) = 1$.
- For any $i \in [\ell]$, if $|F_i| > t$ then for each $a \in F_i^*$ and $b \in F_i \setminus \{c_i\}$ swap $(b, a) \in \mathcal{S}$ with $\alpha(b, a) = \frac{1}{|F_i|-1}$.

Then we have:

- $\sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot (\text{Med}(F - D + A) - \text{Med}(F)) \leq (3 + 2/t) \cdot \text{Med}(F^*) - \text{Med}(F)$.
- For each $w \in F^*$, the extent to which w is added $\sum_{(D,A) \in \mathcal{S}: w \in A} \alpha(D, A) = 1$.
- For each $v \in F$, the extent to which v is dropped $\sum_{(D,A) \in \mathcal{S}: v \in D} \alpha(D, A) \leq 1 + \frac{1}{t}$.

We use the same set \mathcal{S} of swaps for the k median forest problem and will show the following:

$$\sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot (\text{Tree}(F - D + A) - \text{Tree}(F)) \leq (3+2/t) \cdot \text{Tree}(F^*) - \text{Tree}(F) \quad (2)$$

Added with the similar inequality in Thm 5 for Med (since both inequalities use the same set \mathcal{S} of swaps and respective multipliers) we obtain:

$$\sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot (\Phi(F - D + A) - \Phi(F)) \leq (3 + 2/t) \cdot \Phi(F^*) - \Phi(F).$$

Finally by local optimality of F , the left-hand-side above is non-negative, and we have:

Theorem 6. *The t -swap local search algorithm for k median forest is a $(3 + \frac{2}{t})$ -approximation.*

It remains to prove (2), which we do in the rest of the section. Consider a graph H which is the complete graph on vertices $V \cup \{r\}$ (for a new vertex r). If $E = \binom{V}{2}$ denotes the edges in the metric, H has edges $E \cup \{(r, v) : v \in V\}$. The edges $\{(r, v) : v \in V\}$ are called *root-edges* and edges E are *true-edges*. Let M denote the *spanning tree* of H consisting of edges $MST(V/F) \cup \{(r, v) : v \in F\}$; similarly M^* is the spanning tree $MST(V/F^*) \cup \{(r, v) : v \in F^*\}$. For ease of notation, for any subset $S \subseteq V$, when it is clear from context we will use S to also denote the set $\{(r, v) : v \in S\}$ of root-edges. We start with the following exchange property (which holds more generally for any matroid), see Equation (42.15) in Schrijver [18].

Theorem 7 ([18]). *Given two spanning trees T_1 and T_2 in a graph H and a partition $\{T_1(i)\}_{i=1}^p$ of the edges of T_1 , there exists a partition $\{T_2(i)\}_{i=1}^p$ of edges of T_2 such that $(T_2 \setminus T_2(i)) \cup T_1(i)$ is a spanning tree in H for each $i \in [p]$. (This also implies $|T_2(i)| = |T_1(i)|$ for all $i \in [p]$).*

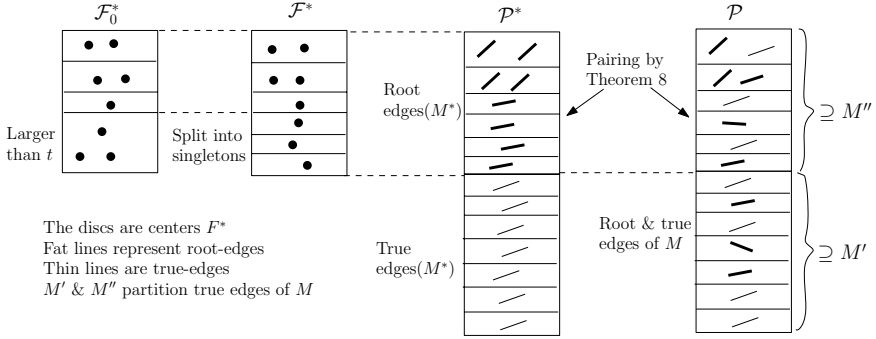


Fig. 1. The partitions used in local search proof (eg. has $k = 8$ and $t = 2$)

We will apply Thm 7 on trees M^* and M . Throughout, M^* and M represent the corresponding edge-sets. Recall the partition $\mathcal{F}_0^* := \{F_i^*\}_{i=1}^\ell$ of F^* from Thm 5, we refine \mathcal{F}_0^* by splitting parts of size larger than t into singletons, and let \mathcal{F}^* denote the resulting partition (see Fig. 1). The reason behind splitting the large parts of $\{F_i^*\}_{i=1}^\ell$ is to ensure the following property (recall swaps \mathcal{S} from Thm 5).

Claim 8. For each swap $(D, A) \in \mathcal{S}$, $A \subseteq F^*$ appears as a part in \mathcal{F}^* . Moreover, for each part A' in \mathcal{F}^* there is some swap $(D', A') \in \mathcal{S}$.

Consider the partition \mathcal{P}^* of M^* with parts $\mathcal{F}^* \cup \{e\}_{e \in M^* \setminus F^*}$, i.e. each true edge lies in a singleton part and the root edges form the partition \mathcal{F}^* defined above. Let \mathcal{P} denote the partition of M obtained by applying Thm 7 with partition \mathcal{P}^* of M^* ; note also that there is a pairing between parts of \mathcal{P} and \mathcal{P}^* . Let $M' \subseteq M \cap E$ denote the true edges of M that are paired with true edges of M^* ; and $M'' = (M \cap E) \setminus M'$ are the remaining true edges of M (see also Fig. 1). We will bound the cost of M' and M'' separately.

Claim 9. $\sum_{e \in M'} d_e \leq \sum_{h \in E \cap M^*} d_h$.

Proof. Fix any $e \in M'$. By the definition of M' it follows that there is a true-edge $h \in E \cap M^*$ such that part $\{h\}$ in \mathcal{P}^* is paired with part $\{e\}$ in \mathcal{P} . In particular, $M - e + h$ is a spanning tree in H . Note that the root edges in $M - e + h$ are exactly F , and so $M - e + h$ is a spanning tree in the original metric graph (V, E) when we contract vertices F . Since $M = MST(V/F)$ is the minimum such tree, we have $d(M) - d_e + d_h \geq d(M)$, implying $d_e \leq d_h$. Summing over all $e \in M'$ and observing that each edge $h \in E \cap M^*$ can be paired with at most one $e \in M'$, we obtain the claim. \square

The true-edges of M induce a forest. Consider the connected components in this forest: for each $f \in F$, let $C_f \subseteq V$ denote the vertices connected to f . Note that $\{C_f : f \in F\}$ partitions V .

Now consider the forest induced by true edges of M^* and direct each edge towards an F^* -vertex (note that each tree in this forest contains exactly one

F^* -vertex). Observe that each vertex $v \in V \setminus F^*$ has exactly one out-edge σ_v , and F^* -vertices have none.

For each $f \in F$, define $T_f := \{\sigma_v : v \in C_f\}$ the set of out-edges incident from vertices of C_f . Since $\{C_f : f \in F\}$ partitions V , it follows that $\{T_f\}_{f \in F}$ partitions $E \cap M^*$, and:

Claim 10. $\sum_{f \in F} d(T_f) = d(E \cap M^*)$.

We are now ready to bound the increase in the Tree cost under swaps \mathcal{S} . By Claim 8 it follows that for each swap $(D, A) \in \mathcal{S}$, A is a part in \mathcal{F}^* (and so in \mathcal{P}^*). Let E_A be the (possibly empty) set of true-edges of M that are paired with the part A of \mathcal{P}^* .

Claim 11. $\{E_A : (D, A) \in \mathcal{S}\}$ is a partition of M'' .

Proof. Consider the partition \mathcal{P} of M given by Thm 7 applied to \mathcal{P}^* . By definition, $M' \subseteq E \cap M$ are the true edges of M paired (by \mathcal{P} and \mathcal{P}^*) with true edges of M^* ; and $M'' = (E \cap M) \setminus M'$ are paired with parts from \mathcal{F}^* (i.e. root edges of M^*). For each part $\pi \in \mathcal{F}^*$ (and also \mathcal{P}^*) let $E(\pi) \subseteq M''$ denote the M'' -edges paired with π . It follows that $\{E(\pi) : \pi \in \mathcal{F}^*\}$ partitions M'' . Using the second fact in Claim 8 and the definition E_{AS} , we have $\{E_A : (D, A) \in \mathcal{S}\} = \{E(\pi) : \pi \in \mathcal{F}^*\}$, a partition of M'' . \square

We will now prove the following key lemma.

Lemma 2. For each swap $(D, A) \in \mathcal{S}$,

$$\text{Tree}(F - D + A) - \text{Tree}(F) \leq 2 \cdot \sum_{f \in D} d(T_f) - d(E_A).$$

Proof. By Claim 8, $A \subseteq F^*$ is a part in \mathcal{P}^* . Recall that E_A denotes the true-edges of M paired with A ; let F_A denote the root-edges of M paired with A . Then using Thm 7 it follows that $(M \setminus (E_A \cup F_A)) \cup A$ is a spanning tree in H . Hence, the remaining true-edges $S_A := (E \cap M) \setminus E_A$ is a forest with each component containing some center from $F \cup A$. In other words, S_A connects each vertex to some vertex of $F \cup A$. For any $f \in F \cup A$ let C'_f denote vertices in the component of S_A containing f . Note that $\{C'_f : f \in F \cup A\}$ is a refinement of the previously defined partition $\{C_f : f \in F\}$.

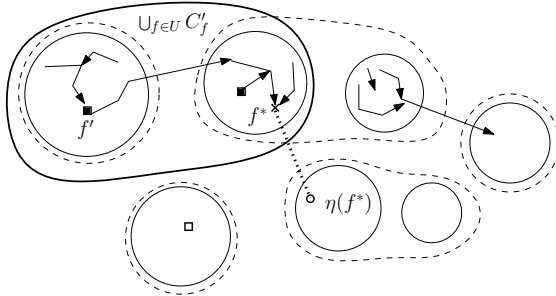
Components containing a center from D , but not from $F - D + A$ might not be connected to a center in $F - D + A$. Consider the (true) edge set $S'_A := S_A \cup_{f \in D} T_f$. We will add a set N of true edges so that $S'_A \cup N$ connects each D -vertex to some vertex of $F - D + A$. Since S_A already connects all vertices to $F \cup A$, it would follow that $S'_A \cup N$ connects all vertices to $F - D + A$, and therefore

$$\text{Tree}(F + A - D) \leq d(S'_A) + d(N) \leq \text{Tree}(F) - d(E_A) + \sum_{f \in D} d(T_f) + d(N).$$

To prove the lemma it now suffices to construct a set N with $d(N) \leq \sum_{f \in D} d(T_f)$, such that $S'_A \cup N$ connects each D -vertex to $F - D + A$. Below, we use $\delta(V')$ to denote the edges of S'_A between V' and $V \setminus V'$ for any $V' \subseteq V$.

Constructing N Consider any minimal $U \subseteq D$ such that $\delta\left(\bigcup_{f \in U} C'_f\right) = \emptyset$. By minimality of U , it follows that $\bigcup_{f \in U} C'_f$ is connected in S'_A . By construction of the swaps \mathcal{S} in Thm 5, we have

Claim 12. For any $f^* \in F^* \setminus A$ we have $\eta(f^*) \notin D$.



Squares denote vertices in D ; here $|D| = 3$. Solid squares are vertices in $U \subseteq D$; here $|U| = 2$. Components $\{C_f : f \in F\}$ are the dashed regions. Solid lines are edges in $\bigcup_{f \in U} T_f$. Components $\{C'_f : f \in F \cup A\}$ are the solid circles. Dotted edge $(f^*, \eta(f^*))$ is added to set N .

Fig. 2. Addition of edges to set N

We need the following claim (see also example in Fig. 2).

Claim 13. There exists $f^* \in F^* \cap \left(\bigcup_{f \in U} C'_f\right)$ and $f' \in U$ such that $\bigcup_{f \in U} T_f$ contains a path between f' and f^* .

Proof. Let any $f' \in U$. Consider the directed path P from f' obtained by following *out-edges* σ until the *first occurrence* of a vertex v that is either in F^* or in $V \setminus \left(\bigcup_{f \in U} C'_f\right)$. Since F^* -vertices are the only ones with no out-edge σ , and the set of all true-edges $\{\sigma_w : w \in V\} = E \cap M^*$ is acyclic, there must exist such a vertex $v \in F^* \cup \left(V \setminus \left(\bigcup_{f \in U} C'_f\right)\right)$. To see that $P \subseteq \bigcup_{f \in U} T_f$, observe that $C'_f \subseteq C_f$ for all $f \in D \supseteq U$; recall that C_s (resp. C'_s 's) are the connected components in M (resp. $S_A \subseteq M$). So $P \subseteq \{\sigma_w : w \in \bigcup_{f \in U} C'_f\} \subseteq \{\sigma_w : w \in \bigcup_{f \in U} C_f\} = \bigcup_{f \in U} T_f$. Suppose for contradiction that vertex $v \notin F^*$. Then $v \in V \setminus \left(\bigcup_{f \in U} C'_f\right)$, but this implies $\delta\left(\bigcup_{f \in U} C'_f\right) \neq \emptyset$ since path $P \subseteq \bigcup_{f \in U} T_f \subseteq S'_A$ leaves $\bigcup_{f \in U} C'_f$. So $v \in F^* \cap \left(\bigcup_{f \in U} C'_f\right)$ and $P \subseteq \bigcup_{f \in U} T_f$ is a path from f' to v . □

Consider f^* and f' as given Claim 13. If $f^* \in A$ then the component $\bigcup_{f \in U} C'_f$ of S'_A is already connected to $F - D + A$. Otherwise by Claim 12 we have $\eta(f^*) \notin D$; in this case we add edge $(f^*, \eta(f^*))$ to N which connects component $\bigcup_{f \in U} C'_f$ to $\eta(f^*) \in F - D \subseteq F - D + A$. Now using Claim 13, $d(f^*, \eta(f^*)) \leq d(f^*, f') \leq$

$\sum_{f \in U} d(T_f)$ **[4]** In either case, U is connected to $F - D + A$ in $S'_A \cup N$, and cost of N increases by at most $\sum_{f \in U} d(T_f)$.

We apply the above argument to every minimal $U \subseteq D$ with $\delta \left(\bigcup_{f \in U} C'_f \right) = \emptyset$. The increase in cost of N due to each such U is at most $\sum_{f \in U} d(T_f)$. Since such minimal sets U s are disjoint, we have $d(N) \leq \sum_{f \in D} d(T_f)$. Clearly $S'_A \cup N$ connects each D -vertex to $F - D + A$. \square

Using Lemma **[2]** for each $(D, A) \in \mathcal{S}$ weighted by $\alpha(D, A)$ (from Thm **[5]**) and adding them together,

$$\begin{aligned} & \sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot [\text{Tree}(F - D + A) - \text{Tree}(F)] \\ & \leq 2 \cdot \sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot \sum_{f \in D} d(T_f) - \sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot d(E_A) \end{aligned} \tag{3}$$

$$= 2 \sum_{f \in F} \left(\sum_{(D,A) \in \mathcal{S}: f \in D} \alpha(D, A) \right) \cdot d(T_f) - \sum_{e \in M''} \left(\sum_{(D,A) \in \mathcal{S}: e \in E_A} \alpha(D, A) \right) \cdot d_e \tag{4}$$

$$\leq 2 \left(1 + \frac{1}{t} \right) \sum_{f \in F} d(T_f) - \sum_{e \in M''} d_e \tag{5}$$

$$= 2 \left(1 + \frac{1}{t} \right) \cdot d(E \cap M^*) - d(M'') \tag{6}$$

Above **(3)** is by Lemma **[2]**, **(4)** is by interchanging summations using the fact that $E_A \subseteq M''$ (for all $(D, A) \in \mathcal{S}$) from Claim **[11]**. The first term in **(5)** uses the property in Thm **[5]** that each $f \in F$ is dropped (i.e. $f \in D$) to extent at most $1 + \frac{1}{t}$; the second term uses the property in Thm **[5]** that each $f^* \in F^*$ is added to extent one in \mathcal{S} and Claim **[11]**. Finally **(6)** is by Claim **[10]**.

Adding the inequality $0 \leq d(E \cap M^*) - d(M')$ from Claim **[9]** yields:

$$\sum_{(D,A) \in \mathcal{S}} \alpha(D, A) \cdot [\text{Tree}(F - D + A) - \text{Tree}(F)] \leq \left(3 + \frac{2}{t} \right) \cdot d(E \cap M^*) - d(E \cap M),$$

since M' and M'' partition the true edges $E \cap M$. Thus we obtain Inequality **(2)**.

References

1. Altinkemer, K., Gavish, B.: Heuristics for unequal weight delivery problems with a fixed error guarantee. *Operations Research Letters* 6, 149–158 (1987)
2. Altinkemer, K., Gavish, B.: Heuristics for delivery problems with constant error guarantees. *Transportation Research* 24, 294–297 (1990)

¹ This is the key point in the proof where we use uniformity in the metrics for the k -median and MST objectives.

3. Arya, V., Garg, N., Khandekar, R., Meyerson, A., Munagala, K., Pandit, V.: Local search heuristics for k -median and facility location problems. *SIAM J. Comput.* 33(3), 544–562 (2004)
4. Balakrishnan, A., Ward, J.E., Wong, R.T.: Integrated facility location and vehicle routing models: Recent work and future prospects. *Amer. J. Mathematical and Management Sciences* 7, 35–61 (1987)
5. Berman, O., Jaillet, P., Simchi-Levi, D.: Location-routing problems with uncertainty. In: Drezner, Z. (ed.) *Facility Location: A Survey of Applications and Methods*, pp. 427–452. Springer, Heidelberg (1995)
6. Charikar, M., Guha, S., Tardos, É., Shmoys, D.B.: A constant-factor approximation algorithm for the k -median problem. *J. Comput. Syst. Sci.* 65(1), 129–149 (2002)
7. Gørtz, I.L., Nagarajan, V.: Locating depots for capacitated vehicle routing. *CoRR*, abs/1103.0985 (2011)
8. Gupta, A., Tangwongsan, K.: Simpler analyses of local search algorithms for facility location. *CoRR*, abs/0809.2554 (2008)
9. Haimovich, M., Kan, A.H.G.R.: Bounds and heuristics for capacitated routing problems. *Mathematics of Operations Research* 10(4), 527–542 (1985)
10. Harks, T., König, F., Matuschke, J.: Approximation algorithms for capacitated location routing. T.U. Berlin Preprint 010-2010
11. Jain, K., Vazirani, V.V.: Approximation algorithms for metric facility location and k -median problems using the primal-dual schema and lagrangian relaxation. *J. ACM* 48(2), 274–296 (2001)
12. Krishnaswamy, R., Kumar, A., Nagarajan, V., Sabharwal, Y., Saha, B.: The matroid median problem. In: *SODA* (2011)
13. Laporte, G.: Location-routing problems. In: Golden, B.L., Assad, A.A. (eds.) *Vehicle Routing: Methods and Studies*, pp. 163–198. North-Holland, Amsterdam (1988)
14. Laporte, G.: A survey of algorithms for location-routing problems. *Investigación Operativa* 1, 93–123 (1989)
15. Li, C., Simchi-Levi, D.: Worst-case analysis of heuristics for multidepot capacitated vehicle routing problems. *ORSA Journal on Computing* 2(1), 64–73 (1990)
16. Min, H., Jayaraman, V., Srivastava, R.: Combined location-routing problems: A synthesis and future research directions. *Eur. J. Oper. Res.* 108, 1–15 (1998)
17. Nagy, G., Salhi, S.: Location-routing: Issues, models and methods. *Eur. J. Oper. Res.* 177(2), 649–672 (2007)
18. Schrijver, A.: *Combinatorial Optimization*. Springer, Heidelberg (2003)
19. Toth, P., Vigo, D.: *The vehicle routing problem*. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia (2002)

Satisfying Degree- d Equations over $GF[2]^n$

Johan Håstad*

Royal Institute of Technology

Abstract. We study the problem where we are given a system of polynomial equations defined by multivariate polynomials over $GF[2]$ of fixed constant degree $d > 1$ and the aim is to satisfy the maximal number of equations. A random assignment approximates this problem within a factor 2^{-d} and we prove that for any $\epsilon > 0$, it is NP-hard to obtain a ratio $2^{-d} + \epsilon$. When considering instances that are perfectly satisfiable we give a probabilistic polynomial time algorithm that, with high probability, satisfies a fraction $2^{1-d} - 2^{1-2d}$ and we prove that it is NP-hard to do better by an arbitrarily small constant. The hardness results are proved in the form of inapproximability results of Max-CSPs where the predicate in question has the desired form and we give some immediate results on approximation resistance of some predicates.

1 Introduction

The study of polynomial equations is a basic question of mathematics. In this paper we study a problem we call Max- d -Eq where we are given a system of m equations of degree d in n variables over $GF[2]$. As we consider the case of d constant, all polynomials are given in the dense representation. Many problems can be coded as polynomial equations and in particular it is easy to code 3-Sat as equations of degree 3 and thus determining whether we can simultaneously satisfy all equations is NP-complete. It is hence natural to study the question of satisfying the maximal number of equations and our interests turn to approximation algorithms. We say that an algorithm is a C -approximation algorithm if it always returns a solution which satisfies at least $C \cdot \text{OPT}$ equations where OPT is the number of equations satisfied by the optimal solution. The PCP-theorem [21] shows that it is NP-hard to approximate the Max- d -Eq within some constant $C < 1$ and from the results of [6] it is not difficult to get an explicit constant of inapproximability. Given the importance of the problem it is, however, natural to try to determine the exact approximability of the problem and this is the purpose of this paper.

The result of [6] proves that the optimal approximability constant for linear equations ($d = 1$) is $\frac{1}{2}$. This approximability is obtained by simply picking a random assignment independently of the equations at hand. To prove tightness it is established that for any $\epsilon > 0$ it is NP-hard to approximate the answer better than within a factor $\frac{1}{2} + \epsilon$. This is proved by constructing a suitable

* This research was supported by ERC grant 226 203.

Probabilistic Checkable Proof (PCP). It turns out that these results extend almost immediately to the higher degree case giving the optimal constant 2^{-d} for degree- d equations. We proceed to study the case when all equations can be simultaneously satisfied.

In the case of linear equations, it follows by Gaussian elimination that once it is possible to satisfy all equations one can efficiently find such a solution. The situation for higher degree equations turns out to be more interesting. Any implied affine condition can be used to eliminate a variable but this turns out to be the limit of what can be achieved. To be more precise, from a characterization of low weight code words of Reed-Muller codes by Kasami and Tokura [7] it follows that any equation satisfied by a fraction lower than $2^{1-d} - 2^{1-2d}$ must imply an affine condition. This number turns out to be the sharp threshold of approximability for satisfiable instances of systems of degree- d equations.

The upper bounds is obtained by using implied affine conditions to eliminate variables and then choosing an assignment to the remaining variables at random. For $d \geq 3$ we are not able to derandomize this algorithm and thus in general this is a probabilistic algorithm.

The lower bound is proved by constructing a PCP very much inspired by [6] and indeed nothing in the current paper relies on material not known at the time of that paper. In particular, we prove standard NP-hardness results and do not use any sophisticated results in harmonic analysis.

As a by-product of our proofs we make some observations in the area of maximum constraint satisfaction problems (max-CSPs). The problem Max- P is given by a predicate P of arity k and an instance is given by a sequence of k -tuples of literals. The task is to find an assignment such that the maximal number of the resulting k -tuples of bits satisfy P . Let $r(P)$ be the probability that a random assignment satisfies P . Note that $r(P)$ is the approximation ratio achieved by the algorithm that simply picks a random assignment independent of the instance under consideration. We have the following definition.

Definition 1. *A predicate P is approximation resistant if, for any $\epsilon > 0$, it is NP-hard to approximate Max- P within $r(P) + \epsilon$.*

There is also a stronger notion of hardness.

Definition 2. *A predicate P is approximation resistant on satisfiable instances if, for any $\epsilon > 0$, it is NP-hard to distinguish instances of Max- P where all constraints can be satisfied simultaneously from those where only a fraction $r(P) + \epsilon$ of the constraints can be satisfied simultaneously.*

Given a predicate P of arity k we construct a predicate, P^L , of arity $3k$ by replacing each input by the exclusive-or of three bits. A straightforward extension of our techniques show that for any P , the resulting predicate P^L is approximation resistant and if P does not imply an affine condition the result also applies to satisfiable instances. Using these results it is possible to construct a predicate that is approximation resistant while for satisfiable instances there is a better approximation ratio that is still strictly smaller than one but larger than the ratio given by the random assignment.

An outline of the paper is as follows. In Section 2 we give some preliminaries and the rather easy result for non-perfect completeness is given in Section 3. The most technically interesting part of the paper is given in Section 4 where we study systems of equations where all equations can be satisfied simultaneously. We give the results on max-CSPs in Section 5 and end with some final remarks in Section 6. Some proofs are only sketched due to space limitations.

2 Preliminaries

We are interested in polynomials over $GF[2]$. Most polynomials we use are of degree d but also polynomials of degree one, that we call “affine forms” play a special role. We are not interested in the polynomials as formal polynomials, but rather as functions mapping $GF[2]^n$ to $GF[2]$ and hence we freely use that $x_i^2 = x_i$ and thus any term in our polynomials can be taken to be multilinear. We start with the following standard result.

Theorem 1. *Any multivariate polynomial P of degree d that is nonzero takes the value 1 for at least a fraction 2^{-d} of the inputs.*

It is not difficult to see that this result is tight by considering $P(x) = \prod_{i=1}^d x_i$, or more generally, products of d linearly independent affine forms. It is important for us that these are the only cases of tightness. This follows from a characterization by Kasami and Tokura [7] of all polynomials that are non-zero for at most a fraction 2^{1-d} of the inputs. A consequence of their characterization is the following theorem.

Theorem 2. [7] *Let P be a degree d polynomial over $GF[2]$ that does not contain an affine factor. Then the fraction of points on which $P(x) = 1$ is at least $2^{1-d} - 2^{1-2d}$.*

We make use of the Fourier transform and as we are dealing with polynomials over $GF[2]$ we let the inputs come from $\{0, 1\}^n$. For any $\alpha \subseteq [n]$ we have the character χ_α defined by

$$\chi_\alpha(x) = (-1)^{\sum_{i \in \alpha} x_i}$$

and the Fourier expansion of a function f is given by

$$f(x) = \sum_{\alpha \subseteq [n]} \hat{f}_\alpha \chi_\alpha(x).$$

Suppose that $R \leq L$ and we are given a projection π mapping $[L]$ to $[R]$. We define a related operator π_2 acting on sets such that $\pi_2(\beta) = \alpha$ for $\beta \subseteq [L]$ and $\alpha \subseteq [R]$ iff exactly the elements of α has an odd number of preimages that belong to β . If we have an $x \in \{0, 1\}^R$ and define $y \in \{0, 1\}^L$ by setting $y_i = x_{\pi(i)}$ then $\chi_\beta(y) = \chi_{\pi_2(\beta)}(x)$.

As is standard we use the long code introduced by Bellare et al [5]. If $v \in [L]$ then the corresponding long code is a function $A : \{0, 1\}^L \rightarrow \{-1, 1\}$ where

$A(x) = (-1)^{x_v}$. We want our long codes to be folded, which means that they only contain values for inputs with $x_1 = 1$. The value when $x_1 = 0$ is defined to be $-A(\bar{x})$. This ensures that the function is unbiased and that the Fourier coefficient corresponding to the empty set is 0.

3 The Case of Non-perfect Completeness

We start with the algorithm.

Theorem 3. *Given a system of m polynomial equations of degree d over $GF[2]$, it is possible to, in polynomial time, to find an assignment that satisfies at least $m2^{-d}$ equations.*

Proof. In fact, by Theorem [1](#), a random assignment satisfies each equation with probability 2^{-d} and thus just picking a random assignment gives a randomized algorithm fulfilling the claim of the theorem in expectation.

To get a deterministic algorithm we use the method of conditional expectations. To be more precise we assign values to the variables in order and we choose values for the variables such that the expected number of satisfied equations if the remaining variables are set randomly never drops below $m2^{-d}$. When all variables are set any equation is satisfied with probability either 0 or 1 and hence at least $m2^{-d}$ equations are satisfied. To make this procedure efficient we use the lower estimate that at least a fraction $2^{-d'}$ of the inputs satisfies any nontrivial equation that is currently of degree d' .

The lower bound follows rather immediately from known results.

Theorem 4. *For any $\epsilon > 0$ it is NP-hard to approximate Max- d -Eq within $2^{-d} + \epsilon$.*

Proof. In [\[6\]](#) it is proved that it is NP-hard to distinguish systems of linear equations where a fraction $1 - \epsilon$ of the equations can be satisfied from those where only a fraction $\frac{1}{2} + \epsilon$ can be satisfied. Suppose we are given an instance of this problem with m equations which, possibly by adding one to both sides of the equation, can be assumed to be of the form

$$A_i(x) = 1.$$

Taking all d -wise products of such equations we end up with m^d equations, each of the form

$$\prod_{j=1}^d A_{i_j}(x) = 1,$$

which clearly is a polynomial equation of degree at most d . This system has the same optimal solution as the linear system and if it satisfies δm linear equations then it satisfies $\delta^d m^d$ degree- d equations. The theorem now follows, by adjusting ϵ from the result of [\[6\]](#).

We remark that, by appealing to the results by Raz and Moshkovitz [\[8\]](#), we can even obtain results for non-constant values of ϵ .

4 Completely Satisfiable Systems

When studying systems where it is possible to simultaneously satisfy all equations the situation changes. Suppose we have an equation of the form $P(x) = 1$ and that this equation implies the affine condition $A(x) = 1$. Then, as the system is satisfiable, we can use this equation to eliminate one variable from the system, preserving the degrees of all equations. This is done by taking any variable x_i that appears in A and replacing it by $x_i + A(x) + 1$ (note that this function does not depend on x_i as the two occurrences of this variable cancel). This substitution preserves the satisfiability of the system and the process stops only when none of the current equations implies an affine condition.

Using Theorem [2](#) we see that when this process ends each equation is satisfied by at least a fraction $2^{1-d} - 2^{1-2d}$ of the inputs. It seems reasonable to hope that for each perfectly satisfiable system we can efficiently find an assignment that satisfies this fraction of the equations. There are two points in the outlined argument that require closer inspection. The first is the question of how to actually determine whether a polynomial equation implies an affine condition and the second is to make sure that once the process of finding implied affine conditions has ended we can indeed deterministically find a solution that satisfies the expected number of equations. Let us first address the issue of determining whether a given equation implies an affine condition.

Suppose $P(x) = 1$ implies $A(x) = 1$ for some unknown affine function A . Let us assume that x_1 appears in A with a nonzero coefficient. We may write

$$P(x) = P_0(x) + P_1(x)x_1$$

where neither P_0 nor P_1 depends on x_1 . Consider

$$Q(x) = P(x) + A(x)P_1(x). \tag{1}$$

As x_1 appears with coefficient one in A it follows that Q does not depend on x_1 and let us assume that Q is not identically 0. Choose any values for $x_2, x_3 \dots x_n$ to make $Q(x) = 1$ and set x_1 to make $A(x) = 0$. It follows from [\(I\)](#) that $P(x) = 1$ and thus we have found a counterexample to the assumed implication. We can hence conclude that $Q \equiv 0$ and we have

$$P(x) = A(x)P_1(x).$$

We claim furthermore that this procedure is entirely efficient. Namely given P and the identity of one variable occurring in A , P_1 is uniquely defined. Once P_1 is determined the rest of the coefficients of A can be found by solving a linear system of equations. As there are only n candidates for a variable in A and solving a linear system of equations is polynomial time we conclude that the entire process of finding implied affine conditions can be done in polynomial time.

Once this process halts we need to implement the method of conditional expectations to find an assignment that satisfies the expected number of equations.

As opposed to the case of Theorem 3 where we could use the lower bound of $2^{-d'}$ for the fraction of inputs that satisfy any degree- d' equation we here need to find a more accurate bound to calculate the conditional expectation. We do not know how to do this in deterministic polynomial time and hence we pick a random assignment of the remaining variables and see if it satisfies the target number of equations. If it does not, we keep repicking random assignments until we are successful. We conclude.

Theorem 5. *There is a probabilistic polynomial time algorithm that given a system of m simultaneously satisfiable equations of degree d over $GF[2]$ finds an assignment that satisfies at least $(2^{1-d} - 2^{1-2d})m$ equations.*

We note that the above argument only established that we get at least $(2^{1-d} - 2^{1-2d})m$ satisfied equations on average and we need to prove that we get this number with a some non-negligible probability. This follows, however, by a standard argument and we leave the details to the reader.

Let us remark that for $d = 2$ it is possible to make the algorithm deterministic. This follows from the fact that we can transform any degree 2 polynomial into a normal form from which we can read off the fraction of inputs for which it is equal to 1. We omit the details and let us turn to the lower bound.

Theorem 6. *For any $\epsilon > 0$ it is NP-hard to distinguish satisfiable instances of Max- d -Eq from those where the optimal solution satisfies a fraction $2^{1-d} - 2^{1-2d} + \epsilon$ of the equations.*

Proof. Consider the predicate, P , on $6d$ variables given by

$$P(x) = \prod_{i=1}^d L_i(x) + \prod_{i=d+1}^{2d} L_i(x), \tag{2}$$

where $L_i(x) = x_{3i-2} + x_{3i-1} + x_{3i}$, i.e. each L_i is the exclusive or of three variables and no variable appears in two linear forms. Theorem 6 now follows from Theorem 7 below as the probability that a random assignment satisfies P is exactly $2^{1-d} - 2^{1-2d}$.

Theorem 7. *The predicate P defined by (2) is approximation resistant on satisfiable instances.*

Proof. We reduce the standard projecting label cover instance to Max- P for this predicate P . This is the same starting point as in [6] but let us formulate it in more modern terms.

We are given a bipartite graph with vertices U and V . Each vertex $u \in U$ should be given a label $\ell(u) \in [L]$ and each vertex $v \in V$ should be given a label $\ell(v) \in [R]$. For each edge (u, v) there is a mapping $\pi_{u,v}$ and a labeling satisfies this edge iff $\pi_{u,v}(\ell(u)) = \ell(v)$.

In [6] we used the fact that for any constant ϵ there are constant values for L and R such that it is NP-hard to determine whether the optimal labeling

satisfies all constraints or only a fraction ϵ of the constraints, and this is all that we need also here. Using [8] one can extend this to non-constant size domains, but let us ignore this point.

As is standard, we transform the label cover instance into a PCP by long-coding a good assignment, and for each vertex u we have a table $g_u(y)$ for $y \in \{0, 1\}^L$, and similarly we have a table $f_v(x)$ for $x \in \{0, 1\}^R$ for each $v \in V$. As mentioned in the preliminaries we assume that these long codes are folded and hence each table is unbiased.

Before we describe how the verifier checks this PCP we define an “error” distribution, D_μ , on $2d$ bits $(\mu_i)_{i=1}^{2d}$. First pick a random bit b with uniform probability and if $b = 0$ set $\mu_i = 1$ for $1 \leq i \leq d$ and select values for the other d bits uniformly from the $2^d - 1$ binary strings that contains at least one 0. If $b = 1$ we do the symmetric assignment exchanging the two halves. We need two simple facts about the distribution D_μ . The first is obvious from construction.

Lemma 1. *With probability one it is true that*

$$\prod_{i=1}^d \mu_i + \prod_{i=d+1}^{2d} \mu_i = 1.$$

Secondly we have.

Lemma 2. *For any nonempty set S and $d \geq 2$, we have*

$$|E_{D_\mu} [(-1)^{\sum_{i \in S} \mu_i}]| \leq \frac{1}{2}.$$

Proof. If S is contained in one of the two halves we observe that the distribution on this half is obtained by picking a string from the uniform distribution with probability $\frac{1}{2}(1 + (2^d - 1)^{-1})$ and otherwise picking the all one string. It follows that in this case

$$|E_{D_\mu} [(-1)^{\sum_{i \in S} \mu_i}]| = \frac{1}{2}(1 - (2^d - 1)^{-1}) < \frac{1}{2}.$$

If, on the other hand, S contains inputs from both halves then by conditioning on which half gets the all one assignment it is easy to see that

$$|E_{D_\mu} [(-1)^{\sum_{i \in S} \mu_i}]| \leq (2^d - 1)^{-1} < \frac{1}{2}.$$

Let us return to defining our PCP by the actions of the verifier. For readability we drop the obvious subscripts on f, g and π .

1. Pick an edge (u, v) which comes with a projection constraint $\pi : [L] \mapsto [R]$.
2. Pick $x^{(i)} \in \{0, 1\}^R$ and $y^{(i)} \in \{0, 1\}^L$ uniformly at random, $1 \leq i \leq 2d$.
3. For each $j \in [L]$ pick an element $\mu^{(j)}$ with the distribution D_μ and construct $z^{(i)}$ by setting $z_j^{(i)} = x_{\pi(j)}^{(i)} + y_j^{(i)} + \mu_i^{(j)} \pmod 2$.

4. Read the $6d$ bits \square corresponding to $f(x^{(i)})$, $g(y^{(i)})$, and $g(z^{(i)})$. Accept if these $6d$ bits satisfy P where the three bits fed into L_i are $f(x^{(i)})$, $g(y^{(i)})$, and $g(z^{(i)})$.

We have first have the easy completeness lemma.

Lemma 3. *The verifier accepts a correct proof of a correct statement with probability 1.*

Proof. Suppose the proof gives labels $\ell(u)$ to $\ell(v)$ to u and v , respectively. Then $g_u(y^{(i)}) = (-1)^{y^{\ell(u)}}$, $g_u(z^{(i)}) = (-1)^{z^{\ell(u)}}$, $f_v(x^{(i)}) = (-1)^{x^{\ell(v)}}$. As $\pi(\ell(u)) = \ell(v)$ the exclusive-or (product in the ± 1 notation) of these bits equal $(-1)^{\mu_i^{\ell(u)}}$. The lemma now follows from Lemma \square

We turn to soundness.

Lemma 4. *If the verifier accepts with probability at least $2^{1-d} - 2^{1-2d} + \epsilon$ then there is a labeling in the label cover problem that satisfies at least a fraction $c_d \epsilon^2$ of the conditions for some constant $c_d > 0$ depending only on d .*

Proof. Expand the predicate P by its multilinear expansion. Since the constant term, \hat{P}_\emptyset , is $2^{1-d} - 2^{1-2d}$ we conclude that given the assumption of the lemma there are non-empty sets S_1, S_2 and S_3 such that

$$|E[\prod_{i \in S_1} f(x^{(i)}) \prod_{i \in S_2} g(y^{(i)}) \prod_{i \in S_3} g(z^{(i)})]| \geq c_d \epsilon, \tag{3}$$

for some constant c_d depending only on d .

Not all terms of the form \square appear in the expansion of P but as we can bound any such term and we make some use of this fact in Section \square we treat an arbitrary term.

First note that if $S_2 \neq S_3$ the expectation in \square is zero as for any i in the symmetric difference we get a factor $g(y^{(i)})$ or $g(z^{(i)})$ that is independent of the other factors and as g is folded the expectation of such a term is 0. To get a non-zero value we also need $S_1 = S_3$ as otherwise negating $x^{(i)}$ in the symmetric difference we get cancelling terms. Thus we need to study

$$E \left[\prod_{i \in S} f(x^{(i)}) g(y^{(i)}) g(z^{(i)}) \right]. \tag{4}$$

Expanding each function by the Fourier transform we get the expectation

$$E \left[\prod_{i \in S} \left(\sum_{\alpha^i, \beta^i, \gamma^i} \hat{f}_{\alpha^i} \hat{g}_{\beta^i} \hat{g}_{\gamma^i} \chi_{\alpha^i}(x^{(i)}) \chi_{\beta^i}(y^{(i)}) \chi_{\gamma^i}(z^{(i)}) \right) \right]. \tag{5}$$

¹ We interpret -1 as the bit 1 and 1 as the bit 0.

If we mentally expand this product of sums and look at the expectation of each term we see, as $y_j^{(i)}$ is independent of all other variables, that terms with $\gamma^i \neq \beta^i$ give contribution 0. The same is true if $\pi_2(\beta^i) \neq \alpha^i$. Let μ_i denote the vector $(\mu_i^{(j)})_{j=1}^L$ then

$$\chi_{\pi_2(\beta^i)}(x^{(i)})\chi_{\beta^i}(y^{(i)})\chi_{\beta^i}(z^{(i)}) = \chi_{\pi_2(\beta^i)}(x^{(i)})\chi_{\beta^i}(y^{(i)})\chi_{\beta^i}(x_{\pi}^{(i)} + y^{(i)} + \mu_i) = \chi_{\beta^i}(\mu_i),$$

and thus (5) reduces to

$$E \left[\prod_{i \in S} \left(\sum_{\beta^i} \hat{f}_{\pi_2(\beta^i)} \hat{g}_{\beta^i}^2 \chi_{\beta^i}(\mu_i) \right) \right]. \tag{6}$$

We have

$$\prod_{i \in S} \chi_{\beta^i}(\mu_i) = \prod_{j \in \cup_i \beta^i} (-1)^{\sum \mu_i^{(j)}} \tag{7}$$

where the sum in the exponent is over the set of i such that $j \in \beta^i$. By Lemma 2 it follows that the absolute value of the expectation of (7) is bounded by

$$2^{-|\cup_i \beta^i|} \leq 2^{-\sum_{i \in S} |\beta^i|/2d},$$

and hence we can conclude from (4) that

$$E_{u,v} \left[\prod_{i \in S} \left(\sum_{\beta^i} |\hat{f}_{\pi_2(\beta^i)}| \hat{g}_{\beta^i}^2 2^{-|\beta^i|/2d} \right) \right] \geq c_d \epsilon. \tag{8}$$

As S is nonempty and any factor is bounded from above by one we conclude that

$$E_{u,v} \left[\sum_{\beta} |\hat{f}_{\pi_2(\beta)}| \hat{g}_{\beta}^2 2^{-|\beta|/2d} \right] \geq c_d \epsilon. \tag{9}$$

Cauchy-Schwarz inequality implies that

$$\sum_{\beta} |\hat{f}_{\pi_2(\beta)}| \hat{g}_{\beta}^2 2^{-|\beta|/2d} \leq \left(\sum_{\beta} \hat{g}_{\beta}^2 \right)^{1/2} \left(\sum_{\beta} \hat{f}_{\pi_2(\beta)}^2 \hat{g}_{\beta}^2 2^{-|\beta|/d} \right)^{1/2} \tag{10}$$

$$\leq \left(\sum_{\beta} \hat{f}_{\pi_2(\beta)}^2 \hat{g}_{\beta}^2 2^{-|\beta|/d} \right)^{1/2}. \tag{11}$$

And thus from (9), and $E[X^2] \geq E[X]^2$ we can conclude that

$$E_{u,v} \left[\sum_{\beta} \hat{f}_{\pi_2(\beta)}^2 \hat{g}_{\beta}^2 2^{-|\beta|/d} \right] \geq c_d^2 \epsilon^2. \tag{12}$$

We can now extract a probabilistic labeling using the standard procedure. For each u we choose a set β with probability \hat{g}_β^2 and return a random element in β . Similarly for each v we choose a set α with probability \hat{f}_α^2 and return a random element in α . The expected fraction of satisfied constraints is at least

$$E_{u,v} \left[\sum_{\beta} \hat{f}_{\pi_2(\beta)}^2 \hat{g}_\beta^2 \frac{1}{|\beta|} \right] \tag{13}$$

and as

$$\frac{1}{x} \geq \frac{1}{d} 2^{-x/d}$$

for any $x \geq 1$ we have that (13) is at least $\frac{c_d^2}{d} \epsilon^2$ and, adjusting the value of c_d this completes the proof of Lemma 4.

Theorem 7 now follows from Lemma 4 and Lemma 3 by the standard way of transforming a PCP with an acceptance criteria given by a predicate P to a hardness result for the corresponding constraint satisfaction problem Max- P .

5 Consequences for Max-CSPs

Let us draw some conclusions from the argument in the proof of Theorem 7. In this section, let P be an arbitrary predicate of arity k . Define P^L be the predicate of arity $3k$ obtained by replacing each input bit of P by the exclusive-or of three independent bits, similarly to constructing the predicate of the previous section. We have the following theorem.

Theorem 8. *For any predicate P that accepts at least one input, the predicate P^L is approximation resistant.*

Proof. (Sketch) Let $\alpha \in \{0, 1\}^k$ be an input accepted by P . Define a distribution D_μ by setting $\mu_i = \alpha_i$ with probability $1 - \delta$ and otherwise $\mu_i = \overline{\alpha_i}$, independently for each i , but otherwise follow the protocol in the proof of Theorem 7. The completeness of this protocol is at least $1 - \delta$, but as δ is an arbitrarily small constant and we only need almost-perfect completeness this is not a problem. The soundness analysis of this verifier is now similar to that of the analysis in the proof of Theorem 7 using

$$\left| E \left[\prod_{i \in S} \chi_{\beta^i}(\mu_i) \right] \right| = (1 - 2\delta)^{\sum_{i \in S} |\beta^i|},$$

resulting in an almost identical argument but with different constants.

It is not difficult to see that for any P , P^L supports a measure that is pairwise independent. This implies that the results of Austrin and Mossel 4 would have been sufficient to give approximation resistance assuming the unique games conjecture. In our case we get NP-hardness which is an advantage and it is also possible to get a general theorem with perfect completeness.

Theorem 9. *For any predicate P such that $P^{-1}(1)$ is non-contained in a $(k-1)$ -dimensional affine subspace of $\{0, 1\}^k$, the predicate P^L is approximation resistant for satisfiable instances.*

Proof. (Sketch) We choose μ uniformly from the set of strings accepted by P . As $\sum_{i \in S} \mu_i$ is not constant for any non-empty S , the equivalent of Lemma 2 is true with the constant $\frac{1}{2}$ replaced by some other constant strictly smaller than one affecting the values of some constants but not the rest of the argument.

It is tempting to guess that for any P that does imply an affine condition and hence Theorem 9 does not apply, P^L would not be approximation resistant on satisfiable instances. This does not seem to be obviously true and let us outline the problems.

It is true that P^L is a polynomial of degree at most k and we can use the implied affine conditions to eliminate some variables as we did in the proof of Theorem 5. The final stage when we have no more implied affine constraints is, however, more difficult to control. The resulting constraints are given by affine constraints in conjunction with the original P . By the assumption on perfect satisfiability we can conclude that the each equation is still satisfiable but not much more.

If, however, our predicate is of limited degree when viewed as a polynomial we have more information on the result. Clearly during the process of eliminating affine constraints, the degree does not increase, and in fact it decreases when we remove the known affine factor within each polynomial. We get the following conclusion.

Theorem 10. *Suppose predicate P of arity k is given by a polynomial of degree d that contains r linearly independent affine factors. Then if P accepts less than a fraction $2^{1-(d-r)} - 2^{1-2(d-r)}$ of the inputs, P^L is approximation resistant but not approximation resistant on satisfiable instances, unless $NP \subseteq BPP$.*

Proof. (Sketch) The predicate is approximation resistant by Theorem 8. On perfectly satisfiable instances we can run the algorithm of Theorem 5, and as we remove affine constraints the resulting degree is at most $d - r$.

The simplest example of a predicate for which this theorem applies is the predicate, P , given by the equation

$$x_1(x_2x_3 + x_4x_5) = 1$$

which has $d = 3$ and $r(P) = \frac{3}{16}$. For this instantiation of P , P^L is approximation resistant but not approximation resistant for satisfiable instances. To get a hardness result for satisfiable constraints we can use Theorem 7 for the predicate

$$x_2x_3 + x_4x_5 = 1$$

which is approximation resistant with factor $\frac{3}{8}$ on satisfiable instances. We get a matching algorithm as the affine factor can be removed and the equations that remain are of degree 2.

Let us finally point out that all our approximation resistance results establish the stronger property of “uselessness” introduced by Austrin and Håstad [3]. This follows as we are able to bound arbitrary non-trivial characters and not only the characters appearing in the considered predicates.

6 Final Words

The current paper gives optimal approximability results for satisfying the maximal number of low degree equations over $GF[2]$. The methods used in the proofs are more or less standard and thus the main contribution of this paper is to obtain tight results for a natural problem. There is a provable difference between perfectly satisfiable and almost-perfectly satisfiable systems in that we can satisfy strictly more equations in the former case. The difference is not as dramatic as in the linear case, but still striking.

For the case of Max-CSPs we obtained a few approximation resistance results for, admittedly, non-standard predicates. We feel, however, that the examples give, a not major but nonempty, contribution towards understanding the difference of approximation resistant predicates and those predicates that have this property also on satisfiable instances. Our example of an approximation resistant predicate which has another, nontrivial, approximation constant on satisfiable instances is the first of its kind. Although not surprising this result gives another piece in the puzzle to understand Max-CSPs.

Acknowledgement. I am grateful to Parikshit Gopalan for alerting me to the paper [7] and providing me with an electronic version of that paper.

References

1. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and intractability of approximation problems. *Journal of the ACM* 45, 501–555 (1998)
2. Arora, S., Safra, S.: Probabilistic checking of proofs: a new characterization of NP. *Journal of the ACM* 45, 70–122 (1998)
3. Austrin, P., Håstad, J.: On the usefulness of predicates. Manuscript (2011)
4. Austrin, P., Mossel, E.: Approximation resistant predicates from pairwise independence. *Computational Complexity* 18, 249–271 (2009)
5. Bellare, M., Goldreich, O., Sudan, M.: Free bits, PCPs and non-approximability—towards tight results. *SIAM Journal on Computing* 27, 804–915 (1998)
6. Håstad, J.: Some optimal inapproximability results. *Journal of ACM* 48, 798–859 (2001)
7. Kasami, T., Tokura, N.: On the weight structure of Reed-Muller codes. *IEEE Transactions on Information Theory* 16, 752–759 (1970)
8. Moshkovitz, D., Raz, R.: Two query PCP with sub-constant error. *Journal of the ACM* 57 (2010)

Black-Box Reductions in Mechanism Design

Zhiyi Huang^{1,*}, Lei Wang², and Yuan Zhou³

¹ University of Pennsylvania, Philadelphia PA 19104, USA
hzhyyi@cis.upenn.edu

² Georgia Institute of Technology, Atlanta GA 30332, USA
lwang@cc.gatech.edu

³ Carnegie Mellon University, Pittsburgh PA 15213, USA
yuanzhou@cs.cmu.edu

Abstract. A central question in algorithmic mechanism design is to understand the additional difficulty introduced by truthfulness requirements in the design of approximation algorithms for social welfare maximization. In this paper, by studying the problem of single-parameter combinatorial auctions, we obtain the first black-box reduction that converts *any* approximation algorithm to a truthful mechanism with essentially the *same* approximation factor in a prior-free setting. In fact, our reduction works for the more general class of *symmetric single-parameter* problems. Here, a problem is symmetric if its allocation space is closed under permutations.

As extensions, we also take an initial step towards exploring the power of black-box reductions for general single-parameter and multi-parameter problems by showing several positive and negative results. We believe that the algorithmic and game theoretic insights gained from our approach will help better understand the tradeoff between approximability and the incentive compatibility.

1 Introduction

In an *algorithmic mechanism design* problem, we face an optimization problem where the necessary inputs are private valuations held by self-interested agents. The high-level goal of *truthful* mechanisms is to reveal these valuations via the bids of the agents and to optimize the objective simultaneously. In this paper, we will focus on the objective of social welfare maximization.

It is well known that the *VCG* mechanism ([24][7][13]) which maximizes the social welfare exactly is truthful. As usual in computer science, computational tractability is a necessary requirement. However, VCG is not computationally efficient in general. And unfortunately, the simple combination of approximation algorithms and VCG usually fails to preserve truthfulness. This raises the important open question (see [21]) of whether the design of truthful mechanisms is fundamentally harder than the design of approximation algorithms for social welfare maximization.

* This work is supported in part by the National Science Foundation under Grant No. 0716172 and ONR MURI Grant N000140710907.

Recently, several positive results indicated that one can always convert an approximation algorithm to a truthful mechanism with the same approximation factor in the *Bayesian* setting where the distributions of the agents are public knowledge (see [15,3,14]). However, not much is known in the *prior-free* setting where no distribution is known.

In this paper, by studying the problem of *single-parameter combinatorial auctions*, we show the first black-box reduction that converts any approximation algorithm to a universal truthful mechanism with the same approximation factor in the prior-free setting.

In the single-parameter combinatorial auction problem, we are given a set \mathcal{J} of m items and a *public* valuation function $f : 2^{\mathcal{J}} \rightarrow \mathbf{R}$. Assume that f is given via an oracle which takes a set S as input and returns $f(S)$. In addition, we have n agents each of whom has a *private* multiplier v_i^* such that the item set S provides $v_i^* f(S)$ amount of utility to agent i . The goal is to design a truthful mechanism which maximizes $\sum_i v_i f(S_i)$, where $S_1 \cdots S_n$ is a partition of \mathcal{J} .

This problem has its motivation in the TV ad auctions where the items are time slots and each agent is an advertiser whose private multiplier is her *value-per-viewer*. In [12], the authors provided a logarithmic approximate truthful mechanism for this problem under the assumption that f is submodular. However, the optimal approximation algorithm for the underlying social welfare maximization has a ratio of $1 - 1/e$ given by Vondrak ([25]). By our result, applying Vondrak's algorithm as a black-box, we immediately obtain a truthful mechanism with the optimal constant approximation ratio.

Main Result. In fact, our black-box reduction not only works for this particular problem but for a broad class of *symmetric single parameter* problems. Formally, a mechanism design problem (with n agents) is *single-parameter* if each feasible allocation is represented as an n -dimensional real vector \mathbf{x} , and each agent i has a private value v_i such that her valuation of allocation \mathbf{x} is given by $v_i x_i$. We further define that a problem is *symmetric* if the set of feasible allocations is closed under permutations: if \mathbf{x} is feasible, so is $\pi \circ \mathbf{x}$ for any permutation π . Here $\pi \circ \mathbf{x}$ is defined as the vector $(x_{\pi(1)}, \dots, x_{\pi(n)})$.

Theorem 1. *For a symmetric single-parameter mechanism design problem Π , suppose we are given an α -approximate ($\alpha > 1$) algorithm \mathcal{A} as a black-box, then for any constant $\epsilon > 0$, we can obtain a polynomial time truthful mechanism with approximation factor $\alpha(1 + \epsilon)$.*

Many interesting mechanism design problems such as position auctions in sponsored search are in the class of symmetric single-parameter problems. In particular, it contains the problem of single-parameter combinatorial auctions that we are interested in.

Corollary 1. *For the single-parameter submodular combinatorial auction problem, there is an optimal $1-1/e$ approximate truthful mechanism.*

Our construction is based on the technique of *maximum-in-range*. Here, a maximum-in-range mechanism outputs the allocation maximizing the social welfare over a fixed range of allocations. Using the algorithm \mathcal{A} as a black-box, we

construct a range \mathcal{R} such that social welfare maximization over \mathcal{R} is efficient. And we will prove that the approximation factor obtained is essentially α .

In our reduction, we make no assumption on the black-box algorithm \mathcal{A} . In addition, while the black-box algorithm may be randomized, our reduction does not introduce any further randomization. If the algorithm is deterministic, then our mechanism is deterministically truthful.

Extensions: Positive and Negative Results. A natural extension of our result is to consider the general (possibly asymmetric) single-parameter mechanism design problems. By a novel relation between mechanism design and constraint satisfaction problems, we derive a significant lower bound of the approximability of maximum-in-range mechanisms for general single-parameter problems, which to some extent suggests the difficulty in designing factor-preserving black-box reductions.

However, we are able to generalize our algorithmic technique to some special *multi-parameter* settings. We study the *constant-dimension* and *symmetric* mechanism design problem. We generalize our construction in the symmetric single-parameter case to this problem and obtain a black-box reduction that converts any algorithm into a truthful and quasi-poly-time mechanism with essentially the same approximation guarantee. Alternatively, we can obtain a black-box reduction that converts any algorithm into a truthful and polynomial time mechanism with logarithmic degradation in the approximation factor.

Related Work. There has been a significant amount of work related to black-box reductions in mechanism design. In the single-parameter setting, the first black-box reduction was given by Briest et al. [4]. The authors studied the single-parameter binary optimization problem and they showed that any algorithm which is an FPTAS can be converted to a truthful mechanism that is also an FPTAS. Secondly, Babaioff et al. [1] studied the single-value combinatorial auction problem and they constructed a black-box reduction that converts an algorithm to a truthful mechanism with the approximation factor degraded by a logarithmic factor. Finally, the recent work by Goel et al. [12] provided a black-box reduction with a super constant degrade in approximation factor for partially public combinatorial auction.

For multi-parameter problems, there is no factor-preserving black-box reduction in general (e.g. [22]). This motivates the study of *truthfulness in expectation*, which is a weaker notion of incentive compatibility. Here, a randomized mechanism is truthful in expectation, if truth telling maximizes an agent's expected payoff. The initial effort in black-box reduction for multi-parameter problems is due to Lavi and Swamy [18], they showed a method to convert a certain type of algorithms called integrality-gap-verifiers to truthful in expectation mechanisms with the same approximation factors. Recently, Dughmi and Roughgarden [9] studied the class of packing problems. Via an elegant black-box reduction and smooth analysis, they showed that if a packing problem admits an FPTAS, then it admits a truthful in expectation mechanism that is an FPTAS as well. Balcan et al. [2] considered black-box reductions from the revenue maximization aspect. By the technique of sample complexity in machine learning, they gave

revenue-preserving reductions from truthful mechanism design to the algorithmic pricing problems. At last, Dughmi et al. [10] introduce a method to convert convex rounding scheme into truthful in expectation mechanism and achieve an optimal $(1 - \frac{1}{e})$ -approximation for the combinatorial auction problem when the valuations are a special type of submodular functions.

The previous discussion is about *prior-free* mechanism design. Another important area in algorithmic game theory is the *Bayesian* mechanism design where each agent’s valuation is drawn from some publicly known prior distribution. Hartline and Lucier [15] studied this problem in the single-parameter setting. They constructed a clever black-box reduction that converts any non-monotone algorithm into a monotone one without compromising its social welfare. Following this work, Bei and Huang [3] and Hartline et al. [14] independently showed such black-box reductions in the multi-parameter setting as well.

2 Preliminaries

In this section, we will outline the basic concepts in mechanism design relevant to our paper.

Truthfulness. Let \mathcal{X} be the set of all feasible allocations, and $v_i(\mathbf{x})$ be the private valuation of agent i if allocation $\mathbf{x} \in \mathcal{X}$ is picked. A typical goal of a mechanism is to reveal agents’ private valuation functions via their bids and optimize the obtained social welfare simultaneously. Formally, suppose we are given n agents and let $\mathbf{v} = (v_1, \dots, v_n)$ be the valuation functions reported by the agents. Based on this, a (deterministic) mechanism M will specify an allocation $\mathbf{x}(\mathbf{v}) \in \mathcal{X}$ and a payment $\mathbf{p}(\mathbf{v})$. We say M is *deterministically truthful* (or truthful), if the following conditions hold: for any i, \mathbf{v}_{-i} and any v_i, v'_i , we have $v_i(\mathbf{x}(v_i, \mathbf{v}_{-i})) - p_i(v_i, \mathbf{v}_{-i}) \geq v_i(\mathbf{x}(v'_i, \mathbf{v}_{-i})) - p_i(v'_i, \mathbf{v}_{-i})$.

When a mechanism is randomized, there are two notions of truthfulness: (1) *Universal truthfulness*: A universally truthful mechanism is a probability distribution over deterministically truthful mechanisms; (2) *Truthfulness in expectation*: A mechanism is truthful in expectation if an agent maximizes her *expected utility* by being truthful. Here, an agent’s utility is defined as her valuation minus payment. It is easy to see that every deterministically truthful mechanism is universally truthful and every universally truthful mechanism is truthful in expectation.

Single-parameter Mechanism Design. In a *single-parameter* mechanism design problem, each allocation is represented as an n -dimensional real vector \mathbf{x} (where n is the number of agents), and each agent i has a private value v_i such that her valuation of allocation \mathbf{x} is given by $v_i x_i$. It is known [20] that for a single-parameter problem, a mechanism is truthful if and only if (1) the allocation rule is *monotone*: suppose $v_i \leq v'_i$, then $x_i(v_i, \mathbf{v}_{-i}) \leq x_i(v'_i, \mathbf{v}_{-i})$; (2) each agent i ’s payment is determined by $p_i(\mathbf{v}) = v_i x_i(v_i, \mathbf{v}_{-i}) - \int_0^{v_i} x_i(t, \mathbf{v}_{-i}) dt$.

Maximum-in-range Mechanisms. The *maximum-in-range* technique is a general approach in the field of mechanism design. It works as follows: The mechanism

fixes a range \mathcal{R} of allocations *without* any knowledge of the agents' valuations. Given any \mathbf{v} , let $\mathbf{x}^* = \operatorname{argmax}_{\mathbf{x} \in \mathcal{R}} \sum_j v_j(\mathbf{x})$ and $\mathbf{x}_{-i}^* = \operatorname{argmax}_{\mathbf{x} \in \mathcal{R}} \sum_{j \neq i} v_j(\mathbf{x})$ respectively. Now define payment p_i of agent i to be $\sum_{j \neq i} v_j(\mathbf{x}_{-i}^*) - \sum_{j \neq i} v_j(\mathbf{x}^*)$. It is now not difficult to see that with this payment function, it is in best interest of every agent to report their true valuations, irrespective of what others report. The major challenge in designing maximum-in-range mechanism is to balance between the size of the range and the approximation factor obtained. A larger range can obtain better approximation but yield greater computational complexity.

3 Symmetric Single-Parameter Mechanism Design

Recall that a single-parameter mechanism design problem is symmetric if the allocation space \mathcal{X} is closed under permutations: if $\mathbf{x} \in \mathcal{X}$, then $\pi \circ \mathbf{x} = (x_{\pi(1)}, \dots, x_{\pi(n)}) \in \mathcal{X}$ for any permutation π . In this section, we will prove Theorem 1. For a symmetric single-parameter problem Π , given any constant $\epsilon > 0$ and any α -approximate algorithm \mathcal{A} as a black-box, we design a polynomial time truthful mechanism with approximation factor $(1 + \epsilon)\alpha$.

Our construction is based on the *maximum-in-range* technique. Given an algorithm \mathcal{A} , we define a range \mathcal{R} by applying \mathcal{A} as a black-box on a carefully chosen collection of *typical bid vectors*. Our mechanism is then maximum-in-range over \mathcal{R} . We will show: (1) To maximize social welfare over \mathcal{R} for a given bid vector, we only need to examine polynomially many allocations in \mathcal{R} , hence our mechanism is efficient; (2) Every bid vector can be mapped to a typical bid with approximately the same social welfare, hence our mechanism performs almost as well as the algorithm \mathcal{A} . This proves the approximation factor.

Now we describe our range construction in detail for a given symmetric single-parameter problem Π , black-box algorithm \mathcal{A} and constant $\epsilon > 0$.

3.1 Construction of the Range

Let $V = \mathbf{R}_+^n$ be the collection of all possible bid vectors. Next we will provide a three-step procedure choosing a subset $T \subseteq V$ as our collection of *typical bids*.

The first step is *normalization*: By properly reordering the agents and scaling their bids, we only consider the set T_0 of bids where $\mathbf{v} \in T_0$ if and only if $1 = v_1 \geq \dots \geq v_n$; The second step is *discretization*. In this step, our goal is to obtain a finite set of bid vectors that approximately represent the whole valuation space V . To do this, given any vector $\mathbf{v} \in T_0$, we first apply the operation of *tail cutting*: We choose a small value u (e.g. $1/n^M$ for some constant M) and round all the entries smaller than u to 0; then, we discretize the interval $[u, 1]$ by considering $Q = \{\eta^k : k \geq 0\} \cap [u, 1]$ where $\eta < 1$ is a fixed constant. We will round down each of the remaining entries of \mathbf{v} after the tail cutting to the closest value in Q . If we do the above for each $\mathbf{v} \in T_0$, we obtain a finite set of vectors T_1 ; The final step is *equalization*. We fix a constant $\beta > 1$ and partition $[n]$ into $\log_\beta n$ groups. For each vector in T_1 , we equalize its entries within each

1: **Normalization.** Let $T_0 = \{v : 1 = v_1 \geq \dots \geq v_n\}$;

2: **Discretizing.** Let $Q = \{\eta^k : 0 \leq k \leq \lceil \log_{1/\eta}(n^M) \rceil\}$ where $M \geq \log_2 \frac{8}{\epsilon}$ is a constant. For any real value z , define $\lfloor z \rfloor_\eta = \eta^{\lceil \log_\eta z \rceil} \in Q$. Then we define a function $D : T_0 \mapsto T_0$ as follows: for each $v \in T_0$ and for each i , define

$$D(v)_i = \begin{cases} \lfloor v_i \rfloor_\eta & v_i \geq u = \frac{1}{n^M} \\ 0 & \text{otherwise} \end{cases}$$

Let $T_1 = D(T_0)$;

3: **Equalization.** Let $n_k = \lfloor \beta^k \rfloor$ where $\beta > 1$ is a fixed constant and $0 \leq k \leq \lfloor \log_\beta n \rfloor$. Define a function $E : T_1 \rightarrow T_1$ as follows: for each $v \in T_1$ and $1 \leq i \leq n$, $E(v)_i$ is set to be v_{n_k} when $n_k \leq i < n_{k+1}$. At last, let $T = E(T_1)$.

group by setting them to be the value of the largest entry in the group. We then obtain the set of vectors T , and each vector in T is called a *typical bid*.

Now we provide the detailed description. In the following, we fix constants $\beta > 1$ and $\eta < 1$ such that $\frac{\beta}{\eta} = 1 + \epsilon/2$. For a bid vector v , let $x^A(v)$ be the allocation obtained by applying algorithm \mathcal{A} on v . Since the allocation space is closed under permutations, we may assume $x^A(v)_1 \geq x^A(v)_2 \geq \dots \geq x^A(v)_n$. At last, let $\mathcal{R}_0 = \{x^A(v) : v \in T\}$ and we finally define our range as $\mathcal{R} = \{\pi \circ x : x \in \mathcal{R}_0, \pi \in \Pi_n\}$ where Π_n consists of all permutations over n elements.

Now we analyze the performance of our mechanism. Since the mechanism is maximum-in-range, it is truthful. We will show that it has polynomial running time and an approximation factor of $\alpha(1 + \epsilon)$.

Running Time. We show that the social welfare maximization over \mathcal{R} is solvable within polynomial time, hence our maximum-in-range mechanism is efficient. We will first show $|\mathcal{R}_0|$ is polynomial in n .

Lemma 1. $|\mathcal{R}_0| \leq |T| \leq n^{1/\log_2 \beta + M/\log_2(1/\eta)}$.

Proof. The first inequality follows from the definition of \mathcal{R}_0 . Now we prove the second one. Observe that for each vector v in T_1 , $E(v)$ is uniquely determined by the values $\{v_{n_k} : 0 \leq k \leq \lfloor \log_\beta n \rfloor\} \subseteq Q \cup \{0\}$. Moreover, we have that $v_{n_{k-1}} \geq v_{n_k}$ for all k . Therefore, let H be the class of non-increasing functions from $\{0, 1, \dots, \lfloor \log_\beta n \rfloor\}$ to $Q \cup \{0\}$, thus $|T| \leq |H|$. Since $|Q| = \lceil \log_{1/\eta}(n^M) \rceil$, It is not difficult to see, $|H| \leq \binom{\lceil \log_\beta n \rceil + \lceil \log_{1/\eta}(n^M) \rceil}{\lceil \log_{1/\eta}(n^M) \rceil} \leq 2^{\lceil \log_\beta n \rceil + \lceil \log_{1/\eta}(n^M) \rceil} \leq n^{1/\log_2 \beta + M/\log_2(1/\eta)}$.

Now we are ready to prove the running time guarantee. Let $\text{opt}_{\mathcal{R}}(v)$ be the allocation maximizes the social welfare over \mathcal{R} for the given bid vector v . Let σ be the permutation such that $v_{\sigma(1)} \geq \dots \geq v_{\sigma(n)}$. Obviously, for each $x \in \mathcal{R}_0$, we have $v \cdot (\sigma^{-1} \circ x) \geq v \cdot (\pi \circ x)$ for all permutation π . Therefore, $\text{opt}_{\mathcal{R}}(v) \in \{\sigma^{-1} \circ x : x \in \mathcal{R}_0\}$. By Lemma 1, $|\{\sigma^{-1} \circ x : x \in \mathcal{R}_0\}| = |\mathcal{R}_0| \leq n^{1/\log_2 \beta + M/\log_2 \eta}$, this implies that $\text{opt}_{\mathcal{R}}(v)$ can be found in polynomial time.

Approximation Factor. We show that the approximation factor of our mechanism is $\alpha(1 + \epsilon)$. Given any bid vector \mathbf{v} , by reordering and scaling properly, we may assume $\mathbf{v} \in T_0$, then we consider the typical bid $E(D(\mathbf{v}))$. We show that for any sorted allocation \mathbf{x} , the social welfare $\mathbf{v} \cdot \mathbf{x}$ is $(1 + \epsilon)$ -approximated by $E(D(\mathbf{v})) \cdot \mathbf{x}$, hence an α -approximate solution for social welfare maximization with respect to $E(D(\mathbf{v}))$ is an $\alpha(1 + \epsilon)$ -approximate solution for \mathbf{v} . This proves the desired approximation guarantee.

Now we provide the detail. We first show that by considering $D(\mathbf{v})$ instead of $\mathbf{v} \in T_0$, the social welfare is rounded down by at most a factor of $\eta(1 - \epsilon/4)$.

Lemma 2. *For any $\mathbf{v} \in T_0$ and any allocation \mathbf{x} s.t. $x_1 \geq \dots \geq x_n$, we have $D(\mathbf{v}) \cdot \mathbf{x} \leq \mathbf{v} \cdot \mathbf{x} \leq \frac{1}{\eta(1-\epsilon/4)} D(\mathbf{v}) \cdot \mathbf{x}$.*

Proof. The first inequality holds by definition. Now we prove the second one. We first show the following lemma which says that the social welfare affected by ‘‘tail cutting’’ is bounded by a fraction of $\epsilon/4$. The proof of the lemma is deferred to full version due to space reasons.

Lemma 3. $\sum_{i:v_i \geq 1/n^M} v_i x_i \geq (1 - \epsilon/4) \sum_{i=1}^n v_i x_i$.

Let $\mathbf{v}' = D(\mathbf{v})$, it is easy to see: $\mathbf{v}' \cdot \mathbf{x} = \sum_{i:v_i \geq 1/n^M} v'_i x_i \geq \eta \sum_{i:v_i \geq 1/n^M} v_i x_i \geq \eta(1 - \frac{\epsilon}{4}) \mathbf{v} \cdot \mathbf{x}$.

Secondly, we show that the social welfare increases by at most a factor of β by considering $E(\mathbf{v})$ instead of \mathbf{v} for any $\mathbf{v} \in T_1$.

Lemma 4. *For any $\mathbf{v} \in T_1$ and any allocation \mathbf{x} s.t. $x_1 \geq \dots \geq x_n$, we have $\mathbf{v} \cdot \mathbf{x} \leq E(\mathbf{v}) \cdot \mathbf{x} \leq \beta \mathbf{v} \cdot \mathbf{x}$.*

Proof. The first inequality is implied by the definition of E . We will prove the second one. Let $\mathbf{v}' = E(\mathbf{v})$ and $L = \lceil \log_{1/\eta} n \rceil$. Since $\mathbf{v}, \mathbf{v}' \in T_1$, we have that $v_i, v'_i \in Q$ for each i . Thus, if we let $l_i = \log_{\eta} v_i$ and $l'_i = \log_{\eta} v'_i$ respectively for each i , then l_i 's and l'_i 's are non-decreasing sequences. By our construction, it is easy to see: (1) for all $1 \leq i \leq n$, $l'_i \leq l_i$; (2) for all $l \in [0, L]$, $|\{i : l'_i \leq l\}| \leq \beta |\{i : l_i \leq l\}|$. Since $x_1 \geq \dots \geq x_n$, we have the following:

Lemma 5. *For any $l \in [0, L]$ and $1 \leq i \leq n$, $\sum_{i:l'_i \leq l} x_i \leq \beta \sum_{i:l_i \leq l} x_i$.*

Observe that if we define $W_l = \eta^l$ for $0 \leq l \leq L$ and $W_{L+1} = 0$, then $\sum_{i=1}^n x_i v_i = \sum_{i=1}^n x_i W_{l_i} = \sum_{i=1}^n x_i \sum_{l=l_i}^L (W_l - W_{l+1}) = \sum_{l=0}^L (W_l - W_{l+1}) \sum_{i:l_i \leq l} x_i$.

Similarly, we have $\sum_{i=1}^n x_i v'_i = \sum_{l=0}^L (W_l - W_{l+1}) \sum_{i:l'_i \leq l} x_i$. By Lemma 5, we have $\mathbf{v}' \cdot \mathbf{x} \leq \beta \mathbf{v} \cdot \mathbf{x}$.

By Lemma 2 and Lemma 4, we have the following:

Corollary 2. *For any $\mathbf{v} \in T_0$ and any allocation \mathbf{x} such that $x_1 \geq \dots \geq x_n$, we have: $\eta(1 - \epsilon/4) \mathbf{v} \cdot \mathbf{x} \leq E(D(\mathbf{v})) \cdot \mathbf{x} \leq \beta \mathbf{v} \cdot \mathbf{x}$.*

Now we prove the approximation guarantee of our mechanism. Given any bid vector \mathbf{v} , without loss of generality, we may assume $\mathbf{v} \in T_0$. Let \mathbf{z}^* be the optimal solution of social welfare maximization for \mathbf{v} and \mathbf{x}^* be the solution output by our mechanism. In addition, let \mathbf{y}^* be the optimal solution for the typical bid $E(D(\mathbf{v}))$. Then, by Corollary 2, we have $\mathbf{v} \cdot \mathbf{x}^* \geq \frac{1}{\beta} E(D(\mathbf{v})) \cdot \mathbf{x}^*$. Since our algorithm is maximum-in-range, allocation \mathbf{x}^* is at least as good as the allocation by algorithm \mathcal{A} with respect to typical bid vector $E(D(\mathbf{v}))$. Hence, we have $E(D(\mathbf{v})) \cdot \mathbf{x}^* \geq \frac{1}{\alpha} E(D(\mathbf{v})) \cdot \mathbf{y}^*$. Further, by optimality of \mathbf{y}^* and Corollary 2, we have $E(D(\mathbf{v})) \cdot \mathbf{y}^* \geq E(D(\mathbf{v})) \cdot \mathbf{z}^* \geq \eta \left(1 - \frac{\epsilon}{4}\right) \mathbf{v} \cdot \mathbf{z}^*$.

In all, we have $\mathbf{v} \cdot \mathbf{x}^* \geq \frac{\eta}{\alpha\beta} \left(1 - \frac{\epsilon}{4}\right) \mathbf{v} \cdot \mathbf{z}^*$. Since we choose β and η such that $\beta/\eta = 1 + \epsilon/2$, we have $\mathbf{v} \cdot \mathbf{z}^* \leq \alpha(1 + \epsilon/2) \mathbf{v} \cdot \mathbf{x}^*/(1 - \epsilon/4) \leq \alpha(1 + \epsilon) \mathbf{v} \cdot \mathbf{x}^*$. This completes our analysis.

4 General SPMD: Limitation of MIR Mechanisms

In the previous section, we study the black-box reductions in the symmetric single-parameter setting. In this section, we give some negative results for factor-preserving black-box reduction in general single-parameter mechanism design (SPMD). We derive a significant approximability gap between maximum-in-range mechanisms and approximation algorithms in the most general single-parameter setting. To do this, we establish a novel relation between SPMD and maximum constraint satisfaction problems (MAXCSP), and show that the approximation ratio of a MIR mechanism for some SPMD problem can be arbitrarily worse than that of the best approximation algorithm for the “corresponding” MAXCSP problem.

Specifically, for every MAXCSP problem Γ that is NP-Hard, we set up a corresponding SPMD problem Γ' , mapping (which can be done in polynomial time) each instance $\mathcal{I} \in \Gamma$ to a profile of agent valuation $v_{\mathcal{I}}$, while $\text{opt}_{\Gamma}(\mathcal{I}) = \text{opt}_{\Gamma'}(v_{\mathcal{I}})$. For every (efficient) MIR mechanism, we show that unless $\text{NP} \subseteq \text{P/poly}$, the approximation guarantee of the mechanism (on Γ') can be no better than that of a random assignment for the corresponding MAXCSP problem Γ , and therefore is arbitrarily worse than the guarantee of the best approximation algorithms, for some carefully chosen Γ .

For the sake of exposition, we choose Γ to be MAX k -ALLEQUAL (which can be any MAXCSP problem, although the gap between performance of MIR mechanisms and that of approximation algorithms might be different). The MAX k -ALLEQUAL problem is defined as follows.

Definition 1 (MAX k -ALLEQUAL). *Given a set C of clauses of the form $l_1 \equiv l_2 \equiv \dots \equiv l_k$ (k constant), where each literal l_i is either a Boolean variable x_j or its negation \bar{x}_j . The goal is to find an assignment to the variables x_i so as to maximize the number of satisfied clauses.*

The MAX k -ALLEQUAL problem is NP-Hard. In fact, it is NP-Hard to approximate MAX k -ALLEQUAL problem within a factor of $2^{\epsilon\sqrt{k}}/2^k$ for some constant

$c > 0$, according to [23,16,11]. On the algorithmic side, there is an $\Omega(k/2^k)$ -approximation algorithm for MAX k -ALLEQUAL shown in [6]. The algorithm based on SDP-relaxation and randomized rounding, but it can be efficiently derandomized by embedding the SDP solution to a low dimensional space, via derandomized Johnson-Lindenstrauss transformation [17,8,19].

CSP-Based Hard Instance for MIR Mechanisms. We describe the corresponding SPMD problem for MAX k -ALLEQUAL as follows. For a MAX k -ALLEQUAL problem with n variables, we set up $M = (2n)^k$ agents in $M_{\text{MAX } k\text{-ALLEQUAL}}$, each corresponding to a clause $c : l_1 \equiv l_2 \equiv \dots \equiv l_k$. Recall that in a SPMD problem, each agent’s valuation is a single real number that specifies its utility for being served. We conclude the description of our hard instance by defining the winner set, i.e. the set of feasible subset of agents being served. For any Boolean assignment $x : [n] \rightarrow \{\text{true}, \text{false}\}$, let $C(x) \subseteq [M]$ be the set of clauses that are satisfied by x . We define the set of feasible allocation functions $Y \subseteq \{\mathbf{y} : [M] \rightarrow \{0, 1\}\}$ to be $Y = \{\mathbf{1}_{C(x)} | x : [n] \rightarrow \{\text{true}, \text{false}\}\}$.

Given a MAX k -ALLEQUAL instance \mathcal{I} with set C of clauses, define the valuation function for the agents, $\mathbf{v}_{\mathcal{I}} = \mathbf{v} : [M] \rightarrow \{0, 1\}$, to be the indicator function of C , i.e. $\mathbf{v}(c) = \mathbf{1}_C(c) = \begin{cases} 1 & c \in C \\ 0 & \text{otherwise} \end{cases}$.

Note that here we assume that every clause appears at most once in C . But the hard instance can be easily generalized to weighted case, by letting $\mathbf{v}(c)$ be the weight of clause c .

Analysis. It’s easy to check the following fact.

Fact 1. $\text{opt}(\mathcal{I}) = \max_{x: [n] \rightarrow \{\text{true}, \text{false}\}} \{\mathbf{v} \cdot \mathbf{1}_{C(x)}\} = \max_{\mathbf{y} \in Y} \{\mathbf{v} \cdot \mathbf{y}\} = \text{opt}(\mathbf{v}_{\mathcal{I}})$.

Now, we prove that there is a significant gap between the approximation guarantee of any MIR mechanism and that of the approximation algorithms. The following theorem shows that MIR mechanism performs $\Omega(k)$ times worse than the approximation algorithm for the corresponding algorithmic task, for any constant $k > 0$.

Theorem 2. *Assuming $\text{NP} \not\subseteq \text{P/poly}$, there is no polynomial time MIR mechanism with approximation ratio better than $2(1 + \epsilon)/2^k$, for any constant $\epsilon > 0$.*

The detailed proof of Theorem 2 is deferred to full version, due to space reasons. At high level, the proof of Theorem 2 consists of two steps. Assuming there is an MIR mechanism with range \mathcal{R} achieving $2(1+\epsilon)/2^k$ approximation guarantee, we firstly show that \mathcal{R} needs to be exponentially large. Then we use Sauer-Shelah Lemma to argue that when \mathcal{R} is sufficiently large, it must cover all possible assignments for a constant fraction of the n variables in MAX k -ALLEQUAL, and we can use this mechanism exactly solve MAX k -ALLEQUAL problem on this fraction of variables, which is NP-Hard.

The above technique was first introduced in [5] to show the inapproximability result in combinatorial auctions. However, their construction relies on the complicated private structures of agents’ valuations, hence does not apply in our

problem. Our approach can be viewed as a novel generalization of their technique in single-parameter mechanism design. To our knowledge, this is the first example of a lower bound on MIR mechanisms for problems that are linear and succinct.

5 Symmetric Multi-Parameter Mechanism Design

As a natural generalization of single-parameter mechanism design, we consider the multi-parameter problems in this section. Due to space reasons, proofs of the theorems in this section are deferred to full version of this paper.

As before, a problem is *symmetric* if $\mathbf{S} = (S_1, \dots, S_n)$ is a feasible allocation implies that $\pi \circ \mathbf{S} = (S_{\pi(1)}, \dots, S_{\pi(n)})$ is also a feasible allocation for any permutation π . Moreover, a mechanism design problem is Δ -dimension if the valuation of each agent i can be naturally represented by a Δ -dimension vector $\mathbf{u}_i = (u_{i1}, \dots, u_{i\Delta}) \in \mathbb{R}_+^\Delta$. We let $v(S, \mathbf{u})$ denote an agent's value of an allocation S when its valuation function is given by a Δ -dimension vector \mathbf{u} . We will assume that the problem satisfies the following properties:

- **Monotonicity.** For any $1 \leq i \leq n$, \mathbf{S} , \mathbf{u}_i and \mathbf{u}'_i such that $u_{ij} \geq u'_{ij}$ for any $1 \leq j \leq \Delta$, we have $v(S_i, \mathbf{u}_i) \geq v(S_i, \mathbf{u}'_i)$.
- **Sub-linear influence.** For any $1 \leq k \leq \Delta$, $\beta > 1$, \mathbf{u} and \mathbf{u}' such that for any $1 \leq i \leq n$, $u_{ij} = u'_{ij}$ for any $j \neq k$, and $u_{ik} \leq \beta u'_{ik}$, we have $\text{opt}(\mathbf{u}) \leq \beta \text{opt}(\mathbf{u}')$.
- **Negligible tail.** For any $\delta > 0$, let \mathbf{u}_i^δ be the tail-truncated values: $u_{ij}^\delta = u_{ij}$ if $u_{ij} \geq \delta \max_{s,t} u_{st}$ and $u_{ij}^\delta = 0$ otherwise. For any constant $\epsilon > 0$, there is a polynomially small $\delta > 0$, so that for any allocation \mathbf{S} and any values \mathbf{u}_i 's, we have $(1 + \epsilon) \sum_{i=1}^n v(S_i, \mathbf{u}_i^\delta) \geq \sum_{i=1}^n v(S_i, \mathbf{u}_i)$.

These assumptions are without loss of generality in many mechanism design problems. For example, consider the following:

- **Multi-item auction.** In multi-item auctions, we consider n agents and m different types of items, each of which has a finite supply. Each agent i has a private m -dimension vector of values $\mathbf{u}_i = (u_{i1}, \dots, u_{im})$. Agent i 's value of a bundle S with x_j items of type j , $1 \leq j \leq m$, is $v(S, \mathbf{u}_i) = \sum_{j=1}^m x_j u_{ij}$. This is a m -dimensional problem that satisfies our assumptions.
- **Combinatorial auction.** In combinatorial auctions, we consider n agents and m different items. Each agent i has a private 2^m -dimension vector \mathbf{u}_i so that for each subset of items $S \in 2^{[m]}$, agent i 's value of bundle S is $v(S, \mathbf{u}_i) = u_{iS}$. This is a 2^m -dimensional problem that satisfies our assumptions.

Via techniques similar to those in Section 3, we can show the following theorem. The proofs are deferred to the full version of this paper.

Theorem 3. *For any Δ -dimension symmetric mechanism design problem Π where Δ is a constant, suppose \mathcal{A} is an α -approximate algorithm, then for any constant $\epsilon > 0$, we can get an truthful and $(1 + \epsilon)\alpha$ -approximate mechanism that runs in quasi-polynomial time given \mathcal{A} as a black-box.*

Alternatively, we can alleviate the running time by having greater degrade in the approximation factor.

Theorem 4. *For any Δ -dimension symmetric mechanism design problem Π where Δ is a constant, suppose \mathcal{A} is an α -approximate algorithm, then for any constant $\epsilon > 0$, we can get a truthful and α polylog-approximate mechanism that runs in polynomial time given \mathcal{A} as a black-box.*

Acknowledgments. The authors would like to thank Nina Balcan, Yang Cai, Deeparnab Chakrabarty, Florin Constantin, Gagan Goel, Venkatesan Guruswami, Vijay Vazirani and Jiajin Yu for many useful comments and helpful discussions.

References

1. Babaioff, M., Lavi, R., Pavlov, E.: Single-value combinatorial auctions and implementation in undominated strategies. In: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithm (SODA), pp. 1054–1063. ACM, New York (2006)
2. Balcan, M.-F., Blum, A., Hartline, J.D., Mansour, Y.: Mechanism design via machine learning. In: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 605–614 (2005)
3. Bei, X., Huang, Z.: Bayesian incentive compatibility via fractional assignments. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA (to appear, 2011)
4. Briest, P., Krysta, P., Vöcking, B.: Approximation techniques for utilitarian mechanism design. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing (STOC), pp. 39–48. ACM, New York (2005)
5. Buchfuhrer, D., Dughmi, S., Fu, H., Kleinberg, R., Mossel, E., Papadimitriou, C., Schapira, M., Singer, Y., Umans, C.: Inapproximability for vcg-based combinatorial auctions. In: Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms, SODA (2010)
6. Charikar, M., Makarychev, K., Makarychev, Y.: Near-optimal algorithms for maximum constraint satisfaction problems. *ACM Trans. Algorithms* 5, 1–32 (2009)
7. Clarke, E.H.: Multipart pricing of public goods. *Public Choice* 11(1) (September 1971)
8. Diakonikolas, I., Kane, D.M., Nelson, J.: Bounded independence fools degree-2 threshold functions. In: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science, FOCS (2010)
9. Shaddin, D., Tim, R.: Black-box randomized reductions in algorithmic mechanism design. In: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science, FOCS (2010)
10. Dughmi, S., Roughgarden, T., Yan, Q.: From convex optimization to randomized mechanisms: Toward optimal combinatorial auctions for submodular bidders. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, STOC (2011)
11. Engebretsen, L., Holmerin, J.: More efficient queries in PCPs for NP and improved approximation hardness of maximum CSP. *Random Structures & Algorithms* 33(4), 497–514 (2008)

12. Goel, G., Karande, C., Wang, L.: Single-parameter combinatorial auctions with partially public valuations. In: Kontogiannis, S., Koutsoupias, E., Spirakis, P.G. (eds.) *Algorithmic Game Theory*. LNCS, vol. 6386, pp. 234–245. Springer, Heidelberg (2010)
13. Groves, T.: Incentives in teams. *Econometrica* 41(4), 617–631 (1973)
14. Hartline, J., Kleinberg, R., Malekian, A.: Bayesian incentive compatibility via matchings. In: *Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA* (to appear, 2011)
15. Hartline, J.D., Lucier, B.: Bayesian algorithmic mechanism design. In: *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pp. 301–310. ACM, New York (2010)
16. Håstad, J., Wigderson, A.: Simple analysis of graph tests for linearity and pcp. *Random Structures and Algorithms* 22(2), 139–160 (2003)
17. Daniel, M.: Kane and Jelani Nelson. A derandomized sparse Johnson-Lindenstrauss transform. *Arxiv preprint arXiv:1006.3585* (2010)
18. Lavi, R., Swamy, C.: Truthful and near-optimal mechanism design via linear programming. In: *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 595–604. IEEE Computer Society, Washington, DC, USA (2005)
19. Meka, R., Zuckerman, D.: Pseudorandom generators for polynomial threshold functions. In: *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC)*, pp. 427–436 (2010)
20. Myerson, R.: Optimal auction design. *Mathematics of Operations Research* 6(1), 58–73 (1981)
21. Nisan, N., Ronen, A.: Algorithmic mechanism design. In: *Proceedings of the 31st Annual ACM Symposium on Theory of Computing (STOC)*, pp. 129–140 (1999)
22. Papadimitriou, C., Schapira, M., Singer, Y.: On the hardness of being truthful. In: *In 49th Annual IEEE Symposium on Foundations of Computer Science, FOCS* (2008)
23. Samorodnitsky, A., Trevisan, L.: A PCP characterization of NP with optimal amortized query complexity. In: *Proceedings of the 32nd Annual ACM Symposium on Theory of Computing (STOC)*, pp. 191–199. ACM, New York (2000)
24. Vickrey, W.: Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance* 16(1), 8–37 (1961)
25. Vondrak, J.: Optimal approximation for the submodular welfare problem in the value oracle model. In: *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 67–74. ACM, New York (2008)

Multiplicative Approximations of Random Walk Transition Probabilities

Michael Kapralov^{1,*} and Rina Panigrahy²

¹ Stanford University, Stanford CA 94305, USA
kapralov@stanford.edu

² Microsoft Research Silicon Valley, Mountain View, CA, 94043, USA
rina@microsoft.com

Abstract. We study the space and time complexity of approximating distributions of l -step random walks in simple (possibly directed) graphs G . While very efficient algorithms for obtaining *additive* ϵ -approximations have been developed in the literature, no non-trivial results with multiplicative guarantees are known, and obtaining such approximations is the main focus of this paper. Specifically, we ask the following question: given a bound S on the space used, what is the minimum threshold $t > 0$ such that l -step transition probabilities for all pairs $u, v \in V$ such that $P_{uv}^l \geq t$ can be approximated within a $1 \pm \epsilon$ factor? How fast can an approximation be obtained?

We show that the following surprising behavior occurs. When the bound on the space is $S = o(n^2/d)$, where d is the minimum out-degree of G , no approximation can be achieved for non-trivial values of the threshold t . However, if an extra factor of s space is allowed, i.e. $S = \tilde{\Omega}(sn^2/d)$ space, then the threshold t is *exponentially small in the length of the walk l* and even very small transition probabilities can be approximated up to a $1 \pm \epsilon$ factor. One instantiation of these guarantees is as follows: any almost regular directed graph can be represented in $\tilde{O}(ln^{3/2+\epsilon})$ space such that all probabilities larger than n^{-10} can be approximated within a $(1 \pm \epsilon)$ factor as long as $l \geq 40/\epsilon^2$. Moreover, we show how to estimate of such probabilities faster than matrix multiplication time.

For undirected graphs, we also give small space oracles for estimating P_{uv}^l using a notion of bicriteria approximation based on approximate distance oracles of Thorup and Zwick [STOC'01].

1 Introduction

Large scale graphs are now a widely used tool for representing real world data. Many modern applications, such as search engines or social networks, require supporting various queries on large-scale graphs efficiently. An important operation is calculating some measure of distance between two nodes in a graph. Random walks recently emerged as an important tool for measuring distance between nodes in such graphs. PageRank [17] and its personalized version is one of the popular random walk based measures of relatedness of nodes in a graph. Personalized PageRank uses the distribution of a short

* Supported in part by a Stanford Graduate Fellowship and NSF award IIS-0904325. Part of the work was done while the author was an intern at Microsoft Research Silicon Valley.

random walk of exponentially distributed length started at a source node to measure proximity between the source and other nodes in the graph. Methods for computing personalized PageRank have received significant attention recently. Very efficient algorithms are known for computing and updating (personalized) Pagerank vectors in various computation models such as random access and streaming (see, e.g. [12][13][121][22][2]). The techniques that have been used range from methods from linear algebra to Monte Carlo simulation and dynamic programming. It is hard to do justice to the large body of work on this topic in limited space, and we refer the reader to the great surveys [14][4] and recent papers on Pagerank computation (see, e.g. [2]) for a more complete discussion of prior work on Pagerank. The approximation obtained via these methods is usually an *additive* ϵ -approximation for some $\epsilon > 0$. With this measure of approximation, it is in general sufficient to perform $O(\log n/\epsilon)$ random walks of small length to obtain an approximation. While additive approximations are well-studied, no non-trivial multiplicative $(1 \pm \epsilon)$ -factor approximations are known, and obtaining such approximations using small space and time is the main focus of this paper.

Given a simple (possibly directed) graph $G = (V, E)$ and a bound S on the space we can use, we would like to obtain $(1 \pm \epsilon)$ -factor approximations of l -step transition probabilities P_{uv}^l from u to v , for all pairs $u, v \in V$ such that $P_{uv}^l \geq t$, for the smallest possible threshold value t . We show that the following surprising behavior occurs. If the minimum out-degree of the graph G is d , then no $(1 \pm \epsilon)$ -factor approximation can be achieved for any reasonable value of t if the space S available is below $o(n^2/d)$. However, if the available space is a factor $s > 1$ larger than $\tilde{\Omega}(n^2/d)$, then $(1 \pm \epsilon)$ -factor approximation can be achieved for probabilities larger than $t \geq \frac{1}{d}s^{-(\epsilon/4)(l-1)}$. Thus, increasing the amount of space available by a factor of $s > 1$ allows one to approximate transition probabilities that are *exponentially small in the length of the walk* l . One instantiation of such guarantees is that any regular graph can be represented in $\tilde{O}(ln^{3/2+\epsilon}/\epsilon^2)$ space so that $(1 \pm \epsilon)$ -approximations to transition probabilities P_{uv}^l can be recovered for all pairs $u, v \in V$ such that $P_{uv}^l \geq n^{-10}$ as long as $l \geq 40/\epsilon^2$.

These bounds are nearly-tight: we show that the space complexity of obtaining $(1 \pm \epsilon)$ -approximations to P_{uv}^l for all pairs such that $P_{uv}^l \geq \frac{1}{d}(s/2)^{-(l-1)}$ is $\Omega(sn^2/d)$. Additionally, our techniques yield fast algorithms for calculating very precise approximations to l -step random walk transition probabilities for special classes of graphs. For example, it follows that an $(1 \pm \epsilon)$ -factor approximation to P_{uv}^l for all pairs such that $P_{uv}^l \geq n^{-10}$, say, can be obtained in time $\tilde{O}(n^{2+(1+\epsilon)\frac{\omega-2}{\omega-1}})$, where $\omega \geq 2$ is the matrix multiplication constant, for any (almost) regular graph. Thus, we show that very precise approximations can be obtained strictly faster than matrix multiplication if $\omega > 2$. A very interesting question raised by our work is whether our techniques can be used to obtain similar approximations to the problem of multiplying two matrices with non-negative entries faster than matrix multiplication time. We note that besides being of intrinsic theoretical interest, multiplicative approximation guarantees for all transition probabilities above a very small threshold may be useful, for example, in applications to probabilistic model checking, where one is interested in the probability that a Markov chain reaches an ‘error state’ in a given number of steps ([19]).

We also consider the problem of obtaining approximate oracles for random walk transition probabilities for undirected graphs. While almost-optimal tradeoffs for the

problem of approximating distances have been obtained in the large body of work on spanners ([6,7,10,23,25,3,24,18]), we are not aware of any known results on small space oracles for approximating random walk transition probabilities. Our lower bounds on the space complexity of multiplicative approximation suggest that a weaker notion of approximation is required if close to linear space is desired. A natural candidate notion is a relaxation of the length of the random walk similar to the approximations obtained in the well-known distance oracles of Thorup and Zwick[23]. Surprisingly, it turns out that such relaxation is too strong for our problem. However, we show that a more relaxed notion of bicriteria approximation can be used to obtain small space oracles based on the techniques of [23].

1.1 Related Work

Approximate Matrix Multiplication. It is interesting to compare our results on multiplicative approximation to the line of work on approximate matrix computations, which develops extremely efficient approximate randomized algorithms for several problems in numerical linear algebra (e.g. [8,11,9,20,16,5,15]). A lot of these algorithms can be used in the streaming model, requiring close to optimal space and only a small constant number of passes over the data. These algorithms are also very general and apply to arbitrary matrices. There is a vast literature on this topic, and we refer the reader to some of the latest papers for a more complete list of references (see, e.g. [15]). All of the developed algorithms for approximating matrix products yield approximation guarantees in terms of the Frobenius or spectral norm. To highlight the relation to our results, we now compare our algorithm to the earlier algorithm of [8], which is also based on column/row sampling. In [8] the authors show that for any $s > 1$ there exists a randomized algorithm that given two matrices $A, B \in \mathbb{R}^{n \times n}$ outputs a matrix P such that

$$\mathbf{E}[\|P - A \cdot B\|_F^2] \leq \frac{1}{s} \|A\|_F^2 \|B\|_F^2 \quad (1)$$

in time $\tilde{O}(n^2 s)$ by carefully sampling rows of A and columns of B . On the other hand, our multiplication approximation algorithm computes the l -th power of a matrix A , giving entrywise approximation guarantees. Let A be the random walk transition matrix of an (almost)regular graph. We show how to calculate $(1 \pm \epsilon)$ -factor approximation to every sufficiently large entry of A^l in time $\tilde{O}(n^{2+(1+\epsilon)\frac{l-2}{l-1}})$ for any sufficiently large l . It should be noted that the best speedups that our techniques currently yield are for a special class of graphs. On the other hand, we obtain a significantly more precise estimate than (1) faster than matrix multiplication time. Since $\|A\|_F$ can in general be very large, there is no way of setting the parameter s in the algorithm of (1) that would allow to obtain such precision faster than matrix multiplication.

It is interesting to note that the approach we take also uses row and column sampling. However, the process is somewhat different: we show that for each pair $u, v \in V$ such that the (u, v) entry of A^l is sufficiently large there exists j , $1 \leq j \leq l-1$ such that sampling rows of A^j and columns of A^{l-j} at an appropriate rate allows one to get a $1 \pm \epsilon$ approximation for A_{uv}^l . It seems plausible that similar techniques could be used to obtain good approximations to the product of two matrices with non-negative entries

faster than matrix multiplication time. This currently remains a very interesting open problem.

Another connected line of work concerns efficiently representing distances between nodes of a weighted undirected graph.

Distance Oracles. A large body of work on spanners and emulators provides almost optimal tradeoffs for the case when the distance measure of interest is the shortest path distance. A *spanner* is a subgraph H of the target graph G that approximates the shortest path metric on G in some sense. Several notions of spanners have been considered in the literature. A *multiplicative spanner* is a subgraph H of G such that $\delta_G(u, v) \leq \delta_H(u, v) \leq t\delta_H(u, v)$ for some constant $t > 1$ called the *stretch* of H . The landmark paper of [23] provides almost optimal tradeoffs between the size of H and the stretch, assuming a widely believed girth conjecture of Erdős. In particular, [23] construct stretch $2k - 1$ -spanners with $O(n^{1+1/k})$ edges. Moreover, their construction has $O(k)$ query time, thus yielding an approximate distance oracle. Very recently, [18] gave an oracle that returns a path of length at most $2\delta_G(u, v) + 1$ and uses $O(n^{5/3})$ space for unweighted graphs. Related constructions, such as additive spanners and emulators (e.g. [24]), yield better approximation/space tradeoffs. However, no fast method for answering distance queries using these constructions is known.

1.2 Our Results and Techniques

Directed Graphs

For a graph $G = (V, E)$ and a pair of nodes $u, v \in V$ we denote the probability of reaching v from u in l steps of a simple random walk on G by P_{uv}^l .

Our main result is

Theorem 1. *Let $G = (V, E)$ be a (possibly directed) graph with minimum out-degree bounded below by d . For any $s \geq 1$ there exists a data structure that allows returning $1 \pm \epsilon$ -factor approximations to P_{uv}^l for all $u, v \in V$ such that $P_{uv}^l \geq (1/d)s^{-(\epsilon/4)(l-1)}$ using space $\tilde{O}\left(\frac{lsn^2}{\epsilon^2 d}\right)$. Moreover, whp for any u, v the approximation to P_{uv}^l that is returned does not overestimate P_{uv}^l by more than a $1 + \epsilon$ factor. The query time is $\tilde{O}(lns/(\epsilon^2 d))$ and the preprocessing time is $\tilde{O}(l \cdot \min\{n^\omega, lsn^3/(\epsilon^2 d)\})$.*

Remark 1. Note that the preprocessing time is $o(n^\omega)$ for $d = \Omega(sn^{3-\omega})$ and $l = \tilde{O}(1)$.

Note that the threshold above which we can provide $1 \pm \epsilon$ -approximations to P_{uv}^l depends exponentially on the length of the walk l . Thus, even for moderate values of s Theorem 1 potentially allows to approximate transition probabilities for all pairs $u, v \in V$. One instantiation of Theorem 1 is as follows:

Corollary 1. *Let $G = (V, E)$ be a regular graph. Then all l -step transition probabilities $P_{uv}^l \geq n^{-10}$ can be approximated up to $1 \pm \epsilon$ for any $\epsilon > 0$ in space $\tilde{O}(ln^{3/2+\epsilon}/\epsilon^2)$ as long as $l \geq 40/\epsilon^2$.*

Proof. Set $s = n^\epsilon$. If $d = \Omega(n^{1/2+\epsilon})$, then Theorem 1 can be used to compress the representation to $\tilde{O}(ln^{2+\epsilon}/\epsilon^2 d) = \tilde{O}(ln^{3/2+\epsilon}/\epsilon^2)$ space. Otherwise one can store the entire graph in $\tilde{O}(n^{3/2+\epsilon})$ space. \square

Remark 2. Perhaps the most natural interpretation of the guarantees given by our algorithm is as follows. Given the compressed representation of the graph, if for a query (u, v) the value \hat{P}_{uv}^l returned by the algorithm is large (i.e. satisfies $\hat{P}_{uv}^l \geq n^{-10}$), then it is guaranteed to be within $(1 \pm \epsilon)$ factor of the true transition probability. Otherwise, the true transition probability is very small, i.e. below n^{-10} .

Note that it is entirely possible that all l -step transition probabilities in a regular graph are larger than n^{-10} , in which case one can approximate each entry of A^l up to an $1 \pm \epsilon$ factor, where A is the random walk transition matrix of G . Interestingly, our algorithm yields a method for approximating sufficiently large l -step transition probabilities in a regular graph faster than matrix multiplication:

Corollary 2. $P_{uv}^l \geq n^{-10}$ can be approximated up to $1 \pm \epsilon$ for any $\epsilon > 0$ in time $\tilde{O}(ln^{2+(1+\epsilon)(\omega-2)/(\omega-1)})$ as long as $l \geq 40/\epsilon^2$

We show that the upper bound given by Algorithm 2 is of the right form:

Theorem 2. Any algorithm that gives a constant factor approximation to $P_{uv}^l \geq \frac{1}{d_u}(s/2)^{-(l-1)}$ in an undirected graph $G = (V, E)$ with minimum degree d must use $\Omega\left(\frac{sn^2}{l^2d}\right)$ space.

The proof of Theorem 2 is deferred to the full version of the paper.

Undirected Graphs

All proofs from this section are deferred to the full version of the paper due to space considerations. The lower bound given by Theorem 2 suggests that near-linear space cannot be achieved independently of the minimum degree of the graphs if a constant factor approximation to P_{uv}^l is desired. In light of the result of [23] on $2k - 1$ -approximate distance oracles for undirected graphs in space $O(n^{1+1/k})$ it is natural to conjecture that one can output a constant factor approximation to the probability of reaching v from u in $(2j - 1)t$ steps for some $1 \leq j \leq k$ in $O(n^{1+1/k})$ space. Perhaps surprisingly, we show that such approximation cannot be achieved for random walk transition probabilities:

Lemma 1. Any algorithm that given a weighted graph $G = (V, E)$ for any pair of nodes (u, v) outputs an estimate $\hat{p}(u, v)$ such that

$$P_{u,v}^l \leq \hat{p} \leq (1 + \epsilon) \max_{1 \leq j \leq 2k-1} P_{u,v}^{jl}$$

for any constant $\epsilon > 0$ must use $\Omega(n^2/2^{kl})$ space.

Lemma 1 motivates more relaxed bicriteria approximation. In particular, when queried about the value of $P_{u,v}^l$, the algorithm may return an approximation to the logarithm of $P_{u,v}^{jl}$ for any $1 \leq j \leq 2k - 1$. The following theorem shows that a somewhat more powerful notion of approximation is possible:

Theorem 3. Let $G = (V, E)$ be a weighted undirected graph such that $1/\gamma \leq d(u)/d(v) \leq \gamma$ for all $u, v \in V$ for some constant $\gamma \geq 1$. There exists an $O(k^3 n^{1+1/k} \log n)$ space and $O(k^3 \log n)$ query time oracle that given a pair $u, v \in V$ outputs a value \hat{p} such that

$$\gamma^{-1} P_{u,v}^l \leq \hat{p} \leq 4 \max_{1 \leq j \leq k} \left(P_{u,v}^{(2j-1)l} / n^{(j-1)/k} \right)^{1/(2j-1)}.$$

The preprocessing time is bounded by $\tilde{O}(kln^\omega)$, where $\omega \geq 2$ is the matrix multiplication constant.

Note that the difference between the statement of Theorem 3 differs from the notion of bicriteria approximation outlined above in the extra normalization term $n^{(j-1)/k}$, which only improves the guarantee. The algorithm is based on approximate distance oracles of Thorup and Zwick.

This approximation ratio is optimal up to an $O(\log n)$ factor in the following sense:

Theorem 4. Any algorithm that outputs an estimate \hat{p} such that

$$P_{u,v}^l \leq \hat{p} \leq (1 + \epsilon) \max_{1 \leq j \leq k} \left(P_{u,v}^{(2j-1)l} / n^{(j-1)/k} \right)^{1/(2j-1)}$$

for a constant $\epsilon > 0$ must use $\Omega(n^{1+1/k}/2^{lk})$ space.

Theorem 3 applies to γ -regular graphs. For general graphs, we approximate the symmetrized quantity $S_t(u, v) = \sqrt{P_t(u, v)P_t(v, u)} = \sqrt{d(u)/d(v)}P_t(u, v)$.

2 Directed Graphs

In this section we give an algorithm for approximating random walk transition probabilities in simple graphs $G = (V, E)$, proving Theorem 1. Some proofs from this section are deferred to the full version of the paper due to space considerations.

Given a bound $\tilde{O}(l s n^2/d)$ on the available space, where $s \geq 1$, the preprocessing Algorithm 1 samples $O(\log n/\epsilon)$ sets of centers $C_t \subseteq V$, by including each vertex into C_t uniformly at random with probability $r = \frac{4s \log n}{\epsilon^2(1-\epsilon/2)d}$. For each center $c \in C_t$ the algorithm computes and stores P_{uc}^j and P_{cu}^j for all $u \in V$ and all $1 \leq j \leq l - 1$:

- 1: **for** $t = 1, \dots, 4 \log n/\epsilon$ **do**
- 2: Choose $r = \frac{4s \log n}{\epsilon^2(1-\epsilon/2)d}$ centers C_t uniformly at random.
- 3: For each $c \in C_t$ and for each $j, 1 \leq j \leq l - 1$ calculate and store P_{uc}^j, P_{cu}^j for all $u \in V$.
- 4: **end for**

Algorithm 1. Preprocessing algorithm

At query time, given a pair of vertices $u, v \in V$, for each of the $O(\log n/\epsilon)$ sets of centers C_t and for each $1 \leq j \leq l - 1$ Algorithm 2 calculates an estimate

$$\hat{p}_{t,j} := \frac{1}{r} \sum_{c \in C_t} P_{uc}^j P_{cv}^{l-j}. \tag{2}$$

Finally, the algorithm sets

$$\hat{p}_j^{min} := \min_{1 \leq t \leq 4 \log n / \epsilon} \hat{p}_{t,j} \tag{3}$$

and returns the largest of \hat{p}_j^{min} , $1 \leq j \leq l - 1$ as its estimate.

```

1: for  $t = 1, \dots, 4 \log n / \epsilon$  do
2:    $\hat{p}_{t,j} \leftarrow \frac{1}{r} \sum_{c \in C_t} P_{uc}^j P_{cv}^{l-j}$ .
3: end for
4:  $\hat{p} \leftarrow \max_{j=1, \dots, l-1} \min_{t=1, \dots, 4 \log n / \epsilon} \hat{p}_{t,j}$ 
5: return  $\hat{p}$ 

```

Algorithm 2. Estimation algorithm

The estimator in (2) is unbiased, i.e. $\mathbf{E}[\hat{p}_{t,j}] = P_{uv}^l$ for all t and j since the set of centers was chosen uniformly at random. The question that we consider is for which $u, v \in V$ and for what sampling rate r is \hat{P}_{uv}^l is tightly concentrated around its expectation.

It is easy to construct examples that show that in general there need not be any useful concentration for nontrivial sampling probabilities if $1 \pm \epsilon$ factor approximation of P_{uv}^l is desired for all pairs $u, v \in V$. Moreover, one cannot in general guarantee tight concentration for any fixed j under reasonable conditions on u and v . However, in this section we will show that in fact for each $u, v \in V$ such that P_{uv}^l is sufficiently large the estimate $\hat{p}_{t,j}$ from (2) will not underestimate by more than a $1 - \epsilon$ factor whp for at least one choice of j between 1 and $l - 1$ and for all t . The main technical part of our analysis is analysing conditions under which $\hat{p}_{t,j}$ does not underestimate P_{uv}^l , since overestimation can be easily dealt with by independent repetition.

We first give the intuition behind the analysis. We assume that the graph is undirected and strictly regular for simplicity. Then P_{uv}^l for a pair $u, v \in V$ is just the number of l -step walks from u to v divided by d^l , where d is the degree. We need to relate the number of l -step walks from u to v to the probability that the random sampling in (2) underestimates by more than $1 - \epsilon$ factor. Fix j between 1 and $l - 1$ and for a node $c \in V$ let

$$\alpha(j, c) = \frac{P_{uc}^j P_{cv}^{l-j}}{P_{uv}^l}$$

be the fraction of l -step walks from u to v that pass through c at step j . By Bernstein’s inequality the probability of underestimating by more than a factor of $1 - \epsilon$ can be related to the variance of the sequence $\alpha(j, c)$, $c \in V$. Thus we need to relate the variance of $\alpha(j, \cdot)$ to the number of walks from u to v . In order to do that, we consider a uniformly random l -step walk $P = (X_1, \dots, X_{l-1})$ from u to v , where X_j is the (random) j -th vertex in the walk. The number of paths is thus equal to the entropy $H(P)$. However, by a well-known property of the entropy function we have that

$$H(P) = H(X_1, \dots, X_{l-1}) \leq \sum_{j=1}^{l-1} H(X_j | X_{j-1}, \dots, X_1) \leq \sum_{j=1}^{l-1} H(X_j | X_{j-1}). \tag{4}$$

We now note that the distribution of X_j is given by $\alpha(j, \cdot)$ defined above. Thus, it is sufficient to bound

$$\min_{(X,Y), X \propto \alpha(j, \cdot), (X,Y) \in E} H(X|Y) \tag{5}$$

in terms of ϵ and the variance of $\alpha(j, \cdot)$. Here the minimization is over all random variables (X, Y) taking values in $(V \times V) \cap E$ (corresponding to the j -th and $(j - 1)$ -st vertex on the random walk from u to v) such that the distribution of X is $\alpha(j, \cdot)$. Given such a bound, we conclude that if (2) underestimates by more than a factor $1 - \epsilon$ for all j between 1 and $l - 1$, then there must be very few paths between u and v .

We have given the sketch of the proof for regular graphs, where the regularity allows the use of the entropy function for bounding P_{uv}^l . In the more general case of graphs of minimum degree d we use the relative entropy, or Kullback-Leibler divergence, with respect to the inverse degree sequence of the graph. We now give the details of the proof.

Denote the set of l -step walks in G from u and v by $\mathcal{P}_{uv}^l \subseteq V^{l-1}$. We write $P = (v_1, \dots, v_{l-1}) \in \mathcal{P}_{uv}^l$ to denote the path $(u, v_1, \dots, v_{l-1}, v)$. Let $\mu^l : \mathcal{P}_{uv}^l \rightarrow [0, 1]$ be the natural distribution on paths $P = (v_1, \dots, v_{l-1}) \in \mathcal{P}_{uv}^l$ defined as

$$\mu((v_1, \dots, v_{l-1})) = \begin{cases} \frac{1}{P_{uv}^l} \frac{1}{d_u} \prod_{j=1}^{l-1} \frac{1}{d_{v_j}} & \text{if } (v_1, \dots, v_{l-1}) \in \mathcal{P}_{uv}^l \\ 0 & \text{o.w.} \end{cases}$$

Note that this is indeed a distribution, i.e. $\sum_{P \in \mathcal{P}_{uv}^l} \mu(P) = 1$.

Also, for $P = (v_1, \dots, v_{l-1}) \in V^{l-1}$ define

$$\eta^l((v_1, \dots, v_{l-1})) = \frac{1}{d_u} \prod_{j=1}^{l-2} \frac{1}{d_{v_j}}.$$

For a vertex $v \in V$ define $\eta(v) = \frac{1}{d_v}$. We note that η^l is not a distribution in general.

Recall that for two distributions on a set X the relative entropy of p with respect to q is defined as

$$D(p||q) = \sum_{x \in X} p_x \log(q_x/p_x). \tag{6}$$

Note that (6) is defined even when q is only guaranteed to be non-negative. Note that the usual definition uses $\log(p_x/q_x)$, but we use this one in order to get an inequality of the form (4).

We will use the following:

Claim.

$$P_{uv}^l \leq \frac{1}{d} e^{D(\mu||\eta)}.$$

For two random variables X, Y taking values in V define

$$D^*(X|Y||\eta) = \sum_{y \in V} p(y) \sum_{x \in V} p(x|y) \log(\eta(y)/p(x|y)). \tag{7}$$

We will use the following

Lemma 2. Let (X_1, \dots, X_{l-1}) be a random variable taking values in V^{l-1} . Let $X_0 = u$. Then one has

$$D((X_1, \dots, X_{l-1}) || \eta^l) \leq \sum_{j=1}^{l-1} D^*(X_j | X_{j-1} || \eta)$$

Note that this is the equivalent of (4) in the sketch of the proof for the regular case given above.

Definition 1. For a distribution x on V define

$$\bar{D}(x) = \max_{(X,Y): X \propto x, (X,Y) \in E} D^*(X|Y || \eta).$$

This is the equivalent of (5) in the regular case. We will use the following lemma:

Lemma 3. Let x be a distribution on V . Sort elements of x so that $x_1 \geq x_2 \geq \dots \geq x_n$. Then for any $j, 1 \leq j \leq n$ one has

$$\bar{D}(x) \leq \left(\sum_{i < j} x_i \right) \log \frac{1}{dx_j}.$$

Lemma 4. (Bernstein’s inequality) Let $X = \sum_{i=1}^n X_i$, where $|X_i| \leq M$ almost surely. Then

$$\Pr [X - \mathbf{E}[X] < \epsilon \mathbf{E}[X]] < \exp \left(- \frac{\epsilon^2 \mathbf{E}[X]^2}{\sum_{i=1}^n \mathbf{E}X_i^2 + \epsilon M \mathbf{E}[X]/3} \right)$$

We now prove the main lemma in the analysis:

Lemma 5. If our sample underestimates by a factor larger than $1 - \epsilon$ with probability larger than $1 - n^{-4}$, then for every j between 1 and $l - 1$ we have

$$\bar{D}(\alpha(j, \cdot)) \leq -\eta \log(\epsilon^2(1 - \eta)s/4)$$

for some $\eta \in [\epsilon/4, \epsilon/2]$.

Proof. Let $x_1 \geq \dots \geq x_n, \sum_i x_i = 1$ be the distribution $\alpha(j, \cdot)$ (we numbered vertices of G). Let $X_i = \text{Ber}(s/d, x_i)$. We consider two cases. First suppose that $x_1 \geq \epsilon/4$. Then we have that $\bar{D}(x) \leq -(\epsilon/4) \log(\epsilon d/4)$ and we are done. We now assume that $x_1 \leq \epsilon/4$.

Denote by x^ϵ the subsequence $x_{j^\epsilon} \geq \dots \geq x_n$ such that $\sum_{i=j^\epsilon}^n x_i \leq 1 - \epsilon/4$, where j^ϵ is the maximum possible. Since $x_1 \leq \epsilon/4$, we have that $\sum_{i=j^\epsilon}^n x_i \geq 1 - \epsilon/2$.

By Bernstein’s inequality

$$\begin{aligned} \Pr [X^\epsilon - \mathbf{E}[X^\epsilon] < \delta \mathbf{E}[X^\epsilon]] &< \exp \left(- \frac{\delta^2 (s \log n/d)^2}{(s \log n/d) \|x^\epsilon\|_2^2 + x_{j^\epsilon} (s \log n/3d)} \right) \\ &= \exp \left(- \frac{\delta^2 \log n}{(d/s) \|x^\epsilon\|_2^2 + x_{j^\epsilon} (d/3s)} \right) \end{aligned}$$

Thus, if our sampling underestimates by a factor larger than $1 - \epsilon$, we must have that

$$\|x^\epsilon\|_2^2 + x_{j^\epsilon}/3 \geq \delta^2 s/(4d).$$

Let $\eta := 1 - \sum_{i \geq j^\epsilon} x_i$. Since $\|x^\epsilon\|^2 \leq (1 - \eta)x_{j^\epsilon}$, we have $\|x^\epsilon\|^2 + x_{j^\epsilon}/3 \leq (4/3 - \eta)x_{j^\epsilon}$, so $\|x^\epsilon\|^2 + x_{j^\epsilon}/3 \geq \delta^2 s/(4d)$ implies that $x_{j^\epsilon} \geq \delta^2(s/d)/(16/3 - 4\eta)$. Thus, by Lemma 3 we have

$$\bar{D}(x) \leq -\eta \log(\epsilon^2(1 - \eta)s/4),$$

where we chose $\delta = \epsilon/4$. □

It now follows by Claim 2, Lemma 5 and Lemma 2 that if a pair u, v is underestimated by more than a factor of $1 - \epsilon$ with probability at least $1 - n^{-4}$, then

$$P_{uv}^l \leq \frac{1}{d}(\epsilon^2(1 - \epsilon/2)s/4)^{-(\epsilon/4)(l-1)}. \tag{8}$$

It remains to note that with probability at least $1 - n^{-2}$ one has $\hat{p}_j^{min} \leq (1 + \epsilon)P_{uv}^l$ for all u, v . Since by the previous estimate with probability at least $1 - 1/n$ for each u, v that satisfy (8) one has $\hat{p}_{t,j} \geq (1 - \epsilon)P_{uv}^l$, we have proved Theorem 1, which is the main result of this section.

We also show that the ϵ -factor in the exponent is not an artifact of our analysis (the proof is deferred to the full version of the paper):

Lemma 6. *For any $\epsilon < 1/2$ there exist graph $G = (V, E)$ with minimum degree d on which Algorithm 2 underestimates P_{uv}^l by more than $1 - \epsilon$ factor for pairs (u, v) such that $P_{uv}^l \geq \frac{1}{d_u}(s/2)^{-\epsilon(l-1)}$.*

References

1. Avrachenkov, K., Litvak, N., Nemirowsky, D., Osipova, N.: Monte carlo methods in pagerank computation: When one iteration is sufficient. *SIAM J. Numer. Anal.* 45 (2007)
2. Bahmani, B., Goel, A., Chowdhury, A.: Fast incremental and personalized pagerank. In: *PVLDB* (2010)
3. Baswana, S., Sen, S.: A simple linear time algorithm for computing $(2k - 1)$ -spanner of $o(n^{1+1/k})$ size for weighted graphs. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) *ICALP 2003*. LNCS, vol. 2719, Springer, Heidelberg (2003)
4. Berkhin, P.: A survey on pagerank computing. *Internet Mathematics* 2 (2005)
5. Clarkson, K., Woodruff, D.: Numerical linear algebra in the streaming model. In: *STOC* (2009)
6. Cohen, D.: Fast algorithms for constructing t -spanners and paths with stretch t . *SIAM Journal on Computing* 28, 210–236 (1999)
7. Dor, D., Halperin, U., Zwick, S.: All pairs almost shortest paths. *SIAM Journal on Computing* 29 (2000)
8. Drineas, P., Kannan, R.: Fast monte carlo algorithms for approximate matrix multiplication. In: *FOCS* (2001)
9. Drineas, P., Kannan, R., Mahoney, M.: Fast monte carlo algorithms for matrices i: Approximating matrix multiplication. *SIAM J. Computing* 36 (2006)

10. Elkin, M.L., Peleg, D.: $(1 + \epsilon, \beta)$ -spanner constructions for general graphs. In: STOC (2001)
11. Frieze, A., Kannan, R., Vempala, S.: Fast monte-carlo algorithms for finding low-rank approximations. JACM (2004)
12. Jeh, G., Widom, J.: Scaling personalized web search. In: WWW (2003)
13. Kamvar, S.D., Haveliwala, T.H., Manning, C.D., Golub, G.H.: Extrapolation methods for accelerating pagerank computations. In: WWW (2003)
14. Langville, A.N., Meyer, C.D.: Deeper inside pagerank. Internet Mathematics 1 (2004)
15. Magen, A., Zouzias, A.: Low rank matrix-valued chernoff bounds and approximate matrix multiplication. In: SODA (2011)
16. Nguyen, N.H., Do, T.T., Tran, T.D.: A fast and efficient algorithm for low-rank approximation of a matrix. In: STOC (2009)
17. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab (1999)
18. Patrascu, M., Roditty, L.: Distance oracles beyond the thorup-zwick bound. In: FOCS (2010)
19. Ruten, J., Kwiatkowska, M., Norman, G., Parker, D.: Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems. In: Panangaden, P., van Breugel, F. (eds.) Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems. CRM Monograph Series, vol. 23, American Mathematical Society, Providence (2004)
20. Sarlos, T.: Improved approximation algorithms for large matrices via random projections. In: FOCS (2006)
21. Sarlós, T., Benczúr, A.A., Csalogány, K., Fogaras, D., Rácz, B.: To randomize or not to randomize: space optimal summaries for hyperlink analysis. In: WWW (2006)
22. Sarma, A.D., Gollapudi, S., Panigrahy, R.: Estimating pagerank on graph streams. In: PODS (2008)
23. Thorup, M., Zwick, U.: Approximate distance oracles. In: STOC (2001)
24. Thorup, M., Zwick, U.: Spanners and emulators with sublinear distance errors. In: SODA (2006)
25. Woodruff, D.: Lower bounds for additive spanners, emulators and more. In: FOCS (2006)

Approximation Schemes for the Betweenness Problem in Tournaments and Related Ranking Problems

Marek Karpinski^{1,*} and Warren Schudy^{2,**}

¹ Dept. of Computer Science, University of Bonn
² IBM T.J. Watson Research Center

Abstract. We settle the approximability status of the *Minimum Betweenness* problem in tournaments by designing a *polynomial time approximation scheme (PTAS)*. No constant factor approximation was previously known. We also introduce a more general class of so-called *fragile* ranking problems and construct PTASs for them. The results depend on a new technique of dealing with fragile ranking constraints and could be of independent interest.

1 Introduction

We study the approximability of the Minimum Betweenness problem in tournaments (see [2]) that resisted so far efforts of designing polynomial time approximation algorithms with a constant approximation ratio. For the status of the general Betweenness problem, see e.g. [18,9,2,8,12].

In this paper we design the first *polynomial time approximation scheme* (PTAS) for that problem, and generalize it to much more general class of ranking CSP problems, called here *fragile* problems. To our knowledge it is the first non-trivial approximation algorithm for the Betweenness problem in tournaments.

In the Betweenness problem we are given a ground set of *vertices* and a set of *betweenness constraints* involving 3 vertices and a *designated* vertex among them. The cost of a ranking of the elements is the number of betweenness constraints with the designated vertex not between the other two vertices. The goal is to find a ranking *minimizing* this cost. We refer to the Betweenness problem in tournaments, that is in instances with a constraint for every triple of vertices, as the BETWEENNESSTOUR or *fully dense* Betweenness problem (see [2]). We consider also the k -ary extension k -FAST of the Feedback Arc Set in tournaments (FAST) problem (see [16,13]).

We extend the above problems by introducing a more general class of *fragile ranking k -CSP* problems inspired by the fragile (non-ranking) CSPs in [14]. A *constraint S* of a ranking k -CSP problem is called *fragile* if no two rankings of

* Parts of this work done while visiting Microsoft Research.

** Parts of this work done while a student at Brown University and while visiting University of Bonn.

the vertices S that both satisfy the constraint differ by the position of a single vertex. A *ranking k -CSP* problem is called *fragile* if all its constraints are fragile.

We now formulate our main results.

Theorem 1. *There exists a PTAS for the BETWEENNESSTOUR problem.*

The above answers an open problem of [2] on the approximation status of the Betweenness problem in tournaments.

We now formulate our first generalization.

Theorem 2. *There exist PTASs for all fragile ranking k -CSP problems in tournaments.*

Theorem 2 entails, among other things, existence of a PTAS for the k -ary extension of FAST. A PTAS for 2-FAST was given in [16].

Corollary 1. *There exists a PTAS for the k -FAST problem.*

We generalize BETWEENNESSTOUR to arities $k \geq 4$ by specifying for each constraint S a pair of vertices in S that must be placed at the ends of the ranking induced by the vertices in S . Such constraints do not satisfy our definition of fragile, but do satisfy a weaker notion that we call *weak fragility*. The definition of weakly fragile is identical to the definition for fragile except that only four particular single vertex moves are considered, namely swapping the first two vertices, swapping the last two, and moving the first or last vertex to the other end. We now formulate our most general theorem.

Theorem 3. *There exist PTASs for all weak-fragile ranking k -CSP problems in tournaments.*

Corollary 2. *There exists a PTAS for the k -BETWEENNESSTOUR problem.*

Additionally our algorithms are guaranteed not only to find a *low-cost* ranking but also a ranking that is *close to an optimal ranking* in the sense of Kendall-Tau distance. Karpinski and Schudy [15] recently utilized this extra feature to find an improved parameterized algorithm for BETWEENNESSTOUR with runtime $2^{O(\sqrt{OPT/n})} + n^{O(1)}$.

Theorem 4. *The PTASs of Theorem 3 additionally return a set of $2^{\tilde{O}(1/\epsilon)}$ rankings, one of which is guaranteed to be both cheap (cost at most $(1 + O(\epsilon))OPT$) and close to an optimal ranking (Kendall-Tau distance $O\left(\frac{\text{poly}(\frac{1}{\epsilon})OPT}{n^{k-2}}\right)$).*

All our PTASs are randomized but one can easily derandomize them by exhaustively considering every possible random choice.

Section 2 introduces notations and the problems we study. Section 3 introduces our algorithm and an intuitive sense of why it works. The remaining sections analyze the cost of the output of our algorithms. The runtime analysis and many of the proofs are omitted due to space limits. The full version of this paper is available from the arXiv: <http://arxiv.org/abs/0911.2214>.

2 Notation

First we state some core notation. Throughout this paper let V refer to the set of n objects (vertices) being ranked and $\epsilon > 0$ the desired approximation parameter. Our $O(\cdot)$ hides the arity k but not ϵ or n . Our $\tilde{O}(\cdot)$ additionally hides $(\log(1/\epsilon))^{O(1)}$. A *ranking* is a bijective mapping from a ground set $S \subseteq V$ to $\{1, 2, 3, \dots, |S|\}$. An *ordering* is an injection from S into \mathbb{R} . Clearly every ranking is also an ordering. We use π and σ (plus superscripts) to denote rankings and orderings respectively. Let π^* denote an optimal ranking and OPT its cost. We let $\binom{n}{k}$ (for example) denote the standard binomial coefficient and $\binom{V}{k}$ denote the set of subsets of set V of size k .

For any ordering σ let $Ranking(\sigma)$ denote the ranking naturally associated with σ . To help prevent ties we relabel the vertices so that $V = \{1, 2, 3, \dots, |V|\}$. We will often choose to place u in one of $O(1/\epsilon)$ positions $\mathcal{P}(u) = \{j\epsilon n + u/(n+1), 0 \leq j \leq 1/\epsilon\}$ (the $u/(n+1)$ term breaks ties). We say that an ordering is a *bucketed ordering* if $\sigma(u) \in \mathcal{P}(u)$ for all u . Let $Round(\pi)$ denote the bucketed ordering corresponding to π (rounding down), i.e. $Round(\pi)(u)$ equals $\pi(u)$ rounded down to the nearest multiple of ϵn , plus $u/(n+1)$.

Let $v \mapsto p$ denote the ordering over $\{v\}$ which maps vertex v to position $p \in \mathbb{R}$. For set Q of vertices and ordering σ with domain including Q let σ_Q denote the ordering over Q which maps $u \in Q$ to $\sigma(u)$, i.e. the restriction of σ to Q . For orderings σ^1 and σ^2 with disjoint domains let $\sigma^1 \upharpoonright \sigma^2$ denote the natural combined ordering over $Domain(\sigma^1) \cup Domain(\sigma^2)$. For example of our notations, $\sigma_Q \upharpoonright v \mapsto p$ denotes the ordering over $Q \cup \{v\}$ that maps v to p and $u \in Q$ to $\sigma(u)$.

A ranking k -CSP consists of a ground set V of *vertices*, an arity $k \geq 2$, and a *constraint system* c . Informally a constraint system c gives a 0/1 value (satisfied or not) for every ranking of every set $S \subseteq V$ of $|S| = k$ vertices. Formally a constraint system c is a function which maps rankings of vertices $S \subseteq V$ with $|S| = k$ to $\{0, 1\}$. For example if $k = 2$ and $V = \{u_1, u_2, u_3\}$ a constraint system c consists of the six values $c(u_1 \mapsto 1 \upharpoonright u_2 \mapsto 2)$, $c(u_1 \mapsto 2 \upharpoonright u_2 \mapsto 1)$, $c(u_1 \mapsto 1 \upharpoonright u_3 \mapsto 2)$, $c(u_1 \mapsto 2 \upharpoonright u_3 \mapsto 1)$, $c(u_2 \mapsto 1 \upharpoonright u_3 \mapsto 2)$, and $c(u_2 \mapsto 2 \upharpoonright u_3 \mapsto 1)$. A *weighted ranking CSP* has a *weighted constraint system* which maps rankings of vertices $S \subseteq V$, $|S| = k$ to non-negative reals \mathbb{R}^+ . (To simplify terminology we present our results for unweighted CSPs only. We define weighted CSPs only because our algorithm uses one.) We refer to a set of vertices $S \subseteq V$, $|S| = k$ in the context of constraint system c as a *constraint*. We say constraint S is *satisfied* in ordering σ of S if $c(Ranking(\sigma)) = 0$. For brevity we henceforth abuse notation and omit the “*Ranking*” and write simply $c(\sigma)$. The objective of a ranking CSP is to find an ordering σ (w.l.o.g. a ranking) minimizing the number of unsatisfied constraints, which we denote by $C^c(\sigma) = \sum_{S \in \binom{Domain(\sigma)}{k}} c(\sigma_S)$.

We will frequently leave the CSP in question implicit in our notations, for exemplifying saying that a constraint S is *satisfied* without specifying the constraint system. In such cases the CSP should be clear from context. We use k , c and V to denote the arity, constraint system and ground set of the CSP that we are trying to optimize. We also use the shorthand $C(\sigma) = C^c(\sigma)$.

Definition 1. A constraint S of constraint system c is fragile if no two orderings that satisfy it differ by the position of a single vertex unless the two orderings are associated with the same ranking. In other words constraint S is fragile if $c(\pi_S) + c(\pi'_S) \geq 1$ for all distinct rankings π and π' over S that differ by a single vertex move, i.e. $\pi' = \text{Ranking}(v \mapsto p \mid \pi_{S \setminus \{v\}})$ for some $v \in S$ and half-integer $p \in \{1/2, 3/2, 5/2, \dots, k + 1/2\}$.

An alternate definition is that a satisfied fragile constraint becomes unsatisfied whenever a single vertex is moved, which is why it is called “fragile.”

Definition 2. A constraint S of constraint system c is weakly fragile if $c(\pi_S) + c(\pi'_S) \geq 1$ for all rankings π and π' that differ by a swap of the first two vertices, a swap of the last two, or a cyclic shift of a single vertex. In other words $\pi' = \text{Ranking}(v \mapsto p \mid \pi_{S \setminus \{v\}})$ for some $v \in S$ and $p \in \mathbb{R}$ with $(\pi(v), p) \in \{(1, 2 + \frac{1}{2}), (1, k + \frac{1}{2}), (k, k - \frac{3}{2}), (k, \frac{1}{2})\}$.

Observe that weak fragility is equivalent to ordinary fragility for $k \leq 3$.

Our techniques handle ranking CSPs that are *fully dense* with weakly fragile constraints, i.e. every set S of k vertices corresponds to a weakly fragile constraint. Fully dense instances are also known as tournaments (by analogy with feedback arc set and tournament graphs).

Let $b^c(\sigma, v, p)$ denote the cost of the constraints in constraint system c involving vertex v in ordering $\sigma_{\text{Domain}(\sigma) \setminus \{v\}} \mid v \mapsto p$ formed by moving v to position p in ordering σ . Formally $b^c(\sigma, v, p) = \sum_{Q \dots} c(\sigma_Q \mid v \mapsto p)$, where the sum is over sets $Q \subseteq \text{Domain}(\sigma) \setminus \{v\}$ of size $k - 1$. Note that this definition is valid regardless of whether or not v is in $\text{Domain}(\sigma)$. The only requirement is that the range of σ excluding $\sigma(v)$ must not contain p . This ensures that the argument to $c(\cdot)$ is an ordering (injective). Analogously with the objective function C the superscript constraint system c in b^c defaults to the problem c that we are trying to solve when omitted.

We call a weighted ranking CSP instance with arity 2 a *feedback arc set (FAS) instance*. A FAS instance with vertex set V and constraint system w is equivalent to a weighted directed graph with arc weights $w_{uv} = w(u \rightarrow 2 \mid v \rightarrow 1)$ for $u, v \in V$. The objective function $C^w(\sigma)$ defined previously works out to finding an ordering of the vertices V minimizing the weight of the backwards arcs $C^w(\sigma) = \sum_{u, v: \sigma(u) > \sigma(v)} w_{uv}$. Similarly $b^w(\sigma, v, p) = \sum_{u \neq v} \begin{cases} w_{uv} & \text{if } \sigma(u) > p \\ w_{vu} & \text{if } \sigma(u) < p \end{cases}$. If a FAS instance with constraint system w satisfies $\alpha \leq w_{uv} + w_{vu} \leq \beta$ for all u, v and some $\alpha, \beta > 0$ we call it a (weighted) feedback arc set tournament (FAST) instance. We generalize to k -FAST as follows: a k -FAST constraint S is satisfied by one particular ranking of the vertices S and no others. Clearly k -FAST constraints are fragile.

We generalize BETWEENNESSTOUR to $k \geq 4$ as follows. Each constraint S designates two vertices $\{u, v\}$, which must be the first and last positions, i.e. if π is the ranking of the vertices in S then $c(\pi) = \mathbb{1}(\{\pi(u), \pi(v)\} \neq \{1, k\})$. It is easy to see that BETWEENNESSTOUR constraints are weakly fragile.

We use the following two results from the literature.

Theorem 5 ([16]). *Let w be a FAS instance satisfying $\alpha \leq w_{uv} + w_{vu} \leq \beta$ for $\alpha, \beta > 0$ and $\beta/\alpha = O(1)$. There is a PTAS for the problem of finding a ranking π minimizing $C^w(\pi)$ with runtime $n^{O(1)}2^{\tilde{O}(1/\epsilon^6)}$.*

Theorem 6 (e.g. [6,17]). *For any k -ary MIN-CSP and $\delta > 0$ there is an algorithm that produces a solution with cost at most δn^k more than optimal. Its runtime is $n^{O(1)}2^{O(1/\delta^2)}$.*

Theorem 6 entails the following corollary.

Corollary 3. *For any $\delta > 0$ and constraint system c there is an algorithm ADDAPPROX for the problem of finding a ranking π with $C(\pi) \leq C(\pi^*) + \delta n^k$, where π^* is an optimal ranking. Its runtime is $n^{O(1)}2^{\tilde{O}(1/\delta^2)}$.*

3 Intuition and Algorithm

We are in need for some new techniques different in nature from the techniques used for weighted FAST [16].

Our first idea is somehow analogous to the approximation of a differentiable function by a tangent line. Given a ranking π and any ranking CSP, the change in cost from switching to a similar ranking π' can be well approximated by the change in cost of a particular weighted feedback arc set problem (see proof of Lemma [15]). Furthermore if the ranking CSP is fragile and fully dense the corresponding feedback arc set instance is a (weighted) tournament (Lemma [9]). So if we somehow had access to a ranking similar to the optimum ranking π^* we could create this FAST instance and run the existing PTAS for weighted FAST [16] to get a good ranking.

We do not have access to π^* but we use techniques inspired by [14] to get close. We pick a random sample of vertices and guess their location in the optimal ranking to within (an additive) ϵn . We then create an ordering σ^1 greedily from the random sample. We show that this ordering is close to π^* , in that $|\pi^*(v) - \sigma^1(v)| = O(\epsilon n)$ for all but $O(\epsilon n)$ of the vertices (Lemma [5]).

We then do a second greedy step (relative to σ^1), creating σ^2 . We then identify a set U of *unambiguous* vertices (defined in Algorithm [1]) for which we know $|\pi^*(v) - \sigma^2(v)| = O(\epsilon n)$ (Lemma [8]). We temporarily set aside the $O(OPT/(\epsilon n^{k-1}))$ (Lemma [7]) remaining vertices. These two greedy steps are similar in spirit to previous work on ordinary (non-ranking) everywhere-dense fragile CSPs [14] but substantially more involved.

We then use σ^2 to create a (weighted) FAST instance w that locally represents the CSP. It would not be so difficult to show that w is a close enough representation for an additive approximation, but we want a multiplicative $1 + \epsilon$ approximation. Showing this requires overcoming two obstacles that are our main technical contribution.

Firstly the error in σ^2 causes the weights of w to have significant error (Lemma [11]) even in the extreme case of $OPT = 0$. At first glance even an exact solution to this FAST problem would seem insufficient, for how can solving a problem

similar to the desired one lead to a precisely correct solution? Fortunately FAST is somewhat special. It is easy to see that a zero-cost instance of FAST cannot be modified to change its optimal ranking without modifying an arc weight by at least $1/2$. We extend this idea to cases where OPT is small but non-zero (Lemma 15).

The second obstacle is that the incorrect weights in FAST instance w may increase the optimum cost of w far above OPT , leaving the PTAS for FAST free to return a poor ranking. To remedy this we create a new FAST instance \bar{w} by canceling weight on opposing arcs, i.e. reducing w_{uv} and w_{vu} by the same amount. The resulting simplified instance \bar{w} clearly has the same optimum ranking as w but a smaller optimum value. The PTAS for FAST requires that the ratio of the maximum and the minimum of $w_{uv} + w_{vu}$ must be bounded above by a constant so we limit the amount of cancellation to ensure this (Lemma 9). It turns out that this cancellation trick is sufficient to ensure that the PTAS for FAST does not introduce too much error (Lemma 12).

Finally we greedily insert the relatively few ambiguous vertices into the ranking output by the PTAS for FAST 16.

For any ordering σ with domain U we will shortly define a weighted feedback arc set instance w^σ which approximates the overall problem c in the neighborhood of ordering σ . In particular changes in the objective $C = C^c$ are approximately equal to changes in C^{w^σ} . Before giving the definition of w^σ we describe how it was chosen. For simplicity of illustration let us suppose that $|V| = k$ and hence we have only one constraint; the general case will follow by making w_{uv}^σ a sum over contributions by the various constraints $S \supseteq \{u, v\}$. We are looking for good approximations for the costs of nearby ordering, so let us consider the nearest possible ordering: let σ' be identical to σ except that two adjacent vertices, call them u and v , are swapped: $\sigma(u) < \sigma(v)$ and $\sigma'(u) > \sigma'(v)$. Clearly $C^{w^\sigma}(\sigma') - C^{w^\sigma}(\sigma) = w_{uv}^\sigma - w_{vu}^\sigma$. It is therefore natural to set $w_{vu}^\sigma = c(\sigma)$ and $w_{uv}^\sigma = c(\sigma')$, hence $C(\sigma') - C(\sigma) = C^{w^\sigma}(\sigma) - C^{w^\sigma}(\sigma)$ as desired.

So what about w_{uv}^σ for u and v that are not adjacent in σ ? It turns out that we can pick practically anything for the other w_{uv}^σ as long as we keep $C^{w^\sigma}(\sigma)$ small and $w_{uv}^\sigma + w_{vu}^\sigma$ relatively uniform. We extend the above definition to non-adjacent u, v with $\sigma(u) < \sigma(v)$ as follows: set $w_{vu}^\sigma = c(\sigma)$ and $w_{uv}^\sigma = c(\sigma')$, where σ' is identical to σ except that v is placed immediately before u . (Another natural option would be to set $w_{vu} = 0$ and $w_{uv} = 1$ for non-adjacent u, v with $\sigma(u) < \sigma(v)$.)

With this motivation in hand we now give the formal definition of w^σ . For any ordering σ with domain U let w_{uv}^σ equal the number of the constraints $\{u, v\} \subseteq S \subseteq U$ with $c(\sigma') = 1$ where (1) $\sigma' = (\sigma_{S \setminus \{v\}} \mid v \rightarrow p)$, (2) $p = \sigma(u) - \delta$ if $\sigma(v) > \sigma(u)$ and $p = \sigma(v)$ otherwise, and (3) $\delta > 0$ is sufficiently small to put p adjacent to $\sigma(u)$. In other words if v is after u in σ it is placed immediately before u in σ' . Observe that $0 \leq w_{uv}^\sigma \leq \binom{|U|-2}{k-2}$.

The following Lemma follows easily from the definitions.

Lemma 1. *For any ordering σ we have (1) $C^{w^\sigma}(\sigma) = \binom{k}{2}C(\sigma)$ and (2) $b^{w^\sigma}(\sigma, v, \sigma(v)) = (k - 1) \cdot b(\sigma, v, \sigma(v))$ for all v .*

Algorithm 1. A $(1 + O(\epsilon))$ -approximation for weak fragile rank k -CSPs in tournaments.

Input: Vertex set V , $|V| = n$, arity k , system c of fully dense arity k constraints, and approximation parameter $\epsilon > 0$.

- 1: Run `ADDAPPROX`($\epsilon^5 n^k$) and return the result if its cost is at least $\epsilon^4 n^k$
 - 2: Pick sets T_1, \dots, T_t uniformly at random with replacement from $\binom{V}{k-1}$, where $t = \frac{14 \ln(40/\epsilon)}{\binom{k}{2}^\epsilon}$. Guess (by exhaustion) bucketed ordering σ^0 , which is the restriction of $\text{Round}(\pi^*)$ to the sampled vertices $\bigcup_i T_i$, where π^* is an optimal ranking.
 - 3: Compute bucketed ordering σ^1 greedily with respect to the random samples and σ^0 , i.e.:

$$\sigma^1(u) = \operatorname{argmin}_{p \in \mathcal{P}(u)} \hat{b}(u, p) \text{ where } \hat{b}(u, p) = \binom{n}{t} \sum_{i: u \notin T_i} c(\sigma^0_{T_i} \mid v \mapsto p).$$
 - 4: For each vertex v : If $b(\sigma^1, v, p) \leq 13k^4 3^{k-1} \epsilon \binom{n-1}{k-1}$ for some $p \in \mathcal{P}(v)$ then call v *unambiguous* and set $\sigma^2(v)$ to the corresponding p (pick any if multiple p satisfy). Let U denote the set of unambiguous vertices, which is the domain of bucketed ordering σ^2 .
 - 5: Compute feedback arc set instance \bar{w}^{σ^2} over unambiguous vertices U (see text). Solve it using the `FAST PTAS` [16]. Do single vertex moves until local optimality (with respect to the `FAST` objective function), yielding ranking π^3 of U .
 - 6: Create ordering σ^4 over V defined by $\sigma^4(u) = \begin{cases} \pi^3(u) & \text{if } u \in U \\ \operatorname{argmin}_{p=v/(n+1)+j, 0 \leq j \leq n} b(\pi^3, u, p) & \text{otherwise} \end{cases}$. In other words insert each vertex $v \in V \setminus U$ into $\pi^3(v)$ greedily.
 - 7: Return $\pi^4 = \text{Ranking}(\sigma^4)$.
-

Proof. Observe that all w_{uv}^σ that contribute to $C^{w^\sigma}(\sigma)$ or $b^{w^\sigma}(\sigma, v, \sigma(v))$ satisfy $\sigma(u) > \sigma(v)$ and hence the σ' in the definition of w_{uv}^σ is equal to σ . It follows that each w_{uv}^σ that contributes to $C^{w^\sigma}(\sigma)$ or $b^{w^\sigma}(\sigma, v, \sigma(v))$ is equal to the number of constraints containing u and v that are unsatisfied in σ . The $\binom{k}{2}$ and $k-1$ factors appear because each constraint S contributes to w_{uv}^σ for a variety of $u, v \in S$.

The weighted feedback arc set instance w^σ is insufficient for our purposes because its objective value can be large even when the optimal cost of c is small. To remedy this we cancel the weight on opposing arcs (within limits), yielding another weighted feedback arc set instance \bar{w}^σ . In particular for any ordering σ we define $\bar{w}_{uv}^\sigma = w_{uv}^\sigma - \min(\frac{1}{10 \cdot 3^{k-1}} \binom{|U|-2}{k-2}, w_{uv}^\sigma, w_{vu}^\sigma)$, where U is the domain of σ . Observe that $C^{w^\sigma}(\pi') - C^{\bar{w}^\sigma}(\pi')$ is a non-negative constant independent of ranking π' . Therefore the feedback arc set problems induced by w^σ and \bar{w}^σ have the same optimal rankings but an approximation factor of $(1 + \epsilon)$ is a stronger guarantee for \bar{w}^σ than for w^σ .

For any orderings σ and σ' with domain U , we say that $\{u, v\} \subseteq U$ is a σ/σ' -inversion if $\sigma(u) - \sigma(v)$ and $\sigma'(u) - \sigma'(v)$ have different signs. Let $d(\sigma, \sigma')$ denote the number of σ/σ' -inversions (a.k.a. Kendall Tau distance). We say that v does a *left to right* (σ, p, σ', p') -crossing if $\sigma(v) < p$ and $\sigma'(v) > p'$. We say that v does a *right to left* $(\sigma, p/\sigma', p')$ -crossing if $\sigma(v) > p$ and $\sigma'(v) < p'$. We say that

v does a (σ, p, σ', p') -crossing if v does a crossing of either sort. We say that u σ/σ' -crosses $p \in \mathbb{R}$ if it does a (σ, p, σ', p) -crossing.

With these notations in hand we present our Algorithm \square for approximating a weak fragile rank k -CSP. The non-deterministic “guess (by exhaustive sampling)” on line 2 of our algorithm should be implemented in the traditional manner: place the remainder of the algorithm in a loop over possible orderings of the sample, with the overall return value equal to the best of the π^4 rankings found. Our algorithm can be derandomized by choosing T_1, \dots, T_t non-deterministically rather than randomly; details omitted.

If $OPT \geq \epsilon^4 n^k$ then the first line of the algorithm is sufficient for a PTAS so for the remainder of the analysis we assume that $OPT \leq \epsilon^4 n^k$.

4 Analysis of σ^1

Let $\sigma^\square = Round(\pi^*)$. Call vertex v *costly* if $b(\sigma^\square, v, \sigma^\square(v)) \geq 2 \binom{k}{2} \epsilon \binom{n-1}{k-1}$ and *non-costly* otherwise.

Lemma 2. *The number of costly vertices is at most $\frac{k \cdot OPT}{\epsilon \binom{k}{2} \binom{n-1}{k-1}}$.*

The proof of Lemma \square is omitted. The outline of the proof is $kC(\pi^*) = \sum_v b(\pi^*, v, \pi^*(v)) \approx \sum_v b(\sigma^\square, v, \sigma^\square(v)) \geq (\text{number costly}) 2 \binom{k}{2} \epsilon \binom{n-1}{k-1}$.

Lemma 3. *Let σ be an ordering of V , $|V| = n$, $v \in V$ be a vertex and $p, p' \in \mathbb{R}$. Let B be the set of vertices (excluding v) between p and p' in σ . Then $b(\sigma, v, p) + b(\sigma, v, p') \geq \frac{|B|}{(n-1)3^{k-1}} \binom{n-1}{k-1}$.*

Proof. By definition

$$b(\sigma, v, p) + b(\sigma, v, p') = \sum_{Q: \dots} [c(\sigma_Q \mid v \mapsto p) + c(\sigma_Q \mid v \mapsto p')] \tag{1}$$

where the sum is over sets $Q \subseteq U \setminus \{v\}$ of $k - 1$ vertices.

We consider the illustrative special case of betweenness tournament (or more generally fragile problems with arity $k = 3$) here and defer the general case to the full version. Betweenness constraints have a special property: the quantity in brackets in (\square) is at least 1 for every Q that has at least one vertex between p and p' in π . There are at least $|B|(n - 2)/2$ such sets, which can easily be lower-bounded by the desired $\frac{|B|}{(n-1)3^{3-1}} \binom{n-1}{3-1}$.

For vertex v we say that a position $p \in \mathcal{P}(v)$ is *v -out of place* if there are at least $6 \binom{k}{2} 3^{k-1} \epsilon n$ vertices between p and $\sigma^\square(v)$ in σ^\square . We say vertex v is *out of place* if $\sigma^1(v)$ is v -out of place.

Lemma 4. *The number of non-costly out of place vertices is at most $\epsilon n/2$ with probability at least $9/10$.*

The omitted proof uses Lemma 3 and the definitions of out-of-place and costly to show that $b(\sigma^\square, v, \sigma^\square(v))$ is much smaller than $b(\sigma^\square, v, p)$ for any v -out of place p , and then Chernoff and union bounds to show that $\hat{b}(v, p)$ is sufficiently concentrated about its mean $b(\sigma^\square, v, p)$ so that the minimum $\hat{b}(v, p)$ must occur for a p that is not v -out of place.

Lemma 5. *With probability at least 9/10 the following are simultaneously true:*

1. *The number of out of place vertices is at most ϵn .*
2. *The number of vertices v with $|\sigma^1(v) - \sigma^\square(v)| > 3k^2 3^{k-1} \epsilon n$ is at most ϵn*
3. *$d(\sigma^1, \sigma^\square) \leq 6k^2 3^{k-1} \epsilon n^2$*

Proof. By Lemma 2 and the fact $OPT \leq \epsilon^4 n^k$ we have at most $\frac{k \cdot OPT}{\binom{k}{2} \epsilon^{\binom{n-1}{k-1}}} \leq \epsilon n/2$ costly vertices for n sufficiently large. Therefore Lemma 4 implies the first part of the Lemma. We finish the proof by showing that whenever the first part holds the second and third parts hold as well.

Observe that there are exactly ϵn vertices in σ^\square between any two consecutive positions in $\mathcal{P}(v)$. It follows that any vertex with $|\sigma^1(v) - \sigma^\square(v)| > 3k^2 3^{k-1} \epsilon n \geq (6\binom{k}{2} 3^{k-1} + 1)\epsilon n$ must necessarily be v -out of place, completing the proof of the second part of the Lemma.

For the final part observe that if u and v are a σ^1/σ^\square -inversion and not among the ϵn out of place vertices then, by definition of out-of-place, there can be at most $2 \cdot 6\binom{k}{2} 3^{k-1} \epsilon n$ vertices between $\sigma^\square(v)$ and $\sigma^\square(u)$ in σ^\square . For each u there are therefore only $24\binom{k}{2} 3^{k-1} \epsilon n$ possibilities for v . Therefore $d(\sigma^1, \sigma^\square) \leq \epsilon n^2 + 24\binom{k}{2} 3^{k-1} \epsilon n \cdot n/2 \leq 6\epsilon k^2 3^{k-1} n^2$.

The remainder of our analysis assumes that the event of Lemma 5 holds without stating so explicitly.

5 Analysis of σ^2

The following key Lemma shows the sensitivity of $b(\sigma, v, p)$ to its first and third arguments. The proof is omitted.

Lemma 6. *For any constraint system c with arity $k \geq 2$, orderings σ and σ' over vertex set $T \subseteq V$, vertex $v \in V$ and $p, p' \in \mathbb{R}$ we have*

1. $|b^c(\sigma, v, p) - b^c(\sigma', v, p')| \leq \binom{n-2}{k-2} (\text{number of crossings}) + \binom{n-3}{k-3} d(\sigma, \sigma')$
2. $|b^c(\sigma, v, p) - b^c(\sigma', v, p')| \leq \binom{n-2}{k-2} (|\text{net flow}| + k\sqrt{d(\sigma, \sigma')})$

where $\binom{n-3}{k-3} = 0$ if $k = 2$, (net flow) is $|\{v \in T : \sigma'(v) > p'\}| - |\{v \in T : \sigma(v) > p\}|$, and (number of crossings) is the number of $v \in T$ that do a (σ, p, σ', p') -crossing.

Observe that the quantity *net flow* in Lemma 6 is zero whenever $p = p'$ and σ and σ' are both *rankings*. Therefore we have the following useful corollary.

Corollary 4. *Let π and π' be rankings over vertex set U and w a FAST instance over U . Then $|b^w(\pi, v, p) - b^w(\pi', v, p)| \leq 2(\max_{r,s} w_{rs})\sqrt{d(\pi, \pi')}$ for all v and $p \in \mathbb{R} \setminus \mathbb{Z}$.*

We let U denote the set of unambiguous vertices as defined in Algorithm 11.

Lemma 7. *We have $|V \setminus U| \leq \frac{k \cdot OPT}{\epsilon \binom{k}{2} \binom{n-1}{k-1}} = O(\frac{n}{\epsilon} \cdot \frac{OPT}{n^k})$.*

Proof. Observe that the number of vertices that σ^\square/σ^1 -cross a particular p is at most $2 \cdot 6k^2 3^{k-1} \epsilon n$ by Lemma 5 (first part). Therefore we apply Lemmas 5 and 6, yielding

$$\begin{aligned} & |b(\sigma^\square, v, p) - b(\sigma^1, v, p)| \\ & \leq \binom{n-2}{k-2} 12k^2 3^{k-1} \epsilon n + \binom{n-3}{k-3} 6k^2 3^{k-1} \epsilon n^2 \leq 12\epsilon k^4 3^{k-1} \binom{n-1}{k-1} \end{aligned} \quad (2)$$

for all v and p .

Fix a non-costly v . By definition of costly $b(\sigma^\square, v, \sigma^\square(v)) \leq 2\binom{k}{2}\epsilon\binom{n-1}{k-1} \leq k^4 3^{k-1} \epsilon \binom{n-1}{k-1}$, hence $b(\sigma^1, v, \sigma^\square(v)) \leq 13k^4 3^{k-1} \epsilon \binom{n-1}{k-1}$, so $v \in U$.

Finally recall Lemma 2.

We define π^\otimes to be the ranking induced by the restriction of π^* to U , i.e. $\pi^\otimes = \text{Ranking}(\pi^*_U)$.

Lemma 8. *All vertices in the unambiguous set U satisfy $|\sigma^2(v) - \pi^\otimes(v)| = O(\epsilon n)$.*

Proof. The triangle inequality $|\sigma^2(v) - \pi^\otimes| \leq |\sigma^2(v) - \pi^*(v)| + |\pi^*(v) - \pi^\otimes|$ allows us to instead bound the two terms $|\sigma^2(v) - \pi^*(v)|$ and $|\pi^*(v) - \pi^\otimes|$ separately by $O(\epsilon n)$. We bound $|\sigma^2(v) - \pi^*(v)|$ first.

Since π^* is a ranking the number of vertices $|B|$ between $\pi^*(v)$ and $\sigma^2(v)$ in π^* is at least $|\pi^*(v) - \sigma^2(v)| - 1$. Therefore we have

$$\begin{aligned} \frac{|\pi^*(v) - \sigma^2(v)| - 1}{(n-1)3^{k-1}} \binom{n-1}{k-1} & \leq b(\pi^*, v, \sigma^2(v)) + b(\pi^*, v, \pi^*(v)) \quad (\text{Lemma 3}) \\ & \leq 2b(\pi^*, v, \sigma^2(v)) \quad (\text{Optimality of } \pi^*). \end{aligned} \quad (3)$$

We next apply the first part of Lemma 6 to π^* and σ^\square , bounding the number of crossings and $d(\pi^*, \sigma^\square)$ using the definition $\sigma^\square = \text{Round}(\pi^*)$, yielding

$$b(\pi^*, v, \sigma^2(v)) \leq b(\sigma^\square, v, \sigma^2(v)) + O(\epsilon n^{k-1}). \quad (4)$$

Next recalling (2) from the proof of Lemma 7 we have

$$b(\sigma^\square, v, \sigma^2(v)) \leq b(\sigma^1, v, \sigma^2(v)) + O(\epsilon n^{k-1}). \quad (5)$$

Combining (3), (4) and (5) we conclude that $|\pi^*(v) - \sigma^2(v)| = O(\epsilon n)$.

Now we prove $|\pi^*(v) - \pi^\otimes| = O(\epsilon n)$. Lemma 7, the definition of π^\otimes , and the assumption that $OPT \leq \epsilon^4 n^k$ imply that $|\pi^\otimes(v) - \pi^*(v)| \leq \frac{k \cdot OPT}{\epsilon \binom{k}{2} \binom{n-1}{k-1}} = O(\epsilon n)$.

6 Analysis of π^3

Note that all orderings and costs in this section are over the set of unambiguous vertices U as defined in Algorithm [1](#), not V . We note that Lemma [7](#) and the assumption that $OPT \leq \epsilon^4 n^k$ is small imply that $|U| = n - O(\epsilon^3 n)$.

We omit the proofs of the next six lemmas due to lack of space.

Lemma 9. $\frac{1}{3^{k-1}}(1-2/10)^{\binom{|U|-2}{k-2}} \leq \bar{w}_{uv}^{\sigma^2} + \bar{w}_{vu}^{\sigma^2} \leq 2 \binom{|U|-2}{k-2}$, i.e. \bar{w}^{σ^2} is a weighted FAST instance.

Lemma 10. Assume ranking π and ordering σ satisfy $|\pi(u) - \sigma(u)| = O(\epsilon n)$ for all u . For any u, v , let N_{uv} denote the number of $S \supset \{u, v\}$ such that not all pairs $\{s, t\} \neq \{u, v\}$ are in the same order in σ and π . We have $N_{uv} = O(\epsilon n^{k-2})$.

Lemma 11. The following inequalities hold:

1. $w_{uv}^{\sigma^2} \leq w_{uv}^{\pi^{\otimes}} + O(\epsilon n^{k-2})$
2. $\bar{w}_{uv}^{\sigma^2} \leq (1 + O(\epsilon))w_{uv}^{\pi^{\otimes}}$

Lemma 12.

1. $C^{\bar{w}^{\sigma^2}}(\pi^{\otimes}) \leq (1 + O(\epsilon))\binom{k}{2}C(\pi^{\otimes})$
2. $C^{\bar{w}^{\sigma^2}}(\pi^3) \leq (1 + O(\epsilon))\binom{k}{2}C(\pi^{\otimes})$
3. $C^{\bar{w}^{\sigma^2}}(\pi^3) - C^{\bar{w}^{\sigma^2}}(\pi^{\otimes}) = O(\epsilon C(\pi^{\otimes}))$

Lemma 13. $d(\pi^3, \pi^{\otimes}) = O(C(\pi^{\otimes})/n^{k-2})$

Lemma 14. We have $|\pi^3(v) - \pi^{\otimes}(v)| = O(\epsilon n)$ for all $v \in U$.

Lemma 15. $C(\pi^3) \leq (1 + O(\epsilon))C(\pi^{\otimes})$.

Proof. First we claim that

$$|(C(\pi^3) - C(\pi^{\otimes})) - (C^{w^{\sigma^2}}(\pi^3) - C^{w^{\sigma^2}}(\pi^{\otimes}))| \leq E_1, \tag{6}$$

where E_1 is the number of constraints that contain one pair of vertices u, v in different order in π^3 and π^{\otimes} and another pair $\{s, t\} \neq \{u, v\}$ with relative order in π^3, π^{\otimes} and σ^2 not all equal. Indeed constraints ordered identically in π^3 and π^{\otimes} contribute zero to both sides of [\(6\)](#), regardless of σ^2 . Consider some constraint S containing a π^3/π^{\otimes} -inversion $\{u, v\} \subset S$. If the restrictions of the three orderings to S are identical except possibly for swapping u, v then S contributes equally to both sides of [\(6\)](#), proving the claim.

To bound E_1 observe that the number of inversions u, v is $d(\pi^3, \pi^{\otimes}) \equiv D$. For any u, v Lemmas [14](#), [8](#) and [10](#) allow us to show at most $O(\epsilon n^{k-2})$ constraints containing $\{u, v\}$ contribute to E_1 , so $E_1 = O(D\epsilon n^{k-2}) = O(\epsilon C(\pi^{\otimes}))$ (Lemma [13](#)).

Finally bound $C^{w^{\sigma^2}}(\pi^3) - C^{w^{\sigma^2}}(\pi^{\otimes}) = C^{\bar{w}^{\sigma^2}}(\pi^3) - C^{\bar{w}^{\sigma^2}}(\pi^{\otimes}) \leq O(\epsilon C(\pi^{\otimes}))$, where the equality follows from the definition of w and the inequality is the third part of Lemma [12](#).

Extending Lemma [15](#) to a bound on the cost of π^4 (and hence a proof of our main theorems) is relatively straightforward. We omit the proof for lack of space.

Acknowledgements. We would like to thank Venkat Guruswami, Claire Mathieu, Prasad Raghavendra and Alex Samorodnitsky for interesting remarks and discussions.

References

1. Ailon, N.: Aggregation of Partial Rankings, p -ratings and top- m Lists. In: 18th SODA, pp. 415–424 (2007)
2. Ailon, N., Alon, N.: Hardness of Fully Dense Problems. *Inf. Comput.* 205, 1117–1129 (2007)
3. Ailon, N., Charikar, M., Newman, A.: Aggregating Inconsistent Information: Ranking and Clustering. *J. ACM* 55 (2008)
4. Alon, N., Fernandez de la Vega, W., Kannan, R., Karpinski, M.: Random Sampling and Approximation of MAX-CSP Problems. In: 34th ACM STOC, pp. 232–239 (2002); Journal version in *JCSS* 67, 212–243 (2003)
5. Alon, N., Lokshantov, D., Saurabh, S.: Fast FAST. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikolettseas, S., Thomas, W. (eds.) *ICALP 2009*. LNCS, vol. 5555, pp. 49–58. Springer, Heidelberg (2009)
6. Arora, S., Karger, D., Karpinski, M.: Polynomial Time Approximation Schemes for Dense Instances of NP-Hard Problems. In: 27th ACM STOC, pp. 284–293 (1995); Journal version in *J. Comput. System Sciences* 58, 193–210 (1999)
7. Bazgan, C., Fernandez de la Vega, W., Karpinski, M.: Polynomial Time Approximation Schemes for Dense Instances of the Minimum Constraint Satisfaction Problem. *Random Structures and Algorithms* 23, 73–91 (2003)
8. Charikar, M., Guruswami, V., Manokaran, R.: Every Permutation CSP of Arity 3 is Approximation Resistant. In: 24th IEEE CCC (2009)
9. Chor, B., Sudan, M.: A Geometric Approach to Betweenness. *SIAM J. Discrete Math.* 11, 511–523 (1998)
10. Fernandez de la Vega, W., Kannan, R., Karpinski, M.: Approximation of Global MAX-CSP Problems. Technical Report TR06-124, *ECCC* (2006)
11. Frieze, A., Kannan, R.: Quick Approximation to Matrices and Applications. *Combinatorica* 19, 175–220 (1999)
12. Guruswami, V., Hastad, J., Manokaran, R., Raghavendra, P., Charikar, M.: Beating the random ordering is hard: Every ordering csp is approximation resistant. In: *ECCC TR-11-027* (2011)
13. Gutin, G., Kim, E.J., Mnich, M., Yeo, A.: Betweenness parameterized above tight lower bound. *Journal of Computer and System Sciences* 76(8), 872–878 (2010)
14. Karpinski, M., Schudy, W.: Linear Time Approximation Schemes for the Gale-Berlekamp Game and Related Minimization Problems. In: 41st ACM STOC, pp. 313–322 (2009)
15. Karpinski, M., Schudy, W.: Faster algorithms for feedback arc set tournament, kemeny rank aggregation and betweenness tournament (LNCS 6506/2010). In: Cheong, O., Chwa, K.-Y., Park, K. (eds.) *ISAAC 2010*. LNCS, vol. 6506, pp. 3–14. Springer, Heidelberg (2010)
16. Mathieu, C., Schudy, W.: How to Rank with Few Errors. In: 39th ACM STOC, pp. 95–103 (2007); In Submission
http://www.cs.brown.edu/~simon/papers/fast_journal.pdf (2009)
17. Mathieu, C., Schudy, W.: Yet Another Algorithm for Dense Max Cut: Go Greedy. In: *Proc. 19th ACM-SIAM SODA*, pp. 176–182 (2008)
18. Opatrný, J.: Total Ordering Problem. *SIAM J. Comput.* 8, 111–114 (1979)

Network-Design with Degree Constraints

Rohit Khandekar¹, Guy Kortsarz^{2,*}, and Zeev Nutov³

¹ IBM T.J. Watson Research Center

rohitk@us.ibm.com

² Rutgers University, Camden

guyk@camden.rutgers.edu

³ The Open University of Israel

nutov@openu.ac.il

Abstract. We study several network design problems with degree constraints. For the degree-constrained 2-connected subgraph problem we obtain a factor 6 violation for the degrees with 4 approximation for the cost. This improves upon the logarithmic degree violation and no cost guarantee obtained by Feder, Motwani, and Zhu (2006). Then we consider the problem of finding an arborescence with at least k terminals and with minimum maximum outdegree. We show that the natural LP-relaxation has a gap of $\Omega(\sqrt{k})$ or $\Omega(n^{1/4})$ with respect to the multiplicative degree bound violation. We overcome this hurdle by a combinatorial $O(\sqrt{(k \log k)/\Delta^*})$ -approximation algorithm, where Δ^* denotes the maximum degree in the optimum solution. We also give an $\Omega(\log n)$ lower bound on approximating this problem. Then we consider the undirected version of this problem, however, with an extra diameter constraint, and give an $\Omega(\log n)$ lower bound on the approximability of this version. Finally, we consider a closely related prize-collecting degree-constrained Steiner Network problem. We obtain several results in this direction by reducing the prize-collecting variant to the regular one.

1 Introduction

1.1 Problems Considered

A graph H is k -(node-)connected if it contains k internally disjoint paths between every pair of its nodes. In the k -Connected Subgraph problem we are given a graph $G = (V, E)$ with edge-costs and an integer k . The goal is to find a minimum cost k -connected spanning subgraph H of G . In the Degree-Constrained k -Connected Subgraph problem, we are also given degree bounds $\{b(v) : v \in V\}$. The goal is to find a minimum-cost k -connected spanning subgraph H of G such that in H , the degree of every node v is at most $b(v)$. We consider the case $k = 2$.

* Partially supported by NSF Award Grant number 0829959.

Degree-Constrained 2-Connected Subgraph

Instance: An undirected graph $G = (V, E)$ with edge-costs $c_e \geq 0$, and degree bounds $\{b(v) : v \in V\}$.

Objective: Find a minimum cost 2-connected spanning subgraph H of G that satisfies the *degree constraints* $\deg_H(v) \leq b(v)$ for all $v \in V$.

Given a directed graph G , a set S of terminals, and an integer $k \leq |S|$, a k -*arborescence* is an arborescence in G that contains k terminals; in the case of undirected graphs we have a k -*tree*. For a directed/undirected graph or edge-set H let $\Delta(H)$ denote the maximum outdegree/degree of a node in H .

Minimum Degree k -Arborescence

Instance: A directed graph $G = (V, E)$, a root $s \in V$, a subset $S \subseteq V \setminus \{s\}$ of terminals, and an integer $k \leq |S|$.

Objective: Find in G a k -arborescence T rooted at s that minimizes $\Delta(T)$.

For the next problem we only show a lower bound. Hence we show it for the *less* general case of undirected graphs.

Degree and Diameter Bounded k -Tree

Instance: An undirected graph $G = (V, E)$, a subset $S \subseteq V$ of terminals, an integer $k \leq |S|$, and a diameter bound D .

Objective: Find a k -tree T with diameter bounded by D that minimizes $\Delta(T)$.

Let $\lambda_H(u, v)$ denote the the maximum number of edge-disjoint uv -paths in H . In the **Steiner Network** problem we are given a graph $G = (V, E)$ with edge-costs $c_e \geq 0$, a collection $\mathcal{P} = \{\{u_1, v_1\}, \dots, \{u_k, v_k\}\}$ of node pairs, and connectivity requirements $\mathcal{R} = \{r_1, \dots, r_k\}$. The goal is to find a minimum-cost subgraph H of G that satisfies the connectivity requirements $\lambda_H(u_i, v_i) \geq r_i$ for all i .

We consider a combination of the following two versions of the **Steiner Network** problem. In **Degree-Constrained Steiner Network** we are given degree bounds $\{b(v) : v \in V\}$. The goal is to find a minimum-cost subgraph H of G that satisfies the connectivity requirements and the degree constraints $\deg_H(v) \leq b(v)$ for all $v \in V$. In **Prize-Collecting Steiner Network** we are given a submodular monotone non-decreasing penalty function $\pi : 2^{\{1, \dots, k\}} \rightarrow \mathbb{R}_+$ (π is given by an evaluation oracle). The goal is to find a subgraph H of G that minimizes the *value* $\text{val}(H) = c(H) + \pi(\text{unsat}(H))$ of H , where $\text{unsat}(H) = \{i \mid \lambda_H^S(u_i, v_i) < r_i\}$ is the set of requirements *not* (completely) satisfied by H . Formally, the problem we consider is as follows.

Prize-Collecting Degree-Constrained Steiner Network

Instance: A graph $G = (V, E)$ with edge-costs $c_e \geq 0$, a collection $\mathcal{P} = \{\{u_1, v_1\}, \dots, \{u_k, v_k\}\}$ of node pairs, connectivity requirements $\mathcal{R} = \{r_1, \dots, r_k\}$, a submodular monotone non-decreasing penalty function $\pi : 2^{\{1, \dots, k\}} \rightarrow \mathbb{R}_+$ given by an evaluation oracle, and degree bounds $\{b(v) : v \in V\}$.

Objective: Find a subgraph H of G that satisfies the *degree constraints* $\deg_H(v) \leq b(v)$ for all $v \in V$, and minimizes the *value*

$$\text{val}(H) = c(H) + \pi(\text{unsat}(H))$$

of H , where $\text{unsat}(H) = \{i \mid \lambda_H^S(u_i, v_i) < r_i\}$ is the set of requirements *not* satisfied by H .

The Steiner Tree problem is a particular case of Steiner Network when we seek a minimum-cost subtree T of G that contains a specified subset S of terminals. In the degree constrained version of Steiner Tree, we are also given degree bounds on nodes in S and need to satisfy the degree constraints. We consider the case of 0,1 constraints, namely, we require that certain nodes in S should be leaves of T , and do not allow to relax this condition, as was done in previous papers [12,13]. Namely, the degree bounds here are of the “hard capacity” type, and cannot be violated. Formally, our problem can be casted as follows.

Leaf-Constrained Steiner Tree

Instance: A graph $G = (V, E)$ with edge-costs $c_e \geq 0$ and subsets $L \subseteq S \subseteq V$.

Objective: Find a minimum-cost tree T in G that contains S such that every $v \in L$ is a leaf of T .

1.2 Previous and Related Work

Fürer and Raghavachari [5] gave a $\Delta + 1$ approximation for the Minimum Degree Steiner Tree problem. The first result for the min-cost case is due to Ravi et al. [15]; they obtained an $(O(\log n) \cdot b(v), O(\log n))$ approximation for Degree-Constrained MST, namely, the degree of every node v in the output tree is $O(\log n) \cdot b(v)$ while its cost is $O(\log n)$ times the optimal cost. A major breakthrough was obtained by Goemans [7]; his algorithm computes a minimum cost spanning tree with degree at most $\Delta + 2$, with Δ the minimum possible degree.

In [10] and [17] is given an $O(n^\delta)$ approximation algorithm for the Minimum Degree k -Edge-Connected Subgraph problem, for any fixed $\delta > 0$.

It turned out that an extension of the iterative rounding method of Jain [9] may be the leading technique for degree-constrained problems. Singh and Lau [18] were the first to extend this method to achieve the best possible result for Min-Cost Minimum Degree MST; their tree has optimal cost while the maximum degree is at most $\Delta + 1$. Lau et al. [12] obtained a $(2b(v) + 3, 2)$ approximation for the Degree-Constrained Steiner Network problem, which was recently improved to

$(2b(v) + 2, 2)$ in [14]. Lau and Singh [13] further obtained a $(b(v) + O(r_{\max}), 2)$ approximation, where r_{\max} denotes the maximum connectivity requirement.

For directed graphs, Bansal et al. [1] gave an $(\lceil \frac{b(v)}{1-\epsilon} \rceil + 4, \frac{1}{\epsilon})$ -approximation scheme for the Degree-Constrained k -Outconnected Subgraph problem. Some extensions and slight improvements can be found in [16].

Now we mention some work which is directly relevant to our problems. The only known result for node-connectivity degree-constrained problems is by Feder, Motwani, and Zhu [4] who gave an algorithm that computes in $n^{O(k)}$ time a k -connected spanning subgraph H of G such that $\deg_H(v) = O(\log n) \cdot b(v)$. Their algorithm cannot handle costs. The special case $k = |S|$ of the Minimum Degree k -Arborescence problem was already studied in [3], where a $\tilde{O}(\sqrt{k})$ additive approximation was given. Their technique does not seem to extend to the case $k < |S|$. Hajiaghayi and Nasri [8] obtains a constant ratio for a very special case of Degree-Constrained Prize-Collecting Steiner Network problem when which the penalty function π is modular.

1.3 Our Results

Theorem 1. *The Degree-Constrained 2-Connected Subgraph problem admits a bicriteria $(6b(v) + 6, 4)$ -approximation algorithm; namely, a polynomial time algorithm that computes a 2-connected spanning subgraph H of G in which the degree of every node v is at most $6b(v) + 6$, and the cost of H is at most 4 times the optimal cost.*

To prove Theorem 1 we first compute a degree-constrained spanning tree J with +1 degree violation using the algorithm of [18]. Then we compute an augmenting edge-set I such that $J \cup I$ is 2-connected, using the iterative rounding method. To apply this method for degree constrained problems, one proves that any basic LP-solution $x > 0$ has an edge e with high x_e value, or there exists a node $v \in B$ such that $\deg_E(v)$ is close to $b(v)$. Otherwise, one shows a contradiction using the so called “token assignment” method. However, for node-connectivity problems this method differs from the one used for edge-connectivity problems, see Definition 2 and the paragraph after Lemma 6 in Section 2.

Theorem 2. *The Minimum Degree k -Arborescence problem admits a polynomial time approximation algorithm with ratio $O(\sqrt{(k \log k)/\Delta^*})$, where Δ^* is the optimal solution value, namely, the minimal maximum outdegree possible. Furthermore, the problem admits no $o(\log n)$ -approximation, unless $\text{NP} = \text{Quasi(P)}$.*

The algorithm in Theorem 2 uses single-commodity flows and solutions for Sub-modular Cover problem as sub-routines. These techniques may be of independent interest.

Integrity Gap of the Natural LP Relaxation for Minimum Degree k -MST.

To get some indication that the problem might be hard even on undirected graphs, consider the following natural LP-relaxation for Minimum Degree k -MST. The intended integral solution has $y_v = 1$ for nodes picked in the optimum tree T^* , $x_e = 1$ for $e \in T^*$, and d equal to the maximum degree of T^* .

$$\begin{array}{ll}
 \text{Minimize} & d \\
 \text{Subject to} & \sum_{v \neq r} y_v \geq k \\
 & \sum_{e \in \delta(S)} x_e \geq y_v \quad \forall v \in V \setminus \{r\} \quad \forall S \subset V, r \in S, v \notin S \\
 & \sum_{e \in \delta(v)} x_e \leq d \quad \forall v \in V \\
 & 0 \leq x_e, y_v \leq 1 \quad \forall e \in E \quad \forall v \in V
 \end{array} \tag{1}$$

We show that this LP-relaxation has integrality gap $\Omega(\sqrt{k})$ or $\Omega(n^{1/4})$ where $n = |V|$. Consider a rooted at r complete Δ -ary tree T of height h and let $k = \lfloor (\Delta + \Delta^2 + \dots + \Delta^h) / (\Delta + 1) \rfloor$. It is easy to see that giving $x_e = 1/(\Delta + 1)$ to all the edges $e \in T$ and $y_v = 1/(\Delta + 1)$ to all nodes $v \neq r$ satisfies all the constraints with fractional objective value $d = 1$. In order to cover k nodes, any integral tree however has to have a maximum degree of at least δ where $\delta + \delta(\delta - 1) + \delta(\delta - 1)^2 + \dots + \delta(\delta - 1)^{h-1} \geq k$. Such δ satisfies $\delta = \Omega(k^{1/h})$. Thus the optimum integral tree must have maximum degree $\Omega(k^{1/h})$ giving an integrality gap of $\Omega(k^{1/h})$. If we let $h = 2$, we get that $k = \Delta$ and $n = 1 + \Delta + \Delta^2$ and the integrality gap is $\Omega(\sqrt{\Delta})$ which is $\Omega(\sqrt{k})$ or $\Omega(n^{1/4})$.

Theorem 3. *The Degree and Diameter Bounded k -Tree problem admits no $o(\log n)$ -approximation algorithm, unless $\text{NP} = \text{Quasi(P)}$. This is so even for the special case in which some optimal solution tree has diameter 4.*

Let $\delta_F(A)$ denote the set of edges in F between A and $V \setminus A$. For $i \in K$ let $A \odot i$ denote that $|A \cap \{u_i, v_i\}| = 1$. Menger’s Theorem for edge-connectivity (see [11]) states that for a node pair u_i, v_i of a graph $H = (V, E)$ we have $\lambda_H(u_i, v_i) = \min_{A \odot i} |\delta_E(A)|$. Hence if $\lambda_H(u_i, v_i) \geq r_i$ for a graph $H = (V, E)$, then for any A with $A \odot i$ we must have $|\delta_E(A)| \geq r_i$. A standard “cut-type” LP-relaxation for Degree-Constrained Steiner Network problem is as follows.

$$\begin{array}{ll}
 \text{Minimize} & \sum_{e \in E} c_e x_e \\
 \text{Subject to} & \sum_{e \in \delta_E(A)} x_e \geq r_i(A) \quad \forall i \quad \forall A \odot i \\
 & \sum_{e \in \delta_E(v)} x_e \leq b(v) \quad \forall v \\
 & x_e \in [0, 1] \quad \forall e
 \end{array} \tag{2}$$

Theorem 4. *Suppose that for an instance of Prize-Collecting Degree-Constrained Steiner Network for any $\mathcal{P}' \subseteq \mathcal{P}$ the following holds. For an instance of Degree-Constrained Steiner Network defined by \mathcal{P}' there exists a polynomial-time algorithm that computes a subgraph H' of cost at most ρ times the optimal value of LP (2) with requirements restricted to \mathcal{P}' such that $\text{deg}_{H'}(v) \leq \alpha b(v) + \beta$ for*

all $v \in V$. Then we can compute in polynomial time numbers c^* and π^* with $c^* + \pi^* \leq \text{opt}$, such that for any $\mu \in (0, 1)$, Prize-Collecting Degree-Constrained Steiner Network admits a polynomial time algorithm that computes a subgraph H such that $\text{val}(H) \leq \frac{\rho}{1-\mu}c^* + \frac{1}{\mu}\pi^*$ and $\text{deg}_H(v) \leq \frac{\alpha}{1-\mu}b(v) + \beta$ for all $v \in V$.

The above theorem can be used along with the following known results. Louis and Vishnoi [14] obtain $\rho = 2, \alpha = 2, \beta = 2$ for Degree-Constrained Steiner Network. Lau and Singh [13] obtain $\rho = 2, \alpha = 1, \beta = 3$ for Degree-Constrained Steiner Forest and $\rho = 2, \alpha = 1, \beta = 6r_{\max} + 3$ for Degree-Constrained Steiner Network where r_{\max} is the maximum requirement.

In the Group Steiner Tree problem we are given a root s and a collection $\mathcal{P} = \{S_1, \dots, S_k\}$ of node-subsets (groups), and seek a minimum-cost subgraph H of G that contains a path from s to each group S_i . The Group Steiner Tree admits an $O(\log^3 n)$ approximation [6].

Theorem 5. *If Group Steiner Tree admits approximation ratio ρ then so does Leaf-Constrained Steiner Tree. Consequently, Leaf-Constrained Steiner Tree admits an $O(\log^3 n)$ -approximation algorithm.*

Theorems [1] and [2] are proved in Sections [2] and [3], respectively. Theorems [3], [4] and [5], will be proved in the full version, due to space limitations.

2 Algorithm for Degree-Constrained 2-Connected Subgraph

We start by considering the problem of augmenting a connected graph $J = (V, E_J)$ by a minimum-cost edge-set $I \subseteq E$ such that $\text{deg}_I(v) \leq b(v)$ for all $v \in V$ and such that $J \cup I$ is 2-connected.

Definition 1. *For a node v of J let $\mu_J(v)$ be the number of connected components of $J \setminus \{v\}$; v is a cut-node of J if $\mu_J(v) \geq 2$.*

Note that $\mu_J(v) \leq \text{deg}_J(v)$ for every $v \in V$. For $S \subseteq V$ let $\Gamma_J(S)$ denote the set of neighbors of S in J . Let s be a non-cut-node of J . A set $S \subseteq V \setminus \{s\}$ is *violated* if $|\Gamma_J(S)| = 1$ and $s \notin \Gamma_J(S)$. Let \mathcal{S}_J denote the set of violated sets of J . Recall that $\delta_F(S)$ denotes the set of edges in F between S and $V \setminus S$. For $S \in \mathcal{S}_J$ let $\zeta_F(S)$ denote the set of edges in F with one endnode in S and the other in $V \setminus (S \cup \Gamma_J(S))$. By Menger’s Theorem, $J \cup I$ is 2-connected if, and only if, $|\zeta_I(S)| \geq 1$ for every $S \in \mathcal{S}_J$. Thus a natural LP-relaxation for our augmentation problem is $\tau = \min\{c \cdot x : x \in P(J, b)\}$, where $P(J, b)$ is the polytope defined by the following constraints:

$x(\zeta_E(S)) \geq 1$	for all $S \in \mathcal{S}_J$
$x(\delta_E(v)) \leq b(v)$	for all $v \in B$
$0 \leq x_e \leq 1$	for all $e \in E$

Theorem 6. *There exists a polynomial time algorithm that given an instance of Degree-Constrained 2-Connected Subgraph and a connected spanning subgraph (V, J) of G computes an edge set $I \subseteq E \setminus J$ such that $c(I) \leq 3\tau$ and such that $\deg_I(v) \leq 2\mu_J(v) + 3b(v) + 3$ for all $v \in V$.*

Theorem 6 will be proved later. Now we show how to deduce the promised approximation ratio from it. Consider the following two phase algorithm.

Phase 1: With degree bounds $b(v)$, use the $(b(v)+1, 1)$ -approximation algorithm of Singh and Lau [18] for the Degree Constrained Steiner Tree problem to compute a spanning tree J in G .

Phase 2: Use the algorithm from Theorem 6 to compute an augmenting edge set I such that $H = J \cup I$ is 2-connected.

We prove the ratio. We have $c(J) \leq \text{opt}$ and $c(I) \leq 3\tau$, hence $c(H) = c(J) + c(I) \leq 4\text{opt}$. We now prove the approximability of the degrees. Let $v \in V$. Note that $\mu_J(v) \leq \deg_J(v) \leq b(v) + 1$. Thus we have

$$\deg_I(v) \leq 2\mu_J(v) + 3b(v) + 3 \leq 2(b(v) + 1) + 3b(v) + 3 = 5b(v) + 5.$$

This implies

$$\deg_H(v) \leq \deg_J(v) + \deg_I(v) \leq b(v) + 1 + 5b(v) + 5 = 6b(v) + 6.$$

In the rest of this section we will prove the following statement, that implies Theorem 6.

Lemma 1. *Let x be an extreme point of the polytope $P(J, b)$ with $0 < x_e < \frac{1}{3}$ for every $e \in E$. Then there is $v \in B$ such that $\deg_E(v) \leq 2\mu_J(v) + 3b(v) + 3$.*

Lemma 1 implies Theorem 6 as follows. Given a partial solution I and a parameter $\alpha \geq 1$, the residual degree bounds are $b_I^\alpha(v) = b(v) - \deg_I(v)/\alpha$. The following algorithm starts with $I = \emptyset$ and performs iterations. In every iteration, we work with the residual polytope $P(\mathcal{S}_{J \cup I}, b_I^\alpha)$, and remove some edges from E and/or some nodes from B , until E becomes empty. Let $\alpha = 3$ and $\beta(v) = 2\mu_J(v) + 3$ for all $v \in B$. It is easy to see that for any edge-set $I \subseteq E$ we have $\mu_{J \cup I}(v) \leq \mu_J(v)$ for every $v \in V$.

Algorithm as in Theorem 6

Input: A connected graph (V, J) , an edge-set E on V with costs $\{c_e : e \in E\}$, degree bounds $\{b(v) : v \in V\}$, and non-negative integers $\{\beta(v) : v \in V\}$.

Initialization: $I \leftarrow \emptyset$.

If $P(J, b) = \emptyset$, then return "UNFEASIBLE" and STOP.

While $E \neq \emptyset$ do:

1. Find a basic solution $x \in P(\mathcal{S}_{J \cup I}, b_I^\alpha)$.
2. Remove from E all edges with $x_e = 0$.
3. Add to I and remove from E all edges with $x_e \geq 1/\alpha$.
4. Remove from B every $v \in B$ with $\deg_E(v) \leq \alpha b_I^\alpha(v) + \beta(v)$.

EndWhile

Return I .

It is a routine to prove the following statement.

Lemma 2. *The above algorithm computes an edge set I such that $J \cup I$ is 2-connected, $c(I) \leq \alpha\tau$, and $\deg_J(v) \leq \alpha b(v) + \beta(v)$ for all $v \in B$.*

It remains to prove Lemma 1. The following statement is well known and can be proved using standard ‘‘uncrossing’’ methods.

Lemma 3. *The family \mathcal{S}_J of violated sets is uncrossable, namely, for any $X, Y \in \mathcal{S}_J$ we have $X \cap Y, X \cup Y \in \mathcal{S}_J$ or $X \setminus Y, Y \setminus X \in \mathcal{S}_J$.*

Recall that a set-family \mathcal{L} is laminar if for any distinct sets $X, Y \in \mathcal{L}$ either $X \subset Y$, or $Y \subset X$, or $X \cap Y = \emptyset$. Any laminar family \mathcal{L} defines a partial order on its members by inclusion; we carry the usual notion of children, descendants, and leaves of laminar set families. The following statement is proved using a standard ‘‘uncrossing’’ argument.

Lemma 4. *For any basic solution $x \in P(J, b)$ with $0 < x(e) < 1$ for all $e \in E$, there exists a laminar family $\mathcal{L} \subseteq \mathcal{S}$ and $T \subseteq B$, such that x is the unique solution to the linear equation system:*

$$\begin{aligned} x(\zeta_E(S)) &= 1 && \text{for all } S \in \mathcal{L} \\ x(\delta_E(v)) &= b(v) && \text{for all } v \in T \end{aligned}$$

Thus $|\mathcal{L}| + |T| = |E|$ and the characteristic vectors of $\{\zeta_E(S) : S \in \mathcal{L}\}$ are linearly independent.

Let x , \mathcal{L} , and T be as in Lemma 4. Let I' is the set of edges in E with exactly one endnode in B , I'' is set of edges in E with both endnodes in B , and $F = E \setminus (I' \cup I'')$.

Lemma 5. *Let $\{\beta(v) : v \in V\}$ be integers. Then there is $v \in B$ such that $\deg_E(v) \leq \alpha b(v) + \beta(v)$, if the following property holds:*

$$|\mathcal{L}| < \frac{1}{2}(\beta(B) - |B|) + \alpha x(I'') + \frac{1}{2}|I'| + \frac{1}{2}\alpha x(I') + |F| \tag{3}$$

Proof. Note that

$$\begin{aligned} \sum_{v \in B} (\deg_E(v) - \alpha b(v)) &\leq \sum_{v \in B} (\deg_E(v) - \alpha x(\delta_E(v))) \\ &= \sum_{v \in B} (\deg_{I'}(v) + \deg_{I''}(v)) - \alpha \sum_{v \in B} (x(\delta_{I'}(v)) + x(\delta_{I''}(v))) \\ &= |I'| + 2|I''| - \alpha x(I') - 2\alpha x(I'') . \end{aligned}$$

Thus $|I'| + 2|I''| - \alpha x(I') - 2\alpha x(I'') < \beta(B) + |B|$ implies that $\deg_E(v) \leq \alpha b(v) + \beta(v)$ for some $v \in B$. Adding $|I'| + 2|F|$ to both sides gives $2(|I'| + |I''| + |F|) < \beta(B) + |B| + \alpha x(I') + 2\alpha x(I'') + |I'| + 2|F|$. Note that $|I'| + |I''| + |F| = |E| = |\mathcal{L}| + |T|$. Consequently, since $|T| \leq |B|$, it is sufficient to prove that

$$2(|\mathcal{L}| + |B|) < \beta(B) + |B| + \alpha x(I') + 2\alpha x(I'') + |I'| + 2|F| .$$

Multiplying both sides by $\frac{1}{2}$ and rearranging terms gives (3).

Let us say that an edge e covers $S \in \mathcal{L}$ if e has one endnode in S and the other in $V \setminus (S \cup \Gamma_J(S))$. Given $S \in \mathcal{L}$ and an edge set E we will use the following notation.

- \mathcal{C} is the set of children in \mathcal{L} of S ,
- E_S is the set of edges in E covering S but not a child of S ,
- E_C is the set of edges in E covering some child of S but not S .

To show that there is $v \in B$ with $\deg_E(v) \leq \alpha b(v) + \beta(v)$ for $\alpha = 3$ and $\beta(v) = 2\mu_J(v) + 3$, we assign a certain amount of tokens to edges in E and nodes in B , such that the total amount of tokens does not exceed the right hand side of (3). A part of tokens assigned to an edge can be placed at some endnode or at the middle of the edge. A set $S \in \mathcal{L}$ gets the tokens placed at an endnode v of an edge e if $e \in E_S$ and $v \in S$, or if $e \in E_C$ and v does not belong to a child of S . S gets the tokens placed at the middle of e if $e \in E_C$. It is easy to verify that no two sets get the same token part of an edge. S gets also a token from $v \in B$ by the following rule.

Definition 2. We say that $S \in \mathcal{L}$ owns a node v if $v \in S$ but v is not in a child of S . We say that S shares v if $v \in \Gamma_J(S)$.

Clearly, if $S \in \mathcal{L}$ owns v then no other set in \mathcal{L} owns v . Note that if S shares v , then no ancestor or descendant of S in \mathcal{L} shares v . This implies the following.

Lemma 6. For any $v \in B$, the number of sets in \mathcal{L} sharing v is at most $\mu_J(v)$.

Thus if $\mu_J(v) + 1$ tokens are assigned to v , then every set $S \in \mathcal{L}$ that owns or shares v can get 1 token from v . We will argue by induction that we can redistribute the tokens of S and its descendants in \mathcal{L} such that every proper descendant of S in \mathcal{L} gets at least 1 token and S gets 2 tokens. This differs from the usual token distribution in edge-connectivity problems, where nodes are owned but not shared. In node-connectivity, the cut-nodes may be shared by many members of \mathcal{L} , and in our case, by at most $\mu_J(v)$ members.

For $\beta(v) = 2\mu_J(v) + 3$ and $\alpha = 3$, (3) becomes:

$$|\mathcal{L}| < \mu_J(B) + |B| + 3x(I'') + \frac{1}{2}|I'| + \frac{3}{2}x(I') + |J|. \tag{4}$$

Initial token assignment (total amount of tokens \leq the r.h.s of (4)):

- $\mu_J(v) + 1$ tokens to every $v \in B$,
- x_e tokens to each endnode in B of an edge e ,
- $\frac{1}{2}$ token to each endnode in $V \setminus B$ of an edge e .

Lemma 7. We can redistribute the tokens of S and its descendants in \mathcal{L} such that every proper descendant of S in \mathcal{L} gets at least 1 token and S gets 2 tokens.

Proof. Since $0 < x_e < \frac{1}{3}$ for every $e \in E$, $|\zeta_E(S)| \geq 4$ for every $S \in \mathcal{L}$. Suppose that S is a leaf. If $S \cap B = \emptyset$, then S gets $\frac{1}{2}$ token from an endnode of every edge in $\zeta_E(S)$, and in total at least 2 tokens. If there is $v \in S \cap B$ then S owns v and gets 1 token from v . S gets $x(\zeta_E(S)) = 1$ tokens from edges in $\zeta_E(S)$. Consequently, S gets in total at least 2 tokens, as claimed. Now suppose that S is not a leaf. S gets $|\mathcal{C}|$ tokens from its children, hence if $|\mathcal{C}| \geq 2$ then we are done. Thus we are left with the case that S has a unique child C , and needs 1 token not from C . If S contains or shares some $v \in B$ then we are done. Otherwise, S gets $\frac{1}{2}$ token from the corresponding endnode of each edge in $E_S \cup E_C$. By the linear independence an integrality of cuts $|E_S \cup E_C| \geq 2$, hence S gets the desired token from the edges in $E_S \cup E_C$.

The proof of Lemma 1, and thus also of Theorem 1, is now complete.

3 Algorithm for Minimum Degree k -Arborescence

We may assume that in the input graph G every node is reachable from the root s , that every terminal has indegree 1 and outdegree 0, and that the set of terminal of every arborescence T coincides with the set of leaves of T . Let $U = V \setminus S$. Before describing the algorithm, we need some definitions.

Definition 3. For $W \subseteq U$ and an integer parameter $\alpha \geq 1$ the network $F_\alpha(W)$ with source s' and sink t' is obtained from G as follows.

1. Assign infinite capacity to every edge of G and capacity α to every node in U .
2. Add a new node s' and add new edges of capacity α each from s' to every node in W .
3. Add two new nodes t, t' , add an edge of capacity 1 from every terminal to t , and add an edge of capacity k from t to t' .

Our algorithm runs with an integer parameter α set eventually to

$$\alpha = \left\lceil \sqrt{k \cdot \Delta^* \cdot (\ln k + 1)} \right\rceil . \tag{5}$$

Although Δ^* is not known, $\Delta^* \leq k$, and our algorithm applies exhaustive search in the range $1, \dots, k$.

Recall that a set-function ν defined on subsets of a ground-set U is *submodular* if $\nu(A) + \nu(B) \geq \nu(A \cup B) + \nu(A \cap B)$ for all $A, B \subseteq U$. Consider the following well known generic problem (for our purposes we state only the unweighted version).

Submodular Cover

Instance: A finite set U and a non-decreasing submodular function $\nu : 2^U \mapsto \mathbb{Z}$.

Objective: A minimum-size subset $W \subseteq U$ such that $\nu(W) = \nu(U)$.

The Submodular Cover Greedy Algorithm (for the unweighted version) starts with $W = \emptyset$ and while $\nu(W) < \nu(U)$ repeatedly adds to W an element $u \in U \setminus W$ that maximizes $\nu(W \cup \{u\}) - \nu(W)$. At the end, W is output. It is proved in [19] that the Greedy Algorithm for Submodular Cover has approximation ratio $\ln \max_{u \in U} \nu(\{u\}) + 1$.

A generalization of the following statement is proved in [2].

Lemma 8 ([2]). *For $W \subseteq U$ let $\nu_\alpha(W)$ be the maximum st-flow value in the network $F_\alpha(W)$. Then ν_α is non-decreasing and submodular, and $\nu_\alpha(U) \leq k$.*

The algorithm is as follows.

1. Execute the **Submodular Cover Greedy Algorithm** with $U = V \setminus S$ and with $\nu = \nu_\alpha$; let $W \subseteq U$ be the node-set computed.
2. Let f be a maximum integral flow in $F_\alpha(W)$ and let $J_W = \{e \in E : f(e) > 0\}$ be the set of those edges in E that carry a positive flow in $F_\alpha(W)$.
Let T_W be an inclusion-minimal arborescence in G rooted at s containing W .
3. Return any k -arborescence contained in the graph $(V, J_W) \cup T_W$.

In the rest of this section we prove that the graph $(V, J_W) \cup T_W$ indeed contains a k -arborescence, and that for any integer $\alpha \geq 1$ it has maximum outdegree at most $\alpha + (\ln k + 1) \cdot k \Delta^* / \alpha$.

This implies the approximation ratio $\alpha / \Delta^* + (\ln k + 1) \cdot k / \alpha = O(\sqrt{(k \log k) / \Delta^*})$ for α given by (5).

Definition 4. *A collection \mathcal{T} of sub-arborescences of an arborescence T is an α -leaf-covering decomposition of T if the arborescences in \mathcal{T} are pairwise node-disjoint, every leaf of T belongs to exactly one of them, and each of them has at most α leaves.*

Lemma 9. *Suppose that G contains a k -arborescence that admits an α -leaf-covering decomposition \mathcal{T} . Let R be the set of roots of the arborescences in \mathcal{T} . Then $\nu_\alpha(R) = k$, and for the set W computed by the algorithm the following holds:*

- (i) $\nu_\alpha(W) = k$ and thus the graph $J_W \cup T_W$ contains a k -arborescence.
- (ii) The graph $(V, J_W) \cup T_W$ has maximum outdegree at most $\alpha + |\mathcal{T}| \cdot (\ln k + 1)$.

Proof. We prove that $\nu_\alpha(R) = k$. For a terminal v in T , let $r_v \in R$ be the root of the (unique) arborescence $T_v \in \mathcal{T}$ that contains v , and let P_v be the path in $F_\alpha(R)$ that consists of: the edge $s'r_v$, the unique path from r_v to v in T_v , and the edges vt' and $t't$. Let f be the flow obtained by sending for every terminal v of T one flow unit along P_v . Then f has value k , since T has k terminals. We verify that f obeys the capacity constraints in $F_\alpha(R)$. For every $r \in R$, the arborescence $T_r \in \mathcal{T}$ which root is r , has at most α terminals; hence the edge $s'r$ carries at most α flow units, which does not exceed its capacity α . This also implies that the capacity α on all nodes in U is met. For every terminal v of T , the edge vt carries one flow unit and has capacity 1. The edge $t't$ carries k flow units and has capacity k . Other edges have infinite capacity.

We prove (i). By Lemma 8, ν_α is non-decreasing and $\nu_\alpha(U) \leq k$. As $\nu_\alpha(U) \geq \nu_\alpha(R) = k$ and $\nu_\alpha(W) = \nu_\alpha(U)$, we conclude that $\nu(W) = k$. This implies that in the graph (V, J_W) , k terminals are reachable from W , and (i) follows.

We prove (ii). In the graph (V, J_W) , the outdegree of any node is at most α . This follows from the capacity α on any node in U . We have $|W| \leq |R| \cdot (\ln k + 1) = |\mathcal{T}| \cdot (\ln k + 1)$, by Lemma 9 and the approximation ratio of the **Submodular Cover Greedy Algorithm**. Since T_W is an arborescence with leaf-set W , the maximum outdegree of T_W is at most $|W| \leq |\mathcal{T}| \cdot (\ln k + 1)$. The statement follows.

The following lemma implies that the optimal tree T^* admits an α -leaf-covering decomposition \mathcal{T} of size $|\mathcal{T}| \leq k\Delta^*/\alpha$ for any $\alpha \geq 1$. This together with Lemma 9 concludes the proof of Theorem 2.

Lemma 10. *Any arborescence T with k leaves and maximum outdegree Δ admits an α -leaf-covering decomposition \mathcal{T} of size $|\mathcal{T}| \leq \Delta \cdot \lfloor k/(\alpha + 1) \rfloor + 1$, for any integer $\alpha \geq 1$.*

Proof. For a node r of an arborescence T with root s let us use the following notation: T_r is the sub-arborescence of T with root r that contains all descendants of r in T , and P_r is the set of internal nodes in the ar -path in T , where a is the closest to r ancestor of r that has outdegree at least 2. Let us say that a node $u \in U$ of T is α -critical if T_u has more than α leaves, but no child of u has this property. It is easy to see that T has an α -critical node if, and only if, T has more than α leaves.

Consider the following algorithm. Start with $\mathcal{T} = \emptyset$. While T has an α -critical node u do the following: add T_r to \mathcal{T} for every child r of u , and remove T_u and P_u from T (note that since we remove P_u no new leaves are created). When the while loop ends, if T is nonempty, add the remaining arborescence $T = T_s$ (which now has at most α leaves) to \mathcal{T} .

By the definition, the arborescences in \mathcal{T} are pairwise node-disjoint, every leaf of T belongs to exactly one of them, and each of them has at most α leaves. It remains to prove the bound on \mathcal{T} . In the loop, when we consider an α -critical node u , at least $\alpha + 1$ leaves are removed from T and at most Δ arborescences are added to \mathcal{T} . Hence $|\mathcal{T}| \leq \Delta \cdot \lfloor k/(\alpha + 1) \rfloor$ at the end of the loop. At most one additional arborescence is added to \mathcal{T} after the loop. The statement follows.

References

1. Bansal, N., Khandekar, R., Nagarajan, V.: Additive guarantees for degree bounded directed network design. *SIAM J. Computing* 39(4), 1413–1431 (2009)
2. Bar-Ilan, J., Kortsarz, G., Peleg, D.: Generalized submodular cover problems and applications. *Theoretical Computer Science* 250, 179–200 (2001)
3. Elkin, M., Kortsarz, G.: An approximation algorithm for the directed telephone multicast problem. *Algorithmica* 45(4), 569–583 (2006)
4. Feder, T., Motwani, R., Zhu, A.: k -connected spanning subgraphs of low degree. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 041 (2006)
5. Fürer, M., Raghavachari, B.: Approximating the minimum-degree Steiner tree to within one of optimal. *J. Algorithms* 17(3), 409–423 (1994)
6. Garg, N., Konjevod, G., Ravi, R.: A polylogarithmic approximation algorithm for the group Steiner tree problem. *J. Algorithms* 37(1), 66–84 (2000)
7. Goemans, M.: Bounded degree minimum spanning trees. In: *FOCS*, pp. 273–282 (2006)
8. Hajiaghayi, M., Nasri, A.: Prize-collecting steiner networks via iterative rounding. In: López-Ortiz, A. (ed.) *LATIN 2010*. LNCS, vol. 6034, pp. 515–526. Springer, Heidelberg (2010)
9. Jain, K.: A factor 2 approximation algorithm for the generalized Steiner network problem. *Combinatorica* 21(1), 39–60 (2001)

10. Klein, P., Krishnan, R., Raghavachari, B., Ravi, R.: Approximation through local optimality: designing networks with small degree. *Networks* 44, 203–215 (2004)
11. Kortsarz, G., Nutov, Z.: Approximating minimum cost connectivity problems. In: Gonzalez, T.F. (ed.) *Approximation Algorithms and Metaheuristics*, ch. 58, Chapman and Hall/CRC (2007)
12. Lau, L.C., Naor, J., Salavatipour, M., Singh, M.: Survivable network design with degree or order constraints. *SIAM J. Computing* 39(3), 1062–1087 (2009)
13. Lau, L.C., Singh, M.: Additive approximation for bounded degree survivable network design. In: *STOC*, pp. 759–768 (2008)
14. Louis, A., Vishnoi, N.: Improved algorithm for degree bounded survivable network design problem. In: Kaplan, H. (ed.) *SWAT 2010*. LNCS, vol. 6139, pp. 408–419. Springer, Heidelberg (2010)
15. Marathe, M.V., Ravi, R., Sundaram, R., Ravi, S.S., Rosenkrantz, D.J., Hunt III, H.B.: Bicriteria network design problems. *J. Algorithms* 28(1), 142–171 (1998)
16. Nutov, Z.: Approximating directed weighted-degree constrained networks. *Theoretical Computer Science* 412(8–10), 901–912 (2011)
17. Ravi, R., Raghavachari, B., Klein, P.: Approximation through local optimality: Designing networks with small degree. In: Shyamasundar, R.K. (ed.) *FSTTCS 1992*. LNCS, vol. 652, pp. 279–290. Springer, Heidelberg (1992)
18. Singh, M., Lau, L.C.: Approximating minimum bounded degree spanning trees to within one of optimal. In: *STOC*, pp. 661–670 (2007)
19. Wolsey, L.A.: An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica* 2, 385–393 (1982)

Improved Approximation Algorithms for the Min-Max Tree Cover and Bounded Tree Cover Problems*

M. Reza Khani¹ and Mohammad R. Salavatipour²

¹ Dept. of Computing Science, Univ. of Alberta
khani@ualberta.ca

² Toyota Tech. Inst. at Chicago, and Dept. of Computing Science, Univ. of Alberta
mreza@cs.ualberta.ca

Abstract. In this paper we provide improved approximation algorithms for the Min-Max Tree Cover and Bounded Tree Cover problems. Given a graph $G = (V, E)$ with weights $w : E \rightarrow \mathbb{N}^+$, a set T_1, T_2, \dots, T_k of subtrees of G is called a tree cover of G if $V = \bigcup_{i=1}^k V(T_i)$. In the Min-Max k -tree Cover problem we are given graph G and a positive integer k and the goal is to find a tree cover with k trees, such that the weight of the largest tree in the cover is minimized. We present a 3-approximation algorithm for this improving the two different approximation algorithms presented in [15] with ratio 4. The problem is known to have an APX-hardness lower bound of $\frac{3}{2}$ [12]. In the Bounded Tree Cover problem we are given graph G and a bound λ and the goal is to find a tree cover with minimum number of trees such that each tree has weight at most λ . We present a 2.5-approximation algorithm for this, improving the 3-approximation bound in [4].

1 Introduction

The study of problems in which the vertices of a given graph are needed to be covered with special subgraphs, such as trees, paths, or cycles, with a bound on the number of subgraphs used or their weights has attracted a lot of attention in operations research and computer science community. Such problems arise naturally in many applications such as vehicle routing. As an example, in a vehicle routing problem with min-max objective, we are given a weighted graph $G = (V, E)$ in which each node represents a client. The goal is to dispatch a number of service vehicles to service the clients and the goal is to minimize the largest client waiting time, which is equivalent to minimizing the total distance traveled by the vehicle which has traveled the most. Observe that the subgraph traveled by each vehicle is a walk that can be approximated with a tree. This problem, under the name of “Nurse station location”, was the main motivation in [5] to study these problems. In a different scenario, we may want to guarantee

* The authors were supported by NSERC. Work of the second author was additionally supported by an Alberta Ingenuity New Faculty Award.

an upper bound for the service time; so we are given a bound on the distance traveled by each vehicle and the objective is to minimize the number of required vehicles needed to satisfy this guarantee. Min-max and bounded vehicle routing problems are part of an active body of research in the literature and have several application (see e.g. [2,5,11,12] and the references there).

In this paper we consider Min-Max k -Tree Cover Problem (MM k TC) and Bounded Tree Cover Problem (BTC) defined formally below. Suppose we are given an undirected graph $G = (V, E)$ and a weight function $w : E \rightarrow \mathbb{N}^+$. For every subgraph H of G we use $V(H)$ and $E(H)$ to denote the set of vertices and edges of H , respectively. A set T_1, T_2, \dots, T_k of subtrees of G is called a tree cover of G if every vertex of V appears in at least one T_i ($1 \leq i \leq k$), i.e. $V = \bigcup_{i=1}^k V(T_i)$. Note that the trees in a tree-cover are not necessarily edge-disjoint (thus may share vertices too). The weight of a tree T_i is $W(T_i) = \sum_{e \in T_i} w(e)$. In the Min-Max k -Tree Cover problem (MM k TC) we are given the weighted graph G and a positive integer k and the goal is to find a tree cover with k trees, which we call a k -tree cover, such that the weight of the largest tree in the cover is minimized. In the Bounded Tree Cover problem (BTC), we are given the weight G and a parameter λ and the goal is to find a tree cover with minimum number of trees such that the weight of every tree in the cover is at most λ . There are other variations of these problems in which one wants to cover the vertices of a graph with paths or cycles (instead of trees), however the known algorithms for these variations (e.g. see [1]) solve the problem for tree versions first and then take a walk of the trees to obtain a path. So apart from their real world applications [5], covering graphs with trees have been the main step for covering graphs with paths and cycles.

Related Works: Even et al. [5] and Arkin et al. [1] gave two different 4-approximation algorithms for MM k TC. It is shown that MM k TC is APX-hard in [12], specifically a lower bound of $\frac{3}{2}$ is given. The best approximation factor for BTC is due to Arkin et al. [1] which give a 3-approximation algorithm. It is easy to see that BTC is APX-hard even in the case when G is a weighted tree with height one, by an easy reduction from the bin packing problem.

Even et al. [5] give a 4-approximation algorithm for the rooted version of MM k TC in which k vertices are given in input and each tree of the trees in a k -tree cover has to be rooted at one of them. Nagamochi and Okada [10] give a $(3 - \frac{2}{k+1})$ -approximation algorithm for MM k TC when all the trees have to be rooted at a given vertex r . They also give a $(2 - \frac{2}{k+1})$ -approximation algorithm for MM k TC when the underlying metric is a tree and $(2 + \epsilon)$ -approximation algorithm for MM k TC when the underlying metric is a tree and each tree should be rooted at a certain vertex r .

In addition to trees, covering with other objects, such as tours, paths, and stars are studied in the literature. Frederickson et al. [6] gave an $(e + 1 - \frac{1}{k})$ -approximation algorithm for covering a metric graph with k tours rooted at a given vertex (called k -traveling salesperson problem or k -TSP) where e is the best approximation ratio for the classic TSP problem. Other different variations of min-max and bounded vehicle routing problems are also studied in the literature

(see e.g. [11,13,9,7]). Another related problem to k -TSP is called k -Traveling Repairman Problem (KTR) in which instead of minimizing the total lengths of the tour the objective function is to minimize the total latency of the nodes where the latency of each node is the distance traveled (time elapsed) before visiting that node for the first time. The case of $k = 1$ is known as the minimum latency problem. The best known approximation algorithm for $k = 1$ is 3.59 due to [3] and the best known approximation for KTR is $2(2 + \alpha)$ [4] where α is the best approximation ratio for the problem of finding minimum tree spanning k vertices a.k.a k -MST (see also [8] and the references there).

Our Result: In this paper we improve the approximation ratios for both $MMkTC$ and BTC problems.

Theorem 1. *There is a polynomial time 3-approximation algorithm for $MMkTC$.*

Theorem 2. *There is a polynomial time 2.5-approximation algorithm for BTC .*

2 Preliminaries

For a connected subgraph $H \subseteq G$ by tree weight of H we mean the weight of a minimum spanning tree (MST) of H and denote this value by $W_T(H)$. Note that this is different from the weight of H , i.e. $W(H)$ which is the sum of weights of *all* the edges of H . In every solution to either $MMkTC$ or BTC problem, we can replace every edge uv of a tree in the cover with the shortest path between u, v in the graph without increasing the cost of the tree and the solution still remains feasible. Therefore, without loss of generality, if the input graph is G and \tilde{G} is the shortest-path metric completion of G , we can assume that we are working with the complete graph \tilde{G} . Any solution to \tilde{G} can be transformed into a feasible solution of G (for $MMkTC$ or BTC) without increasing the cost. The following lemma will be useful in our algorithms for both the $MMkTC$ and BTC problems.

Lemma 1. *Suppose $G = (V, E)$ is a graph which has a k -tree cover $\mathcal{T} = \{T_1, \dots, T_k\}$, with maximum tree weight of λ and let $\lambda' \leq \lambda$ be a given parameter. Assume we delete all the edges e with $w(e) > \lambda'$ (call them heavy edges) and the resulting connected components be C_1, \dots, C_p . Then $\sum_{i=1}^p W_T(C_i) \leq k\lambda + (k - p)\lambda'$.*

Proof. Let $G' = \bigcup_{i=1}^p C_i$ be the graph after deleting the heavy edges. Each tree in \mathcal{T} might be broken into a number of subtrees (or parts) after deleting heavy edges; let \mathcal{T}' denote the set of these broken subtrees, $|\mathcal{T}'| = k'$, and n_i be the number of trees of \mathcal{T}' in component C_i . The total weight of the subtrees in \mathcal{T}' is at most $k\lambda - (k' - k)\lambda'$, since the weight of each tree in \mathcal{T} is at most λ and we have deleted at least $k' - k$ edges from the trees in \mathcal{T} each having weight at least λ' . In each component C_i we use the cheapest $n_i - 1$ edges that connect all the trees of \mathcal{T}' in C_i into one spanning tree of C_i . The weight of each of these

added edges is no more than λ' and we have to add a total of $k' - p$ such edges (over all the components) in order to obtain a spanning tree for each component C_i . Thus, the total weight of spanning trees of the components C_i 's is at most $k\lambda - (k' - k)\lambda' + (k' - p)\lambda' = k\lambda + (k - p)\lambda'$. \square

Through our algorithms we may need to break a large tree into smaller trees that cover the (vertices of) original tree, are edge-disjoint, and such that the weight of each of the smaller trees is bounded by a given parameter. We use the following lemma which is implicitly proved (in a slightly weaker form) in [5].

Lemma 2. *Given a tree T with weight $W(T)$ and a parameter $\beta > 0$ such that all the edges of T have weight at most β , we can edge-decompose T into trees T_1, \dots, T_k with $k \leq \max(\lfloor \frac{W(T)}{\beta} \rfloor, 1)$ such that $W(T_i) \leq 2\beta$ for each $1 \leq i \leq k$.*

Proof. The idea is to “split away” (defined below) trees of weight in interval $[\beta, 2\beta)$ until we are left with a tree of size smaller than 2β . This process of “splitting away” is explained in [5]. Consider T being rooted at an arbitrary node $r \in T$. For every vertex $v \in T$ we use T_v to denote the subtree of T rooted at v ; for every edge $e = (u, v)$ we use T_e to denote the subtree rooted at u which consist of T_v plus the edge e . Subtrees are called light, medium, or heavy depending on whether their weight is smaller than β , in the range $[\beta, 2\beta)$, or $\geq 2\beta$, respectively. For a vertex v whose children are connected to it using edges e_1, e_2, \dots, e_l splitting away subtree $T' = \bigcup_{i=1}^l T_{e_i}$ means removing all the edges of T' and vertices of T' (except v) from T and putting T' in our decomposition. Note that we can always split away a medium tree and put it in our decomposition and all the trees we place in our decomposition are edge-disjoint. So assume that all the subtrees of T are either heavy or light. Suppose T_v is a heavy subtree whose children are connected to v by edges e_1, e_2, \dots such that all subtrees T_{e_1}, T_{e_2}, \dots are light (if any of them is heavy we take that subtree). Let i be the smallest index such that $T' = \bigcup_{a=1}^i T_{e_a}$ has weight at least β . Note that T' will be medium as all T_{e_j} 's are light. We split away T' from T and repeat the process until there is no heavy subtree of T (so at the end the left-over T is either medium or light).

If $W(T) \leq 2\beta$ then we do not split away any tree (since the entire tree T is medium) and the theorem holds trivially. Suppose the split trees are T_1, T_2, \dots, T_d with $d \geq 2$ and $W(T_i) \in [\beta, 2\beta)$ for $1 \leq i < d$. The only tree that may have weight less than β is T_d . Note that in the step when we split away T_{d-1} the total weight of the remaining tree was at least 2β , therefore we can assume that the average weight of T_{d-1} and T_d is not less than β . Thus, the average weight of all T_i 's is not less than β which proves that d cannot be greater than $\lfloor \frac{W(T)}{\beta} \rfloor$. \square

3 A 3-Approximation Algorithm for MMkTC

In this section we prove Theorem 1. Before describing our algorithm we briefly explain the 4-approximation algorithm of [5]. Suppose that the value of the optimum solution to the given instance of MMkTC is OPT and let $\lambda \geq \text{OPT}$ be

a value that we have guessed as an upper bound for OPT. The algorithm of [5] will either produce a k -tree cover whose largest tree has weight at most 4λ or will declare that OPT must be larger than λ , in which case we adjust our guess λ . For simplicity, let us assume that G is connected and does not have any edge e with $w(e) > \lambda$. Let T be a MST of G and $\mathcal{T} = \{T_1, \dots, T_k\}$ be an optimum k -tree cover of G . We can obtain a spanning tree of G from \mathcal{T} by adding at most $k - 1$ edges between the trees of \mathcal{T} . This adds a total of at most $(k - 1)\lambda$ as each edge has weight at most λ . Thus, $W(T) \leq \sum_{i=1}^k W(T_i) + (k - 1)\lambda \leq (2k - 1)\lambda$. Therefore, if we start from a MST of G , say T , and we split away trees of size in $[2\lambda, 4\lambda)$ then we obtain a total of at most $(2k - 1)\lambda/2\lambda k \leq k$ trees each of which has weight at most 4λ . In reality the input graph might have edges of weight larger than λ . First, we delete all such edges (called heavy edges). This might make the graph disconnected. Let $\{G_i\}_i$ be the connected components of the graph after deleting these heavy edges and let T_i be a MST of G_i . For each component G_i the algorithm of [5] splits away trees of weight in $[2\lambda, 4\lambda)$. Using Lemma 2 one can obtain a k_i -tree cover of each G_i with $k_i \leq \max(W_T(G_i)/2\lambda, 1)$ with each tree having weight at most 4λ . A similar argument as the one above shows (Lemma 3 in [5]) that $\sum_i (k_i + 1) \leq k$. One can do a binary search for the smallest value of λ with $\lambda \geq \text{OPT}$ which yields a polynomial 4-approximation.

Now we describe our algorithm. As said earlier, we work with the metric graph \tilde{G} . We use OPT to denote an optimal solution and OPT to denote the weight of the largest tree in OPT. Similar to [5] we assume we have a guessed value λ for OPT and present an algorithm which finds a k -tree cover with maximum tree weight at most 3λ if $\lambda \geq \text{OPT}$. By doing a binary search for λ we obtain a 3-approximation algorithm. First, we delete all the edges e with $w(e) > \lambda/2$ to obtain graph G' . Let C_1, \dots, C_ℓ be the components of G' whose tree weight (i.e. the weight of a MST of that component) is at most λ (we refer to them as *light components*), and let $C_{\ell+1}, \dots, C_{\ell+h}$ be the components of G' with tree weight greater than λ (which we refer to as *heavy components*). The general idea of the algorithm is as follows: For every light component we do one of the following three: find a MST of it as one tree in our tree cover, or we decide to connect it to another light components with an edge of weight at most λ in which case we find a component with MST weight at most 3λ and put that MST as a tree in our solution, or we decide to connect a light component to a heavy component. For heavy components (to which some light components might have been attached) we split away trees with weight in $[\frac{3}{2}\lambda, 3\lambda)$. We can show that if this is done carefully, the number of trees is not too big. We explain the details below.

For every light component C_i let $w_{\min}(C_i)$ be the minimum edge weight (in graph \tilde{G}) between C_i and a heavy component if such an edge exists with weight at most λ , otherwise set $w_{\min}(C_i)$ to be infinity. We might decide to combine C_i with a heavy component (one to which C_i has an edge of weight $w_{\min}(C_i)$). In that case the tree weight of that heavy component will be increased by $A(C_i) = W_T(C_i) + w_{\min}(C_i)$. The following lemma shows how we can cover the set of heavy components and some subset of light components with a small number of trees whose weight is not greater than 3λ .

Lemma 3. *Let $L_s = \{C_{l_1}, \dots, C_{l_s}\}$ be a set of s light-components with bounded $A(C_i)$ values. If $\sum_{1 \leq i \leq s} A(C_{l_i}) + \sum_{\ell+1 \leq i \leq \ell+h} W_T(C_i) \leq x - h\frac{\lambda}{2}$, then we can cover all the nodes in the heavy-components and in components of L_s with at most $\lfloor \frac{2x}{3\lambda} \rfloor$ trees with maximum tree weight no more than 3λ .*

Proof. First we find a MST in each heavy component and in each component of L_s , then we attach the MST of each C_{l_i} to the nearest spanning tree found for heavy components. As we have h heavy components, we get a total of h trees, call them T_1, \dots, T_h . From the definition of $A(C_{l_j})$, the total weight of the constructed trees will be

$$\sum_{i=1}^h W(T_i) = \sum_{1 \leq j \leq s} A(C_{l_j}) + \sum_{\ell+1 \leq i \leq \ell+h} W_T(C_i) \leq x - h\frac{\lambda}{2}, \tag{1}$$

where the last inequality is by the assumption of lemma. Now to each of the h constructed trees we will apply the procedure of Lemma 2 with $\beta = \frac{3}{2}\lambda$ to obtain trees of weight at most 3λ . This gives at most $\sum_{1 \leq i \leq h} \max(\lfloor \frac{2W(T_i)}{3\lambda} \rfloor, 1)$ trees. To complete the proof of lemma it is sufficient to prove the following:

$$\sum_{1 \leq i \leq h} \max(\lfloor \frac{2W(T_i)}{3\lambda} \rfloor, 1) \leq \lfloor \frac{2x}{3\lambda} \rfloor. \tag{2}$$

Consider T_i for an arbitrary value of i . If T_i has been split into more than one tree, by Lemma 2 we know that the amortized weight of the split trees is not less than $\frac{3}{2}\lambda$. If T_i is not split, as T_i contains a spanning tree over a heavy component, $W(T_i) \geq \lambda$. Thus every split tree has weight at least $\frac{3}{2}\lambda$ excepts possibly h trees which have weight at least λ . Therefore, if the total number of split trees is r , they have a total weight of at least $r\frac{3}{2}\lambda - h\frac{\lambda}{2}$. Using Equation (1), it follows that r cannot be more than $\lfloor \frac{2x}{3\lambda} \rfloor$. \square

Definition 1. *For two given parameters a, b , graph H has $\ell + a + b$ nodes: ℓ (regular) nodes v_1, \dots, v_ℓ , where each v_i corresponds to a light component C_i , a dummy nodes called null nodes, and b dummy nodes called heavy nodes. We add an edge with weight zero between two regular nodes v_i and v_j in H if and only if $i \neq j$ and there is an edge in \tilde{G} with length no more than λ connecting a vertex of C_i to a vertex of C_j . Every null node is adjacent to each regular node v_i ($1 \leq i \leq \ell$) with weight zero. Every regular node $v_i \in H$ whose corresponding light component C_i has finite value of $A(C_i)$ is connected to every heavy node in H with an edge of weight $A(C_i)$. There are no other edges in H .*

Theorem 3. *Algorithm MMkTC (Figure 1) finds a k -tree cover with maximum tree weight at most 3λ , if $\lambda \geq \text{OPT}$.*

Proof. Through out the whole proof we assume $\lambda \geq \text{OPT}$. Consider an optimal k -tree cover OPT ; so each $T \in \text{OPT}$ has weight at most λ . First note that every tree $T \in \text{OPT}$ can have at most one edge of value larger than $\lambda/2$; therefore

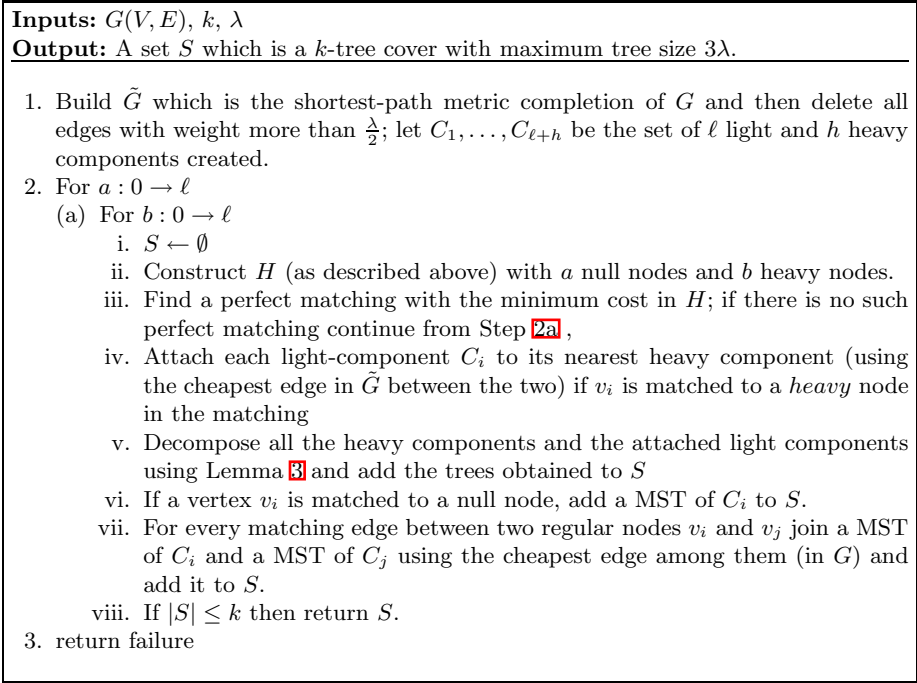


Fig. 1. MMkTC Algorithm

each $T \in \text{OPT}$ is either completely in one component C_i or has vertices in at most two components, in which case we say it is broken. If T is broken it consists of two subtrees that are in two components (we refer to the subtrees as *broken subtree or part of T*) plus an edge of weight $> \lambda/2$ connecting them; we call that edge the *bridge edge* of T . We characterize the optimal trees in the following way: a tree $T \in \text{OPT}$ is called light (heavy) if the entire tree or its broken subtrees (if it is broken) are in light (heavy) components only, otherwise if it is broken and has one part in a light component and one part in a heavy component then we call it a bad tree. We denote the number of light trees, heavy trees, and bad trees of OPT by $k_\ell, k_h,$ and k_b ; therefore $k_\ell + k_h + k_b = k$. We say that a tree $T \in \text{OPT}$ is incident to a component if the component contains at least one vertex of T (see Figure 2).

We define multi-graph $H' = (V', E')$ similar to how we defined H except that edges of H' are defined based on the trees in OPT . V' consists of ℓ vertices, one vertex v'_i for each light component C_i . For each light tree $T \in \text{OPT}$, if T is entirely in one component C_i we add a loop to v'_i and if T is broken and is incident to two light components C_i and C_j then we add an edge between v'_i and v'_j . So the total number of edges (including loops) is k_ℓ . There may be some isolated nodes (nodes without any edges) in H' , these are nodes whose corresponding light components are incident to only bad trees. Suppose M is

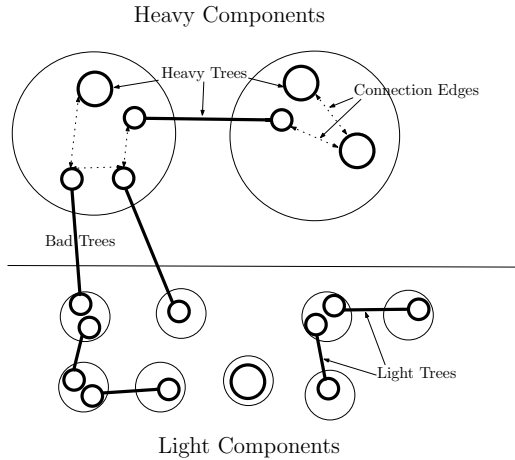


Fig. 2. Structure of G after deleting edges with length greater than $\frac{\lambda}{2}$. Each thin circle corresponds to a component and each solid circle corresponds to an optimum tree or a broken subtree (part) of an optimum tree.

a maximum matching in H' and let U be the set of vertices of H' that are not isolated and are not saturated by M . Because M is maximal, every edge in $E' \setminus M$ is either a loop or is an edge between a vertex in U and one saturated vertex. Therefore:

$$|M| + |U| \leq k_\ell. \tag{3}$$

Note that for every node v'_i (corresponding to a light component C_i) which is incident to a bad tree, that bad tree has a bridge edge (of weight at most λ) between its broken subtree in the light component (i.e. C_i) and its broken subtree in a heavy component. Therefore:

Lemma 4. *For every light component C_i which is incident to a bad tree, and in particular if v'_i is isolated, $A(C_i)$ is finite.*

We define the excess weight of each bad tree as the weight of its broken subtree in the light component plus the bridge edge. Let W_{excess} be the total excess weights of all bad trees of OPT. Note that W_{excess} contains $\sum_{v_i \text{ is isolated}} A(C_i)$, but it also contains the excess weight of some bad trees that are incident to a light component C_i for which v_i is not isolated. Thus:

$$W_{excess} \geq \sum_{v_i \text{ is isolated}} A(C_i). \tag{4}$$

Only at Steps 2(a)v, 2(a)vi, and 2(a)vii the algorithm adds trees to S . First we will show that each tree added to S has weight at most 3λ . At step 2(a)v, according to Lemma 3, all the trees will have weight at most 3λ . At Step 2(a)vi,

as C_i is a light components its MST will have weight at most λ . At Step [2\(a\)vii](#), the MST of C_i and C_j are both at most λ , and as v_i and v_j are connected in H there is an edge with length no more than λ connecting C_i and C_j ; thus the total weight of the tree obtained is at most 3λ . Hence, every tree in S has weight no more than 3λ . The only thing remained is to show that the algorithm will eventually finds a set S that has no more than k trees. We show that in the iteration at which $a = |U|$ and b is equal to the number of isolated nodes in H' : $|S| \leq k$.

Lemma 5. *The cost of the minimum perfect matching computed in step [2\(a\)iii](#) is no more than W_{excess} .*

Proof. Consider the iteration at which $a = |U|$ and b is the number of isolated nodes in H' . Define matching M as follows: for each $v'_i \in U$, $v_i \in H$ is matched to a null node in H , for each isolated node $v'_i \in H'$, $v_i \in H$ is matched to a heavy node in H (note that $A(C_i)$ is finite by Lemma [4](#)), for all other vertices $v'_i \in H'$, v'_i is saturated by M , so the corresponding $v_i \in H$ can be matched according to the matching M . The cost of this matching is $\sum_{v_i \text{ is isolated}} A(C_i) \leq W_{excess}$ by Equation [\(4\)](#) and that we find a minimum perfect matching in step [2\(a\)iii](#). \square

Note that the number of trees added to S at steps [2\(a\)vii](#) and [2\(a\)vi](#) is $|M|$ and $|U|$, respectively. Thus the total number of trees added to S at these two steps is at most $|M| + |U| \leq k_\ell$ by Equation [\(3\)](#). The weight of the minimum perfect matching found in [2\(a\)iii](#) represents the total weight we add to the heavy components in step [2\(a\)iv](#). By Lemma [5](#), we know that the added weight is at most W_{excess} . In Lemma [6](#) we bound the total weight of heavy components and the added extra weight of matching by $(k_h + k_t) * \frac{3}{2}\lambda - h\frac{\lambda}{2}$. Using Lemma [3](#) we know that we can cover them by at most $k_h + k_b$ trees. Thus the total number of trees added to S is at most $k_\ell + k_h + k_b = k$. \square

Lemma 6. $\sum_{\ell+1 \leq i \leq \ell+h} W_T(C_i) + W_{excess} \leq (k_h + k_b) * \frac{3}{2}\lambda - h\frac{\lambda}{2}$, if $\lambda \geq \text{OPT}$.

Proof. Again, we assume that $\lambda \geq \text{OPT}$. We show a possible way to form a spanning tree for each heavy component plus the light components attached to it. Then we bound the total weight of these spanning trees.

We can make a spanning tree over a heavy component C_i by connecting all the trees and broken subtrees of the optimum solution that are in that component by adding edges of weight at most $\lambda/2$ between them since each edge in C_i has weight at most $\lambda/2$ (see Figure [2](#)). Therefore, the tree weight of a heavy component can be bounded by the weight of optimal trees or broken subtrees inside it plus some edges to connect them. Suppose p trees of the heavy trees are broken and q of them are completely inside a heavy component; note that $p + q = k_h$. The rest of broken subtrees in heavy components are from bad trees. So overall we have $2p + q + k_b$ trees or broken subtrees in all the heavy components. Each of the q heavy trees that are not broken contribute at most $q\lambda$ to the left hand side. Those p heavy trees that are broken contribute at most $p\lambda/2$ to the left hand side since each of them has an edge of weight more than

$\lambda/2$ that is deleted and is between heavy components. By definition of W_{excess} , we can assume the contribution of all bad trees to the left hand side is at most $k_b\lambda$. Thus, the total weight of edges e such that e belongs to an optimal tree and also belongs to a heavy component or is part of W_{excess} (i.e. the broken part of a bad tree plus its bridge edge) is at most $(p + q + k_b)\lambda - p\frac{\lambda}{2}$.

Overall we have $2p + q + k_b$ trees or broken subtrees in all the heavy components. In order to form a spanning tree in each heavy component we need at most $2p + q + k_b - h$ edges connecting the optimal trees and broken subtrees in the heavy components, since we have h heavy components. Since each edge in a component has weight at most $\frac{\lambda}{2}$, the total weight of these edges will be at most $(2p + q + k_b - h)\frac{\lambda}{2}$. Therefore, the total weight of spanning trees over all heavy components plus W_{excess} will be at most $(p + q + k_b)\lambda - p\frac{\lambda}{2} + (2p + q + k_b - h)\frac{\lambda}{2} = (k_h + k_b) * \frac{3}{2}\lambda - h\frac{\lambda}{2}$. \square

We know that OPT can be at most $\sum_{e \in E} w(e)$. By Theorem 3 we know that if $\lambda \geq \text{OPT}$, Algorithm 1 will find a k -tree cover with maximum tree weight at most 3λ . If $\lambda < \text{OPT}$ the algorithm may fail or may provide a k -tree cover with maximum weight at most 3λ which is also a true 3-approximation. Now by a binary search in the interval $[0, \sum_{e \in E} w(e)]$, we can find a λ for which our algorithm will give a k -tree cover with bound 3λ and for $\lambda - 1$ the algorithm fails. Thus, for this value of λ we get a 3-approximation factor which completes the proof of Theorem 1. \square

4 A 2.5-Approximation Algorithm for BTC

In this Section we prove Theorem 2. Given an instance of BTC consisting of a graph G and bound λ we use OPT to denote an optimum solution and $k = \text{OPT}$ denote the number of trees in OPT. As before, we can assume we are working with the shortest-path metric completion graph $\tilde{G} = (V, E)$. Our algorithm for this problem is similar to the algorithm for MMkTC but the analysis is different. We delete all the edges with weight greater than $\lambda/4$ in \tilde{G} to obtain graph G' . Let C_1, \dots, C_ℓ be the components of G' whose weight is at most $\lambda/4$, called *light components*, and $C_{\ell+1}, \dots, C_{\ell+h}$ be the components with weight greater than $\lambda/4$ which we refer to as *heavy components*. We define $A(C_i)$, the tree of a light component C_i plus the weight of attaching it to a heavy component as before: it is the weight of minimum spanning tree of C_i , denoted by $W_T(C_i)$, plus the minimum edge weight that connects a node of C_i to a heavy node if such an edge e exists (in \tilde{G}) such that $W_T(C_i) + w(e) \leq \lambda$; otherwise $A(C_i)$ is set to infinity. The proof of the following lemma is identical to that of Lemma 3 with $\frac{3}{2}\lambda$ replaced with $\frac{1}{2}\lambda$.

Lemma 7. *Let $L_s = \{C_{l_1}, \dots, C_{l_s}\}$ be a set of s light-components with bounded $A(C_i)$ values. If $\sum_{1 \leq i \leq s} A(C_{l_i}) + \sum_{\ell+1 \leq i \leq \ell+h} W_T(C_i) \leq x - h\frac{\lambda}{4}$, then we can cover all the nodes in the heavy components and in components of L_s with at most $\lfloor \frac{2x}{\lambda} \rfloor$ trees with maximum tree weight no more than λ .*

We define a graph $H = (L, F)$ formed according to the light components similar to the way we defined it in the MMkTC problem.

Definition 2. For two given parameters a, b , graph H has $\ell + a + b$ nodes: ℓ (regular) nodes v_1, \dots, v_ℓ , where each v_i corresponds to a light component C_i , a dummy nodes called null nodes, and b dummy nodes called heavy nodes. We add an edge with weight zero between two regular v_i and v_j in H if and only if $i \neq j$ and there is an edge e between C_i and C_j in \tilde{G} such that $W_T(C_i) + W_T(C_j) + w(e) \leq \lambda$. Every null node is adjacent to each regular node v_i ($1 \leq i \leq \ell$) with weight zero. Every regular node $v_i \in H$ whose corresponding light component C_i has finite value of $A(C_i)$ is connected to every heavy node in H with an edge of weight $A(C_i)$. There are no other edges in H .

Inputs $G(V, E), \lambda$
Output: A set S containing k' -tree cover with $k' \leq 2.5\text{OPT}$ and maximum tree cost λ .

1. Take \tilde{G} to be the metric completion of G and delete edges with length more than $\frac{\lambda}{4}$ to form graph G' with components $C_1, \dots, C_{\ell+h}$
2. For $a : 0 \rightarrow \ell$
 - (a) For $b : 0 \rightarrow \ell$
 - i. $S_{a,b} \leftarrow \emptyset$
 - ii. Build H according to Definition 2 with a null and b heavy nodes.
 - iii. Find a perfect matching with the minimum cost in H , if there is no such perfect matching continue from Step 2a
 - iv. Attach each light component C_i to its nearest heavy component if v_i is matched to a heavy node
 - v. Decompose all the heavy components and the attached light components as explained in Lemma 7 and add the trees obtained to $S_{a,b}$
 - vi. If a vertex v_i is matched to a null node, add MST of C_i to $S_{a,b}$.
 - vii. For every matching edge between v_i and v_j consider the cheapest edge e between C_i and C_j (in \tilde{G}) and add a minimum spanning trees of $C_i \cup C_j \cup \{e\}$ to $S_{a,b}$.
3. return set $S_{a,b}$ with the minimum number of trees.

Fig. 3. BTC Algorithm

Theorem 2 follows from the following theorem.

Theorem 4. Algorithm BTC (Figure 3) finds a k' -tree cover with maximum tree cost bounded by λ , such that $k' \leq 2.5\text{OPT}$.

Proof. It is easy to check that in all the three steps 2(a)v, 2(a)vi, and 2(a)vii the trees found have weight at most λ : since each is either found using Lemma 7 (Step 2(a)v), or is a MST of a light component (Step 2(a)vi), or is the MST of two light components whose total weight plus the shortest edge connecting them is at most λ (Step 2(a)vii). So it remains to show that for some values of a, b , the total number of trees found is at most 2.5OPT .

First note that if matching M found in Step 2(a)iii assigns nodes v_{i_1}, \dots, v_{i_b} to heavy nodes and has weight W_M then $\sum_{1 \leq i \leq b} A(C_{i_i}) = W_M$. Let W_h denote the total tree weight of heavy components, i.e. $W_h = \sum_{\ell+1 \leq i \leq \ell+h} W_T(C_i)$. Then the number of trees generated using Lemma 7 in Step 2(a)v is at most $\lfloor \frac{2(W_M + W_h + h\lambda/4)}{\lambda} \rfloor$, and the number of trees generated in Steps 2(a)vi and 2(a)iii is exactly $(\ell - b + a)/2$; so we obtain a total of at most $\lfloor \frac{2(W_M + W_h + h\lambda/4)}{\lambda} \rfloor + (\ell - b + a)/2$ trees. We can prove the following lemma (whose proof appears in the full version of this paper):

Lemma 8. *There exist $0 \leq a' \leq n$ and $0 \leq b' \leq n$ such that if H is built with a' null nodes and b' heavy nodes then H has a matching M' such that if Algorithm BTC uses M' then each tree generated has weight at most λ and the total number of trees generated will be at most 2.5OPT .*

This lemma is enough to complete the proof of theorem. Consider an iteration of the algorithm in which $a = a'$ and $b = b'$. Let the minimum perfect matching that the algorithm finds in this iteration be M with weight W_M . Since $W_M \leq W_{M'}$, the total number of trees generated in Step 2(a)v is at most $\lfloor \frac{2(W_M + W_h + h\lambda/4)}{\lambda} \rfloor \leq \lfloor \frac{2(W_{M'} + W_h + h\lambda/4)}{\lambda} \rfloor$. Furthermore, the number of trees generated in Steps 2(a)vi and 2(a)vii is exactly $(\ell - b' + a')/2$, so we obtain a total of at most $\lfloor \frac{2(W_M + W_h + h\lambda/4)}{\lambda} \rfloor + (\ell - b + a)/2$ trees. This together with the fact that $W_M \leq W_{M'}$ and Lemma 8 shows that we get $\leq 2.5\text{OPT}$ trees using M . \square

References

1. Arkin, E.M., Hassin, R., Levin, A.: Approximations for minimum and min-max vehicle routing problems. *Journal of Algorithms* 59, 1–18 (2006)
2. Campbell, A.M., Vandenbussche, D., Hermann, W.: Routing for relief efforts. *Transportation Science* 42, 127–145 (2008)
3. Chaudhuri, K., Godfrey, B., Rao, S., Talwar, K.: Paths, trees, and minimum latency tours. In: *Proceedings of the 44th Annual IEEE Symposium on the Foundations of Computer Science*, pp. 36–45 (2003)
4. Chekuri, C., Kumar, A.: Maximum coverage problem with group budget constraints. In: *Proceedings of APPROX* (2004)
5. Even, G., Garg, N., Könemann, J., Ravi, R., Sinha, A.: Covering graphs using trees and stars. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) *RANDOM 2003 and APPROX 2003*. LNCS, vol. 2764, pp. 24–35. Springer, Heidelberg (2003)
6. Frederickson, G.N., Hecht, M.S., Kim, C.E.: Approximation algorithms for some routing problems. *SIAM J. on Computing* 7, 178–193 (1978)
7. Guttman-Beck, N., Hassin, R.: Approximation algorithms for min-max tree partition. *J. Algorithms* 24, 266–286 (1997)
8. Jothi, R., Raghavachari, B.: Approximating k -traveling repairman problem with repairtimes. *J. of Discrete Algorithms* 5, 293–303 (2007)
9. Li, C.-L., Simchi-Levi, D., Desrochers, M.: On the distance constrained vehicle routing problem. *Oper. Res.* 40, 790–799 (1992)

10. Nagamochi, H.: Approximating the minmax rooted-subtree cover problem. *IEICE Transactions on Fundamentals of Electronics E88-A*, 1335–1338 (2005)
11. Nagarajan, V., Ravi, R.: Approximation algorithms for distance constrained vehicle routing problems (2008)
12. Xu, Z., Wen, Q.: Approximation hardness of min-max tree covers. *Operations Research Letters* 38, 169–173 (2010)
13. Xu, Z., Xu, L.: Approximation algorithms for min-max path cover problems with service handling time. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) *ISAAC 2009*. LNCS, vol. 5878, pp. 383–392. Springer, Heidelberg (2009)

Algorithmic Extensions of Cheeger’s Inequality to Higher Eigenvalues and Partitions

Anand Louis¹, Prasad Raghavendra¹, Prasad Tetali², and Santosh Vempala¹

¹ College of Computing, Georgia Tech, Atlanta
{anand.louis, raghavendra, vempala}@cc.gatech.edu
² School of Mathematics, Georgia Tech, Atlanta
tetali@math.gatech.edu

Abstract. We consider two generalizations of the problem of finding a sparsest cut in a graph. The first is to find a partition of the vertex set into m parts so as to minimize the sparsity of the partition (defined as the ratio of the weight of edges between parts to the total weight of edges incident to the smallest $m - 1$ parts). The second is to find a subset of minimum sparsity that contains at most a $1/m$ fraction of the vertices. Our main results are extensions of Cheeger’s classical inequality to these problems via higher eigenvalues of the graph. In particular, for the sparsest m -partition, we prove that the sparsity is at most $8\sqrt{1 - \lambda_m} \log m$ where λ_m is the m^{th} largest eigenvalue of the normalized adjacency matrix. For sparsest small-set, we bound the sparsity by $O(\sqrt{(1 - \lambda_{m^2}) \log m})$.

1 Introduction

The expansion of a graph is a fundamental and widely studied parameter with many important algorithmic applications [LR99, ARV04, KRV06, She09]. Given an undirected graph $G = (V, E)$, with nonnegative weights $w : E \rightarrow \mathbb{R}_+$ on the edges, the *expansion* of a subset of vertices $S \subset V$ is defined as:

$$\phi_G(S) \stackrel{\text{def}}{=} \frac{w(S, V \setminus S)}{\min\{w(S), w(V \setminus S)\}}$$

where by $w(S)$ we denote the total weight of edges incident to vertices in S and for two subsets S, T , we denote the total weight of edges between them by $w(S, T)$. The degree of a vertex v , denoted by d_v is defined as $d_v \stackrel{\text{def}}{=} \sum_{u \sim v} w(u, v)$. The expansion of the graph is $\phi_G \stackrel{\text{def}}{=} \min_{S \subset V} \phi(S)$.

Cheeger’s inequality connects this combinatorial parameter to graph eigenvalues. Let λ_i denote the i^{th} largest eigenvalue of the normalized adjacency matrix of G , defined as $B \stackrel{\text{def}}{=} D^{-1}A$ where A is the adjacency matrix of G and D is a diagonal matrix with $D(i, i)$ equal to the (weighted) degree of vertex i (each row of B sums to 1).

Theorem 1 (Cheeger’s Inequality ([Al086, AM85])). *Given a graph G , and its row-normalized adjacency matrix B (each row of B sums to 1), let the eigenvalues of B be $1 \geq \lambda_2 \geq \lambda_3 \geq \dots \geq \lambda_n$. Then*

$$2\sqrt{1 - \lambda_2} \geq \phi_G \geq \frac{1 - \lambda_2}{2}.$$

The proof of Cheeger’s inequality is algorithmic and uses the second eigenvector of the normalized adjacency matrix. It gives an efficient algorithm for finding an approximate sparsest cut, i.e., a cut whose sparsity is bounded as in the inequality. Here we consider two natural generalizations of the sparsest cut problem.

1.1 Generalizations of Sparsest Cut

Our first problem is an extension of sparsest cut to partitions with more than two parts.

Sparsest m -partition: Given a weighted undirected graph $G = (V, E)$ and an integer $m > 1$, the sparsity of an m -partition $\mathcal{P} = \{V_1, \dots, V_m\}$ of the vertex set V into m parts is the ratio of the weight of edges between different parts to the sum of the weights of smallest $m - 1$ parts in \mathcal{P} , i.e.,

$$\phi_{G,m}^{sum}(\mathcal{P}) \stackrel{\text{def}}{=} \frac{\sum_{i \neq j} w(V_i, V_j)}{\min_{j \in [m]} w(V \setminus V_j)}$$

The sparsest m -partition has value $\phi_{G,m}^{sum} \stackrel{\text{def}}{=} \min_{\mathcal{P}} \phi_{G,m}^{sum}(\mathcal{P})$.

Variants of such a definition have been considered in the literature. The m -cut problem asks for the minimum weight of edges whose deletion leaves m disjoint parts. Closer to ours is the (α, ϵ) -clustering problem from [KVV04] that asks for a partition where each part has conductance at least α and the total weight of edges removed is minimized.

The second extension we consider is obtained by restricting the size of the set.

Sparsest Small Set: Given a graph $G = (V, E)$ and an integer $m > 0$, the *small-set sparsity* of G is defined as

$$\phi_{G,m}^{small} \stackrel{\text{def}}{=} \min_{S \subset V, w(S) \leq w(V)/m} \frac{w(S, V \setminus S)}{w(S)}$$

The problem is to find a sparsest small set.

The sparsest small set problem has been shown to be closely related to the Unique Games problem (see [RS10, ABS10]). Recently, Arora et. al. ([ABS10]) showed that $\phi_{G,m}^{small} \leq C\sqrt{(1 - \lambda_{m^{100}})} \log_m n$ where C is some absolute constant. They also give a polynomial time algorithm to compute a small set with sparsity satisfying this bound.

1.2 Our Results

For sparsest m -partition, we give the following bound using the m^{th} largest eigenvalue of the normalized adjacency matrix of G .

Theorem 2. For any edge-weighted graph $G = (V, E)$ and integer $|V| \geq m > 0$, there exists an m -partition \mathcal{P} of V such that

$$\phi_{G,m}^{sum}(\mathcal{P}) \leq 8\sqrt{1 - \lambda_m} \log m,$$

where λ_m is the m^{th} largest eigenvalue of the normalized adjacency matrix of G . Moreover, an m -partition with this sparsity bound can be computed in polynomial time.

The above result is a generalization of the upper bound in Cheeger’s inequality (where $m = 2$). Our proof is based on a recursive partitioning algorithm that might be of independent interest. We remark that the dependence on m is necessary and cannot be improved to something smaller than $\sqrt{\log m}$. Moreover, notice that the lower bound of $\Omega(1 - \lambda_2)$ in Cheeger’s inequality cannot be strengthened for $m > 2$: Consider the graph G constructed by taking $m - 1$ cliques C_1, C_2, \dots, C_{m-1} each on $(n - 1)/(m - 1)$ vertices. Let v be the remaining vertex. Let C_1, \dots, C_{m-1} be connected to v by a single edge. Now, G will have $m - 1$ eigenvalues close to 1 because of the $m - 1$ cuts $(\{v\}, C_i)$ for $i \in [m - 1]$, but the m^{th} eigenvalue will be close to 0, as any other cut which is not a linear combination of these $m - 1$ cuts will have to cut through one of the cliques. Therefore, λ_m must be a constant smaller than $1/2$. But $\phi_{G,m}^{sum} = (m - 1)/((m - 2)(n/m)^2) \approx m^2/n^2$. Thus, $1 - \lambda_m \gg \phi_{G,m}^{sum}$ for small enough values of m .

For the sparsest small-set problem, we present the following bound.

Theorem 3. Given a graph $G = (V, E)$ and an integer $|V| > m > 1$, there exists a non-empty subset $S \subset V$ such that $|S| \leq \frac{2|V|}{m}$ and

$$\phi(S) \leq C\sqrt{(1 - \lambda_{m^2}) \log m}$$

where C is a fixed constant. Moreover, such a set can be computed in polynomial time.

The result is a consequence of the rounding technique of [RST10a] and a relation between eigenvalues and the SDP relaxation observed by [Ste10].

A lower bound of $(1 - \lambda_2)/2$ for $\phi_{G,m}^{small}$ follows from Cheeger’s inequality. Furthermore, it is easy to see that this bound cannot be improved in general. Specifically, consider the graph G constructed by adding an edge between a copy of $K_{\lfloor n/m \rfloor}$ and a copy of $K_{\lfloor n(1-1/m) \rfloor}$. In this graph, $\phi_{G,m}^{small} \approx 1/(n/m)^2 = m^2/n^2$, whereas G has only 1 eigenvalue close to 1 and $\lambda_m \approx 0$ for $m > 3$.

We believe that there is room for improvement in both our theorems, and especially for the sparsest small-set, we believe that the dependence should be on a lower eigenvalue (m instead of m^2). We make the following conjecture:

Conjecture 1. There is a fixed constant C such that for any graph $G = (V, E)$ and any integer $|V| > m > 1$,

$$\phi_{G,m}^{sum}, \phi_{G,m}^{small} \leq C\sqrt{(1 - \lambda_m) \log m},$$

where λ_m is the m^{th} largest eigenvalue of the normalized adjacency matrix of G .

The bounds in this conjecture are matched by the *Gaussian graphs*. For a constant $\epsilon \in (-1, 1)$, let $N_{k,\epsilon}$ denote the infinite graph over \mathbb{R}^k where the weight of an edge (x, y) is the probability that two standard Gaussian random vectors X, Y with correlation¹ ϵ equal x and y respectively. The first k eigenvalues of $N_{k,\epsilon}$ are at least $1 - \epsilon$ (see [RST10b]). The following lemma bounds the expansion of small sets in $N_{k,\epsilon}$.

Lemma 1 ([Bor85, RST10b]). *For $m < k$ we have*

$$\phi_{N_{k,\epsilon},m}^{small} \geq \Omega(\sqrt{\epsilon \log m})$$

Therefore, for any value of $m < k$, $N_{k,\epsilon}$ has $\phi_{N_{k,\epsilon},m}^{small} \geq \Omega(\sqrt{(1 - \lambda_m) \log m})$.

2 Monotonicity of Eigenvalues

In this section we collect some useful properties about the behavior of eigenvalues upon deleting edges and merging vertices.

Lemma 2 (Weyl’s Inequality). *Given a Hermitian matrix B with eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$, and a positive semidefinite matrix E , if $\lambda'_1 \geq \lambda'_2 \geq \dots \geq \lambda'_n$ denote the eigenvalues of $B' \stackrel{\text{def}}{=} B + E$, then $\lambda'_i \geq \lambda_i$.*

Proof. The i^{th} eigenvalue of B' can be written as

$$\begin{aligned} \lambda'_i &= \max_{S:\text{rank}(S)=i} \min_{x \in S} \frac{x^T B' x}{x^T x} = \max_{S:\text{rank}(S)=i} \min_{x \in S} \frac{x^T B x + x^T E x}{x^T x} \\ &\geq \max_{S:\text{rank}(S)=i} \min_{x \in S} \frac{x^T B x}{x^T x} = \lambda_i. \end{aligned}$$

Lemma 3. *Let B be the row normalized matrix of the graph G . Let F be any subset of edges of G . For every pair $(i, j) \in F$, remove the edge (i, j) from G and add self loops at i and j to get the graph G' . Let B' be the row-normalized matrix of G' . Let the eigenvalues of B be $1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and let the eigenvalues of B' be $1, \lambda'_2, \lambda'_3, \lambda'_4 \geq \dots \geq \lambda'_n$. Then $\lambda'_i \geq \lambda_i \forall i \in [n]$.*

Proof. Let $D^{\frac{1}{2}}$ be the diagonal matrix whose $(i, i)^{\text{th}}$ entry is $\sqrt{d_i}$. Observe that $DB = B^T D$. Therefore $Q \stackrel{\text{def}}{=} D^{\frac{1}{2}} B D^{-\frac{1}{2}}$ is a symmetric matrix where Moreover, the eigenvalues of Q and B are the same: if ν is an eigenvector of Q with eigenvalue λ , i.e. $D^{\frac{1}{2}} B D^{-\frac{1}{2}} \nu = \lambda \nu$, then $B(D^{-\frac{1}{2}} \nu) = \lambda(D^{-\frac{1}{2}} \nu)$.

Hence, the eigenvalues of Q are $1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and the eigenvalues of $Q' \stackrel{\text{def}}{=} D^{\frac{1}{2}} B' D^{-\frac{1}{2}}$ are $1 \geq \lambda'_2 \geq \lambda'_3 \geq \lambda'_4 \geq \dots \geq \lambda'_n$.

$C \stackrel{\text{def}}{=} D^{\frac{1}{2}} (B' - B) D^{-\frac{1}{2}}$ is the matrix corresponding to the edge subset F . It has non-negative entries along its diagonal and non-positive entries elsewhere

¹ ϵ correlated Gaussians can be constructed as follows : $X \sim N(0, 1)^k$ and $Y \sim (1 - \epsilon)X + \sqrt{2\epsilon - \epsilon^2}Z$ where $Z \sim N(0, 1)^k$.

such that $\forall i \ c_{ii} = -\sum_{j \neq i} c_{ij}$. C is symmetric and positive semi-definite as for any vector x of appropriate dimension, we have $x^T C x = \sum_{ij} c_{ij} x_i x_j = -\frac{1}{2} \sum_{i \neq j} c_{ij} (x_i - x_j)^2 \geq 0$.

Using Lemma [2](#), we get that $\lambda'_i \geq \lambda_i \ \forall i \in [n]$.

Lemma 4. *Let B be the row normalized matrix of the graph G . Let S be a non-empty set of vertices of G . Let G' be the graph obtained from G by shrinking [2](#) S to a single vertex. Let B' be the row normalized adjacency matrix of G' . Let the eigenvalues of B be $1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and let the eigenvalues of B' be $1, \lambda'_2, \lambda'_3, \lambda'_4 \geq \dots \geq \lambda'_{n-|S|+1}$. Then $\lambda_i \geq \lambda'_i$ for $1 \leq i \leq n - |S| + 1$.*

Proof. Let $D^{\frac{1}{2}}$ be the diagonal matrix whose $(i, i)^{th}$ entry is $\sqrt{d_i}$. Observe that $DB = B^T D$. Therefore $Q \stackrel{\text{def}}{=} D^{\frac{1}{2}} B D^{-\frac{1}{2}}$ is a symmetric matrix where Moreover, the eigenvalues of Q and B are the same: if ν is an eigenvector of Q with eigenvalue λ , i.e. $D^{\frac{1}{2}} B D^{-\frac{1}{2}} \nu = \lambda \nu$, then $B(D^{-\frac{1}{2}} \nu) = \lambda(D^{-\frac{1}{2}} \nu)$. The i^{th} eigenvalue of B can be written as $\lambda_i = \max_{S: \text{rank}(S)=i} \min_{x \in S} \frac{x^T B x}{x^T x}$ and hence

$$\begin{aligned} \lambda_i &= \max_{S: \text{rank}(S)=i} \min_{x \in S} 1 - \frac{x^T D^{\frac{1}{2}} (I - B) D^{-\frac{1}{2}} x}{x^T x} \\ &= \max_{S: \text{rank}(S)=i} \min_{x \in S} 1 - \frac{\sum_i \sum_{j>i} d_i b_{ij} (x_i - x_j)^2}{\sum_i d_i x_i^2} \end{aligned}$$

Let $s = |S|$. Let v_1, v_2, \dots, v_n be the vertices of G , let $S = \{v_{n-s+1}, \dots, v_n\}$ and $v_1, v_2, \dots, v_{n-s}, v'_{n-s+1}$ be the vertices of G' where v'_{n-s+1} is the vertex obtained by shrinking S to a single vertex. If d'_i denotes the degree of i^{th} vertex in G' then $d'_i = d_i$ for $1 \leq i \leq n - s$ and $d'_{n-s+1} = \sum_{i \in S} d_i$.

Let T^k be a variable denoting a subspace of \mathbb{R}^k .

$$\begin{aligned} \lambda'_i &= \max_{T^{n-s+1}: \text{rank}(T^{n-s+1})=i} \min_{x \in T^{n-s+1}} 1 - \frac{\sum_{i=1}^{n-s+1} \sum_{j>i} d'_i b'_{ij} (x_i - x_j)^2}{\sum_i d'_i x_i^2} \\ &= \max_{T^{n-s+1}: \text{rank}(T^{n-s+1})=i} \min_{x \in T^{n-s+1}} 1 - \frac{\sum_{i=1}^{n-s} \sum_{j>i} d_i b_{ij} (x_i - x_j)^2}{\sum_{i=1}^{n-s} d_i x_i^2 + (\sum_{i=n-s+1}^n d_i) x_{n-s+1}^2} \\ &\leq \max_{T^n: \text{rank}(T^n)=i} \min_{x \in T^n} 1 - \frac{\sum_{i=1}^n \sum_{j>i} d_i b_{ij} (x_i - x_j)^2}{\sum_i d_i x_i^2} \\ &= \lambda_i \end{aligned}$$

3 Sparse m -Partition

Let A denote the adjacency matrix of the graph. We normalize A by scaling the rows so that the row sums are equal to one. Let B denote this row-normalized matrix.

² A vertex set S is said to be shrunk to a vertex v_S , if all the vertices in S are removed from G and in its place a new vertex v_S is added. All the edges in $E(S, \bar{S})$ are now incident on v_S and all the internal edges in S now become self loops on v_S .

We propose the following recursive algorithm for finding an m -partitioning of G . Use the second eigenvector of G to find a sparse cut (C, \bar{C}) . Let $G' = (V, E')$ be the graph obtained by removing the edges in the cut (C, \bar{C}) from G , i.e. $E' = E \setminus E(C, \bar{C})$. We obtain the matrix B' as follows: For all edges $(i, j) \in E'$, $b'_{ij} = b_{ij}$. For all other i, j such that $i \neq j$, $b'_{ij} = 0$. For all i , $b'_{ii} = 1 - \sum_{j \neq i} b'_{ij}$. Note that B' corresponds to the row-normalized adjacency matrix of G' , if $\forall (i, j) \in E(C, \bar{C})$ we add self loops at vertex i and vertex j in G' . The matrix B' is block-diagonal with two blocks for the two components of G' . The spectrum of B' (eigenvalues, eigenvectors) is the union of the spectra of the two blocks. The first two eigenvalues of B' are now 1 and we use the third largest eigenvector of G' to find a sparse cut in G' . This is the second eigenvector in one of the two blocks and partitions that block. We repeat the above process till we have at least m connected components. This can be viewed as a recursive algorithm, where at each step one of the current components is partitioned into two; the component partitioned is the one that has the highest second eigenvalue among all the current components. The precise algorithm appears in Figure 1.

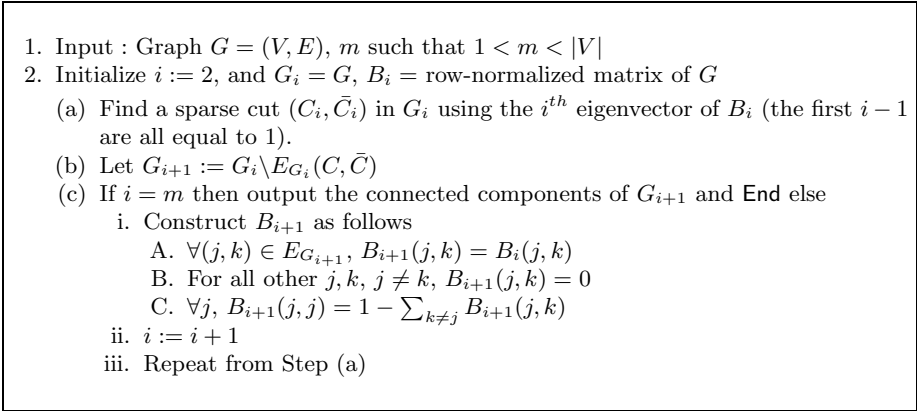


Fig. 1. The Recursive Algorithm

We now analyze the algorithm. Our analysis will also be a proof of Theorem 2.

The matrix B_i for $i > 2$ is not the row-normalized matrix of G_i , but can be viewed as a row normalized matrix of G_i with a self loop on vertices i and j for each edge $(i, j) \in E_{G_i}(C_i, \bar{C}_i)$. The next theorem is a generalization of Cheeger’s inequality to weighted graphs, which relates the eigenvalues of B to the sparsity of G .

Theorem 4 ([KVV04]). *Suppose B is a $N * N$ matrix with nonnegative entries with each row sum equal to 1 and suppose there are positive real numbers $\pi_1, \pi_2, \dots, \pi_N$ summing to 1 such that $\pi_i b_{ij} = \pi_j b_{ji} \forall i, j$. If v is the right eigenvector of B corresponding to the 2^{nd} largest eigenvalue λ_2 and i_1, i_2, \dots, i_N is an ordering of $1, 2, \dots, N$ such that $v_{i_1} \geq v_{i_2} \geq \dots \geq v_{i_N}$, then*

$$2\sqrt{1 - \lambda_2} \geq \min_{l:1 \leq l \leq N} \frac{\sum_{1 \leq u \leq l; l+1 \leq v \leq N} \pi_{i_u} b_{i_u i_v}}{\min\{\sum_{1 \leq u \leq l} \pi_{i_u}, \sum_{l+1 \leq v \leq N} \pi_{i_v}\}} \geq \frac{1 - \lambda_2}{2}$$

Lemma 2 shows that the eigenvalues of B_i are monotonically nondecreasing with i . This will show that $\phi_{G_i}(C_i) \leq 2\sqrt{1 - \lambda_m}$.

We can now prove the main theorem.

Proof (of Theorem 2). Let \mathcal{P} be the set of partitions output by the algorithm and let $S(\mathcal{P})$ denote the sum of weights of the smallest $m - 1$ pieces in \mathcal{P} . Note that we need only the smaller side of a cut to bound the size of the cut : $|E_G(S, \bar{S})| \leq \phi_G|S|$. We define the notion of a **cut-tree** $T = (V(T), E(T))$ as follows: $V(T) = \{V\} \cup \{C_i | i \in [m]\}$ (For any cut (C_i, \bar{C}_i) we denote the part with the smaller weight by C_i and the part with the larger weight by \bar{C}_i . We break ties arbitrarily). We put an edge between $S_1, S_2 \in V(T)$ if $\exists S \in V(T)$ such that $S_1 \subsetneq S \subsetneq S_2$ or $S_2 \subsetneq S \subsetneq S_1$, (one can view S_1 as a ‘top level’ cut of S_2 in the former case).

Clearly, T is connected and is a tree. We call V the root of T . We define the *level* of a node in T to be its depth from the root. We denote the level of node $S \in V(T)$ by $L(S)$. The root is defined to be at level 0. Observe that $S_1 \in V(T)$ is a descendant of $S_2 \in V(T)$ if and only if $S_1 \subsetneq S_2$. Now $E(\mathcal{P}) = \cup_i E_{G_i}(C_i, \bar{C}_i) = \cup_i \cup_{j:L(C_j)=i} E_{G_j}(C_j, \bar{C}_j)$. We make the following claim.

Claim.

$$w(\cup_{j:L(C_j)=i} E(C_j, \bar{C}_j)) \leq 2\sqrt{1 - \lambda_m} S(\mathcal{P})$$

Proof. By definition of level, if $L(C_i) = L(C_j)$, $i \neq j$, then the node corresponding to C_i in the T can not be an ancestor or a descendant of the node corresponding to C_j . Hence, $C_i \cap C_j = \phi$. Therefore, all the sets of vertices in level i are pairwise disjoint. Using Cheeger’s inequality we get that $E(C_j, \bar{C}_j) \leq 2\sqrt{1 - \lambda_m} w(C_j)$. Therefore

$$w(\cup_{j:L(C_j)=i} E(C_j, \bar{C}_j)) \leq 2\sqrt{1 - \lambda_m} \sum_{j:L(C_j)=i} w(C_j) \leq 2\sqrt{1 - \lambda_m} S(\mathcal{P})$$

This claim implies that $\phi(\mathcal{P}) \leq 2\sqrt{1 - \lambda_m} \text{height}(T)$.

The height of T might be as much as m . But we will show that we can assume $\text{height}(T)$ to be $\log m$. For any path in the tree $v_1, v_2, \dots, v_{k-1}, v_k$ such that $\text{deg}(v_1) > 2$, $\text{deg}(v_i) = 2$ (i.e. v_i has only 1 child in T) for $1 < i < k$, we have $w(C_{v_{i+1}}) \leq w(C_{v_i})/2$, as v_{i+1} being a child of v_i in the T implies that $C_{v_{i+1}}$ was obtained by cutting C_{v_i} using its second eigenvector. Thus $\sum_{i=2}^k w(C_{v_i}) \leq w(C_{v_1})$. Hence we can modify the T as follows : make the nodes v_3, \dots, v_k children of v_2 . The nodes v_3, \dots, v_{k-1} now become leaves whereas the subtree rooted at v_k remains unchanged. We also assign the level of each node as its new distance from the root. In this process we might have destroyed the property that a node is obtained from by cutting its parent, but we have the property that $w(\cup_{j:L(C_j)=i} E(C_j, \bar{C}_j)) \leq 4\sqrt{1 - \lambda_m} S(\mathcal{P}) \forall i$.

Claim.

$$w(\cup_{j:L(C_j)=i} E(C_j, \bar{C}_j)) \leq 4\sqrt{1 - \lambda_m} S(\mathcal{P})$$

Proof. If the nodes in level i are unchanged by this process, then the claim clearly holds. If any node v_j in level i moved to a higher level, then the nodes replacing v_j in level i would be descendants of v_j in the original T and hence would have weight at most $w(C_{v_j})$. If the descendants of some node v_j got added to level i , then, as seen above, their combined weight would be at most $w(C_{v_j})$. Hence,

$$w(\cup_{j:L(C_j)=i} E(C_j, \bar{C}_j)) \leq 2(2\sqrt{1 - \lambda_m} \sum_{j:L(C_j)=i} w(C_j)) \leq 4\sqrt{1 - \lambda_m} S(\mathcal{P})$$

Repeating this process we can ensure that no two adjacent nodes in the T have degree 2. Hence, there are at most $\log m$ vertices along any path starting from the root which have exactly one child. Thus the height of the new cut-tree is at most $2 \log m$. Thus $E((\mathcal{P})) \leq 8\sqrt{1 - \lambda_m} \log m S(\mathcal{P})$ and hence $\phi_{G,m}^{sum} \leq \frac{E((\mathcal{P}))}{S(\mathcal{P})} \leq 8\sqrt{1 - \lambda_m} \log m$.

4 Finding Sparsest Small Sets

Given an integer m and an undirected graph $G = (V, E)$, we wish to find the set $S \subset V$ of size at most $|V|/m$ and having minimum expansion. This is equivalent to finding the vector $x \in \{0, 1\}^{|V|}$ which minimizes $\frac{\sum_{i \sim j} w(i, j)(x(i) - x(j))^2}{\sum_i d_i x(i)^2}$ and has at most $|V|/m$ non-zero entries. Ignoring the sparsity constraint, the minimization is equivalent to minimizing $\frac{\sum_{i \sim j} w(i, j) \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2}$ over all collections of vectors $\{v_i | i \in [n]\}$. The challenge is to deal with the sparsity constraint. Since any $x \in \{0, 1\}^{|V|}$ having at most $|V|/m$ non-zero entries satisfies $\sum_{i,j} x(i)x(j) \leq n^2/m^2$ we can relax the sparsity constraint to $\sum_{i,j} \langle v_i, v_j \rangle \leq n^2/m^2$ while maintaining $\sum_i \|v_i\|^2 = n$. This convex relaxation of the problem is shown in Figure 2.

$$\min \frac{\sum_{i,j} w(i, j) \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2}$$

$$\sum_{i,j} \langle v_i^T, v_j \rangle \leq \frac{n^2}{m^2}$$

$$\sum_i \|v_i\|^2 = n$$

Fig. 2. A convex relaxation for sparsest small set

It was pointed out to us by [Ste10] that eigenvectors of the graph form a feasible solution to this convex relaxation. Here we present a proof of the same.

Let w_1, w_2, \dots, w_n denote the eigenvectors of DBD^{-1} and let $1 \geq \lambda_2 \geq \lambda_3 \geq \dots \geq \lambda_n$ be the respective eigenvalues. Let F be the $m^2 * n$ dimensional matrix which has w_1, w_2, \dots, w_{m^2} as its row vectors, i.e. $F = [w_1 \ w_2 \ \dots \ w_{m^2}]^T$. Let f_1, f_2, \dots, f_n be the columns of F . We define $v_i \stackrel{\text{def}}{=} (\sqrt{\frac{n}{m^2}} f_i)$. We will show that $\{v_i | i \in [n]\}$ forms a feasible solution for the convex program, and that the cost of the solution is bounded by $1 - \lambda_{m^2}$.

Lemma 5. *The vectors $v_i, i \in [n]$ satisfy $\sum_{i,j} \langle v_i, v_j \rangle^2 \leq \frac{n^2}{m^2}$.*

Proof.

$$\begin{aligned} \sum_{i,j} \langle v_i, v_j \rangle^2 &= \frac{n^2}{m^4} \sum_{i,j} \left(\sum_t f_{it} f_{jt} \right)^2 = \frac{n^2}{m^4} \sum_{i,j} \sum_{t_1, t_2} f_{it_1} f_{jt_1} f_{it_2} f_{jt_2} \\ &= \frac{n^2}{m^4} \sum_{t_1, t_2} \langle w_{t_1} \otimes w_{t_1}, w_{t_2} \otimes w_{t_2} \rangle = \frac{n^2}{m^4} \sum_{t_1, t_2} \langle w_{t_1}, w_{t_2} \rangle^2 = \frac{n^2}{m^2}. \end{aligned}$$

Lemma 6. $\sum_i \|v_i\|^2 = n$

Proof.

$$\begin{aligned} \sum_i \langle v_i, v_i \rangle &= \frac{n}{m^2} \sum_i \langle f_i, f_i \rangle = \frac{n}{m^2} \sum_i \left(\sum_t f_{it}^2 \right) \\ &= \frac{n}{m^2} \sum_t \|w_t\|_2^2 = \frac{n}{m^2} m^2 = n. \end{aligned}$$

Lemma 7. $\frac{\sum_{i,j} w_{ij} \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2} \leq 1 - \lambda_{m^2}$

Proof.

$$\frac{\sum_{i,j} w_{ij} \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2} = \frac{\sum_l \sum_{i,j} w_{ij} \|v_{li} - v_{lj}\|^2}{\sum_l \sum_i d_i \|v_{li}\|^2} \leq \max_l \frac{\sum_{i,j} w_{ij} \|v_{li} - v_{lj}\|^2}{\sum_i d_i \|v_{li}\|^2} \leq 1 - \lambda_{m^2}.$$

Lemmas 5 and 6 show that the $\{v_i | i \in [n]\}$ form a feasible solution to the convex program and Lemma 7 shows that the cost of this solution is at most $\sqrt{1 - \lambda_{m^2}}$.

We use the rounding scheme of [RST10a] to round this solution of the convex program to get a set S of size $2n/m$ and $\phi(S) \leq \mathcal{O}(\sqrt{(1 - \lambda_{m^2}) \log m})$. We give the rounding procedure in Figure 3.

For any f_i and f_j defined as above, their inner product is defined as $\langle f_i, f_j \rangle \stackrel{\text{def}}{=} \int_{-\infty}^{\infty} f_i(x) f_j(x) dx$. The following lemma is a slightly modified version of a similar lemma in [RST10a] to suit our requirements. For completeness we give the proof in Appendix A.

Lemma 8. 1. $\frac{\sum_{i,j} w_{i,j} \|f_i - f_j\|^2}{\sum_i d_i \|f_i\|^2} \leq \frac{\sum_{i,j} w_{i,j} \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2}$

1. For each $i \in [n]$ define functions $f_i \stackrel{\text{def}}{=} \|v_i\| \sqrt{\Phi(x - v_i^*)}$ where $\Phi(x)$ is probability density function of gaussian with mean 0 and variance $1/\sqrt{\log m}$ and v_i^* denotes the unit vector along the direction of v_i .
2. Sample $t \in \mathcal{N}(0, 1)^{m^2}$.
3. Compute $\theta = 2m * \sum_i f_i(t)$ and define $x_i \stackrel{\text{def}}{=} \max\{f_i(t) - \theta, 0\}$ for each $i \in [n]$.
4. Do a Cheeger rounding on $X \stackrel{\text{def}}{=} [x_1 x_2 \dots x_n]^T$.

Fig. 3. The Rounding Algorithm

$$2. \sum_{i,j} \langle f_i, f_j \rangle \leq 2n/m$$

Lemma 9 ([RST10a]).

1. $E(\text{support}(X)) \leq 2n/m$
2. $\frac{\sum_{i,j} w_{i,j} (x_i - x_j)^2}{\sum_i d_i x_i^2} \leq \frac{\sum_{i,j} w_{i,j} \|f_i - f_j\|^2}{\sum_i d_i \|f_i\|^2}$

Proof (of Theorem 3).

Lemma 8 shows that $\{f_i | i \in [n]\}$ satisfy a stronger sparsity condition than the one in Figure 2 and the value of the objective function of the convex program on $\{f_i | i \in [n]\}$ is at most $\mathcal{O}(\log m)$ times the value of the objective function on $\{v_i | i \in [n]\}$.

Lemma 9 shows that X has at most $2n/m$ non-zero entries and together with Lemma 8 implies that cost of the objective function of the convex program on X is at most $\mathcal{O}(\log m)$ times the cost of the objective function on $\{v_i | i \in [n]\}$.

Performing a Cheeger rounding on X will yield a set of size at most $2n/m$ and expansion $\mathcal{O}(\sqrt{\log m \frac{\sum_{i,j} w_{i,j} \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2}}) \leq \mathcal{O}(\sqrt{(1 - \lambda_{m^2}) \log m})$, where the inequality follows from Lemma 7.

Thus we have Theorem 3.

Acknowledgements. We are grateful to David Steurer who first suggested that eigenvectors could be used to construct a feasible solution to the convex program in Figure 2, and for other kind comments on this paper. This work was supported in part by a student fellowship at the Algorithms and Randomness Center (ARC) and by NSF awards AF-0915903 and AF-0910584.

References

[ABS10] Arora, S., Barak, B., Steurer, D.: Subexponential algorithms for unique games and related problems. In: FOCS (2010)

[Alo86] Alon, N.: Eigenvalues and expanders. *Combinatorica* 6(2), 83–96 (1986)

[AM85] Alon, N., Milman, V.D.: λ_1 , isoperimetric inequalities for graphs, and superconcentrators. *J. Comb. Theory, Ser. B* 38(1), 73–88 (1985)

[ARV04] Arora, S., Rao, S., Vazirani, U.V.: Expander flows, geometric embeddings and graph partitioning. In: Babai, L. (ed.) STOC, pp. 222–231. ACM, New York (2004)

[Bor85] Borell, C.: Geometric bounds on the ornstein-uhlenbeck velocity process. *Probability Theory and Related Fields* 70, 1–13 (1985), doi:10.1007/BF00532234

[KRV06] Khandekar, R., Rao, S., Vazirani, U.V.: Graph partitioning using single commodity flows. In: Kleinberg, J.M. (ed.) STOC, pp. 385–390. ACM, New York (2006)

[KVV04] Kannan, R., Vempala, S., Vetta, A.: On clusterings: Good, bad and spectral. *J. ACM* 51(3), 497–515 (2004)

[LR99] Leighton, F.T., Rao, S.: Multicommodity max-flow min-cut theorems and their use in designing approximation algorithms. *J. ACM* 46(6), 787–832 (1999)

[RS10] Raghavendra, P., Steurer, D.: Graph expansion and the unique games conjecture. In: Schulman, L.J. (ed.) STOC, pp. 755–764. ACM, New York (2010)

[RST10a] Raghavendra, P., Steurer, D., Tetali, P.: Approximations for the isoperimetric and spectral profile of graphs and related parameters. In: STOC, pp. 631–640 (2010)

[RST10b] Raghavendra, P., Steurer, D., Tulsiani, M.: Reductions between expansion problems (2010) (manuscript)

[She09] Sherman, J.: Breaking the multicommodity flow barrier for $\mathcal{O}(\sqrt{\log n})$ -approximations to sparsest cut. In: FOCS, pp. 363–372, IEEE Computer Society, Los Alamitos (2009)

[Ste10] Steurer, D.: Personal communication (2010)

A Proof of Lemma 8

We first state some known facts about Gaussians.

Fact 5. *Let $\Phi_\sigma(u)$ be the probability function of a multi-dimensional gaussian centered at $u \in \mathbb{R}^n$ and having variance σ in each coordinate. Let δ^n denote the standard Lebesgue measure on \mathbb{R}^n . Then*

$$\int \sqrt{\Phi_\sigma(u)\Phi_\sigma(v)}d\delta^n = e^{-\|u-v\|^2/8\sigma^2}$$

The following fact shows that in order for a mapping to preserve distances it is enough to preserve lengths and distances of unit vectors.

Fact 6. *For any two vectors $u, v \in \mathbb{R}^n$, we have*

$$\|u - v\|^2 = (\|u\| - \|v\|)^2 + \|u\|\|v\|\|u^* - v^*\|^2$$

We state Lemma 8 again.

Lemma 10. 1. $\frac{\sum_{i,j} w_{i,j} \|f_i - f_j\|^2}{\sum_i d_i \|f_i\|^2} \leq \frac{\sum_{i,j} w_{i,j} \|v_i - v_j\|^2}{\sum_i d_i \|v_i\|^2}$
 2. $\sum_{i,j} \langle f_i, f_j \rangle \leq 2n/m$

Proof. 1. Since $\|v_i\| = \|f_i\| \forall i \in [n]$ it suffices to show that we have $\|f_i - f_j\|^2 \leq \mathcal{O}(\log m \|v_i - v_j\|^2) \forall i, j \in [n]$. Using Fact 5,

$$\|f_i^* - f_j^*\| = 2 - 2e^{-\log m \|v_i^* - v_j^*\|^2/8} \leq \log m \|v_i^* - v_j^*\|^2/4$$

Now, using Fact 6

$$\begin{aligned} \|f_i - f_j\|^2 &= (\|v_i\| - \|v_j\|)^2 + \|v_i\| \|v_j\| \|f_i^* - f_j^*\|^2 \\ &= (\|v_i\| - \|v_j\|)^2 + \log m \|v_i\| \|v_j\| \|v_i^* - v_j^*\|^2/4 \leq \log m \|v_i - v_j\|^2/4. \end{aligned}$$

The last inequality uses Fact 6 again. This proves the first part.

2. For the second part, we use the fact that $e^{cx} \leq 1 - (1 - e^c)x$.

$$\begin{aligned} \langle f_i^*, f_j^* \rangle &= e^{-\log m (1 - \langle v_i^*, v_j^* \rangle)} \leq e^{-\log m (1 - |\langle v_i^*, v_j^* \rangle|)} \\ &\leq 1 - (1 - e^{-\log m})(1 - |\langle v_i^*, v_j^* \rangle|) \leq e^{-\log m} + |\langle v_i^*, v_j^* \rangle|. \end{aligned}$$

Now,

$$\sum_{i,j} \langle f_i, f_j \rangle \leq \sum_{i,j} \|v_i\| \|v_j\| (1/m + \langle v_i^*, v_j^* \rangle) = 1/m (\sum_i \|f_i\|)^2 + \sum_{i,j} |\langle v_i, v_j \rangle|$$

By Jensen’s inequality, the first term contributes not more than $\sum_i \|f_i\|^2$. The constraints in the convex program imply that $\sum_{i,j} |\langle v_j, v_j \rangle| \leq n/m$. Putting these together we get the second part of the lemma.

Nearly Optimal NP-Hardness of Vertex Cover on k -Uniform k -Partite Hypergraphs*

Sushant Sachdeva and Rishi Saket

Department of Computer Science
Princeton University, Princeton, NJ 08540, USA

Abstract. We study the problem of computing the minimum vertex cover on k -uniform k -partite hypergraphs when the k -partition is given. On bipartite graphs ($k = 2$), the minimum vertex cover can be computed in polynomial time. For general k , the problem was studied by Lovász [23], who gave a $\frac{k}{2}$ -approximation based on the standard LP relaxation. Subsequent work by Aharoni, Holzman and Krivelevich [1] showed a tight integrality gap of $(\frac{k}{2} - o(1))$ for the LP relaxation. While this problem was known to be NP-hard for $k \geq 3$, the first non-trivial NP-hardness of approximation factor of $\frac{k}{4} - \varepsilon$ was shown in a recent work by Guruswami and Saket [13]. They also showed that assuming Khot’s Unique Games Conjecture yields a $\frac{k}{2} - \varepsilon$ inapproximability for this problem, implying the optimality of Lovász’s result.

In this work, we show that this problem is NP-hard to approximate within $\frac{k}{2} - 1 + \frac{1}{2k} - \varepsilon$. This hardness factor is off from the optimal by an additive constant of at most 1 for $k \geq 4$. Our reduction relies on the *Multi-Layered PCP* of [8] and uses a gadget – based on biased Long Codes – adapted from the LP integrality gap of [1]. The nature of our reduction requires the analysis of several Long Codes with different biases, for which we prove structural properties of the so called *cross-intersecting* collections of set families – variants of which have been studied in extremal set theory.

1 Introduction

A k -uniform hypergraph $G = (V, E)$ consists of a set of vertices V and a collection of hyperedges E such that each hyperedge contains exactly k vertices. A vertex cover for G is a subset of vertices $\mathcal{V} \subseteq V$ such that every hyperedge e contains at least one vertex from \mathcal{V} , i.e., $e \cap \mathcal{V} \neq \emptyset$. Equivalently, a vertex cover is a hitting set for the collection of hyperedges E . The complement of a vertex cover is called an *Independent Set*, which is a subset of vertices \mathcal{I} such that no hyperedge $e \in E$ is contained inside \mathcal{I} , i.e., $e \not\subseteq \mathcal{I}$.

The k -HYPVC problem is to compute the minimum vertex cover in a k -uniform hypergraph G . It is an extremely well studied combinatorial optimization problem, especially on graphs ($k = 2$), and is known to be NP-hard. Indeed, the minimum vertex cover problem on graphs was one of Karp’s original 21 NP-complete problems [19]. On the other hand, the simple greedy algorithm that picks a maximal collection of disjoint hyperedges and includes all vertices in the edges in the vertex cover gives a k -approximation, which is also obtained by the standard LP relaxation of the problem.

* Research supported in part by NSF grants CCF-0832797, 0830673, and 0528414.

The best algorithms known today achieve only a marginally better approximation factor of $(1 - o(1))k$ [18,15].

On the intractability side, there have been several results. For the case $k = 2$, Dinur and Safra [9] obtained an NP-hardness of approximation factor of 1.36, improving on a $\frac{7}{6} - \varepsilon$ hardness by Håstad [14]. For general k a sequence of successive works yielded improved NP-hardness factors: $\Omega(k^{1/19})$ by Trevisan [27]; $\Omega(k^{1-\varepsilon})$ by Holmerin [16]; $k - 3 - \varepsilon$ by Dinur, Guruswami and Khot [7]; and the currently best $k - 1 - \varepsilon$ due to Dinur, Guruswami, Khot and Regev [8]. In [8], the authors build upon [7] and the work of Dinur and Safra [9]. Moreover, assuming Khot's Unique Games Conjecture (UGC) [20], Khot and Regev [21] showed an essentially optimal $k - \varepsilon$ inapproximability. This result was further strengthened in different directions by Austrin, Khot and Safra [5] and by Bansal and Khot [6].

Vertex Cover on k -uniform k -partite Hypergraphs

In this paper, we study the problem of finding the minimum vertex cover on k -partite k -uniform hypergraphs, when the underlying partition is given. We denote this problem as k -HYPVC-PARTITE. This is an interesting problem in itself and its variants have been studied for applications related to databases such as distributed data mining [10], schema mapping discovery [11] and optimization of finite automata [17]. On bipartite graphs ($k = 2$), by König's Theorem computing the minimum vertex cover is equivalent to computing the maximum matching which can be done efficiently. For general k , the problem was studied by Lovász who, in his doctoral thesis [23], proved the following upper bound.

Theorem 1 (Lovász [23]). *For every k -partite k -uniform hypergraph G , we have that, $\text{VC}(G)/\text{LP}(G) \leq \frac{k}{2}$, where $\text{VC}(G)$ denotes the size of the minimum vertex cover and $\text{LP}(G)$ denotes the value of the standard LP relaxation. This yields an efficient $\frac{k}{2}$ approximation for k -HYPVC-PARTITE.*

Note that the standard LP relaxation does not utilize the knowledge of the k -partition and therefore, by Lovász's result, $\text{LP}(G)$ is a $\frac{k}{2}$ approximation to $\text{VC}(G)$ even when the k -partition is not known. The above upper bound was shown to be tight by Aharoni, Holzman and Krivelevich [1] who proved the following theorem.

Theorem 2 (Aharoni et al. [1]). *For every $k \geq 3$, there exists a family of k -partite k -uniform hypergraphs G such that $\text{VC}(G)/\text{LP}(G) \geq \frac{k}{2} - o(1)$. Thus, the integrality gap of the standard LP relaxation is $\frac{k}{2} - o(1)$.*

The problem was shown to be APX-hard in [17] and [11] for $k = 3$ which can be extended easily to $k \geq 3$. A recent work of Guruswami and Saket [13] showed the following non-trivial hardness of approximation factor for general k .

Theorem 3 (Guruswami and Saket [13]). *For arbitrary $\varepsilon > 0$ and any integer $k \geq 5$, k -HYPVC-PARTITE is NP-hard to approximate within a factor of $\frac{k}{4} - \varepsilon$. Assuming the UGC yields an optimal hardness factor of $\frac{k}{2} - \varepsilon$ for $k \geq 3$.*

Our Contribution. We show a nearly optimal NP-hardness result for approximating k -HYPVC-PARTITE.

Theorem 4. *For any $\epsilon > 0$ and integer $k \geq 4$, it is NP-hard to approximate the minimum vertex cover on k -partite k -uniform hypergraphs within to a factor of $\frac{k}{2} - 1 + \frac{1}{2k} - \epsilon$.*

Our result significantly improves on the NP-hardness factor obtained in [13] and is off by at most an additive constant of 1 from the optimal for any $k \geq 4$. The next few paragraphs give an overview of the techniques used in this work.

Techniques. It is helpful to briefly review the hardness reduction of [8] for k -HYPVC.

The main idea of their construction can be illustrated by the following *gadget*. Consider a domain R and the set of all its subsets $\mathcal{H} = 2^R$. Sample subsets from \mathcal{H} by choosing each element of R independently with probability $1 - 1/k - \epsilon$ (for some small $\epsilon > 0$), and let the weight of each subset in \mathcal{H} be its sampling probability, thus making the sum of all weights to be 1. The set \mathcal{H} along with the weights is an example of a biased Long Code over R . Construct a k -uniform hypergraph over the vertex set \mathcal{H} by adding an edge between any k subsets whose intersection is empty. In this hypergraph, every element $r \in R$ yields a corresponding independent set of weight $(1 - 1/k - \epsilon)$, by choosing all subsets which contain r . On the other hand, Dinur *et al.* [8] show via analysis based on extremal set theory, that any independent set of weight ϵ must contain k subsets in \mathcal{H} which have a small intersection, thus yielding a special *small* subset of R – a property that is utilized in proving their hardness result. Note that this gap of $1 - 1/k - \epsilon$ vs ϵ for independent set corresponds to a gap of $1/k + \epsilon$ vs $1 - \epsilon$ for the minimum vertex cover.

The construction of [8] combines the above Long Code based gadget with a new *Multi-Layered PCP*. This is a two variable CSP consisting of several *layers* of variables, and constraints between the variables of each pair of layers. The work of [8] shows that it is NP-hard to find a labeling to the variables which satisfies a small fraction of the constraints between *any* two layers, even if there is a labeling that satisfies all the constraints of the instance. The reduction to a k -uniform hypergraph (as an instance of k -HYPVC) involves replacing each variable of the PCP with a biased Long Code and adding the edges of the gadget across different Long Codes.

The starting point for our hardness reduction for k -HYPVC-PARTITE is – as in [8] – the Multi-Layered PCP. While we do not explicitly construct a standalone Long Code based gadget, our reduction can be thought of as adapting the integrality gap construction of Aharoni *et al.* [1] into a Long Code based gadget in a manner that preserves the k -uniformity and k -partiteness of the integrality gap.

Such transformations of integrality gaps into Long Code based gadgets have recently been studied in the works of Raghavendra [25] and Kumar, Manokaran, Tulsiani and Vishnoi [22] for a wide class of CSPs and their appropriate LP and SDP integrality gaps. These Long Code based gadgets can be combined with a Unique Games instance to yield tight UGC based hardness results, where the reduction is analyzed via Mossel’s *Invariance Principle* [24]. Indeed, for k -HYPVC-PARTITE the work of Guruswami and Saket [13] combines the integrality gap of [1] with (a slight modification) of the approach of Kumar *et al.* [22] to obtain an optimal UGC based hardness result.

Since our reduction starts with the Multi-Layered PCP instead of Unique Games, we cannot adopt a Invariance Principle based analysis. Thus, in a flavor similar to that of [8], our analysis is via extremal combinatorics. However, our *gadget* involves several biased Long Codes with different biases and each hyperedge includes vertices from

differently biased Long Codes, unlike the construction in [8]. The different biases are derived from the LP solution to the integrality gap of [11], in such a way that in the gap obtained in the gadget corresponds (roughly) to the value of the integrality gap.

For our analysis, we use structural properties of a *cross-intersecting* collection of set families. A collection of set families is *cross-intersecting* if any intersection of subsets – each chosen from a different family – is large. Variants of this notion have previously been studied in extremal set theory, see for example [2]. We prove an upper bound on the measure of the smallest family in such a collection. This enables a small vertex cover (in the hypergraph) to be *decoded* into a good labeling to the Multi-Layered PCP.

The next section defines and analyzes the above mentioned cross-intersecting set families. Sec. 3 defines the Multi-Layered PCP of Dinur *et al.* [8] and states their hardness for it. In Sec. 4 we describe our reduction and prove Thm 4.

2 Cross-Intersecting Set Families

We use the notation $[n] = \{1, \dots, n\}$ and $2^{[n]} = \{F \mid F \subseteq [n]\}$. We begin by defining cross-intersecting set families:

Definition 5. A collection of k families $\mathcal{F}_1, \dots, \mathcal{F}_k \subseteq 2^{[n]}$, is called k -wise t -cross-intersecting if for every choice of sets $F_i \in \mathcal{F}_i$ for $i \in [k]$, we have $|F_1 \cap \dots \cap F_k| \geq t$.

We will work with the p -biased measure on $2^{[n]}$, which is defined as follows:

Definition 6. Given a bias $0 < p < 1$, define the measure μ_p on $2^{[n]}$ as: $\mu_p(F) := p^{|F|} \cdot (1 - p)^{n - |F|}$. The measure of a family \mathcal{F} is defined as $\mu_p(\mathcal{F}) = \sum_{F \in \mathcal{F}} \mu_p(F)$.

Now, we describe an important technique for analyzing cross-intersecting families – the shift operation (see Def 4.1, pg. 1298 [12]). Given a family \mathcal{F} , define the (i, j) -shift as:

$$S_{ij}^{\mathcal{F}}(F) = \begin{cases} (F \cup \{i\} \setminus \{j\}) & \text{if } j \in F, i \notin F \text{ and } (F \cup \{i\} \setminus \{j\}) \notin \mathcal{F} \\ F & \text{otherwise.} \end{cases}$$

Let the (i, j) -shift of a family \mathcal{F} be $S_{ij}(\mathcal{F}) = \{S_{ij}^{\mathcal{F}}(F) \mid F \in \mathcal{F}\}$. Given a family $\mathcal{F} \subseteq 2^{[n]}$, we repeatedly apply (i, j) -shift for $1 \leq i < j \leq n$ to \mathcal{F} until we obtain a family that is invariant under these shifts. Such a family is called a *left-shifted family* and we will denote it by $S(\mathcal{F})$. The following observations follow from the definition.

Observation 7. Let $\mathcal{F} \subseteq 2^{[n]}$ be a left-shifted family. Consider $F \in \mathcal{F}$ such that $i \notin F$ and $j \in F$ where $i < j$. Then, $(F \cup \{i\} \setminus \{j\})$ must be in \mathcal{F} .

Observation 8. Given $\mathcal{F} \subseteq 2^{[n]}$, there is a bijection between \mathcal{F} and $S(\mathcal{F})$ that preserves the size of the set. Thus, for any p , $\mu_p(\mathcal{F}) = \mu_p(S(\mathcal{F}))$.

The following lemma shows that the cross-intersecting property is preserved under left-shifting. A similar fact for a single left-shifted family was shown in [12] (pg. 1311, Lem. 8.3), which was reproved and used in [8]. We omit the proof.

Lemma 9. Consider families $\mathcal{F}_1, \dots, \mathcal{F}_k \subseteq 2^{[n]}$ that are k -wise t -cross-intersecting. Then, the families $S(\mathcal{F}_1), \dots, S(\mathcal{F}_k)$ are also k -wise t -cross-intersecting.

Next, we prove a key structural property of left-shifted cross-intersecting families which states that for at least one of the families, all of its subsets have a dense prefix.

Lemma 10. *Let $\mathcal{F}_1, \dots, \mathcal{F}_k \subseteq 2^{[n]}$ be left-shifted families that are k -wise t -cross-intersecting for some $t \geq 1$ and let $q_1, \dots, q_k \in (0, 1)$ be k numbers such that $\sum_i q_i \geq 1$. Then, there exists a $j \in [k]$ such that for all sets $F \in \mathcal{F}_j$, there exists a positive integer $r_F \leq n - t$ such that $|F \cap [t + r_F]| > (1 - q_j)(t + r_F)$.*

Proof. Let us assume to the contrary that for every $i \in [k]$, there exists a set $F_i \in \mathcal{F}_i$ such that for all $r \geq 0$, $|F_i \cap [t + r]| \leq (1 - q_i)(t + r)$. The following combinatorial argument shows that the families \mathcal{F}_i cannot be k -wise t -cross-intersecting.

Let us construct an arrangement of balls and bins where each ball is colored with one of k colors. Create n bins labeled $1, \dots, n$. For each i and for every $x \in [n] \setminus F_i$, we place a ball with color i in the bin labeled x . Note that a bin can have several balls, but they must have distinct colors. Given such an arrangement, we can recover the sets it represents by defining F_i^c to be the set of bins that contain a ball with color i .

For all r , our assumption implies that $|F_i^c \cap [t + r]| \geq q_i(t + r)$. Thus, there are at least $\lceil q_i(t + r) \rceil$ balls with color i in bins labeled $1, \dots, t + r$. The total number of balls in bins labeled $1, \dots, t + r$ is,

$$\sum_{i=1}^k |F_i^c \cap [t + r]| \geq \sum_{i=1}^k \lceil q_i(t + r) \rceil \geq \sum_{i=1}^k q_i(t + r) \geq t + r \geq r + 1,$$

where the last two inequalities follow using $\sum_i q_i \geq 1$ and $t \geq 1$.

Next, we describe a procedure to manipulate the above arrangement of balls.

for $r := 0$ to $n - t$
if bin $t + r$ is empty
then if a bin labeled from 1 to $t - 1$ contains a ball **then** move it to bin $t + r$
else if a bin labeled from t to $t + r - 1$ contains two balls
then move one of them to bin $t + r$ **else** output “error”

We need the following lemma.

Lemma 11. *The above procedure satisfies the following properties:*

1. *The procedure never outputs error.*
2. *At every step, any two balls in the same bin have different colors.*
3. *At step r , define $G_i^{(r)}$ to be the set of labels of the bins that do not contain a ball of color i . Then, for all $i \in [k]$, $G_i^{(r)} \in \mathcal{F}_i$.*
4. *After step r , the bins t to $t + r$ have at least one ball each.*

Proof. 1. If it outputs error at step r , there must be at most r balls in bins 1 to $t + r$. At the start of the procedure, there are at least $r + 1$ balls in these bins and during the first r steps, the number of balls in these bins remain unchanged. This is a contradiction.

2. This is true at $r = 0$ and balls are moved only to empty bins. This proves the claim.

3. Whenever we move a ball from bin i to j , we have $i < j$. Since \mathcal{F}_i are left-shifted, by repeated application of Observation 7, we get that at step r , $G_i^{(r)} \in \mathcal{F}_i$.

4. Since the procedure never outputs error, at step r , if the bin $t + r$ is empty, the procedure places a ball in it while not emptying any bin labeled between $[t, t + r - 1]$. ■

The above lemma implies that at the end of the procedure (after $r = n - t$), there is a ball in each of the bins labeled from $[t, n]$. Thus, the sets $G_i = G_i^{(n-t)}$ satisfy $\cap_i G_i \subseteq [t - 1]$ and hence $|\cap_i G_i| \leq t - 1$. Also, we know that $G_i \in \mathcal{F}_i$. Thus, the families \mathcal{F}_i cannot be k -wise t -cross-intersecting. This completes the proof of Lem. 10. ■

The above lemma, along with a Chernoff bound argument, shows that: Given a collection of k -wise t -cross-intersecting families, one of them must have a small measure under an appropriately chosen bias. We omit the proof.

Lemma 12. *For arbitrary $\epsilon, \delta > 0$, there exists some $t = O\left(\frac{1}{\delta^2} \log\left(\frac{1}{\epsilon} \left(1 + \frac{1}{2\delta^2}\right)\right)\right)$ such that the following holds: Given k numbers $0 < q_i < 1$ such that $\sum_i q_i \geq 1$ and k families, $\mathcal{F}_1, \dots, \mathcal{F}_k \subseteq 2^{[n]}$, that are k -wise t -cross-intersecting, there exists a j such that $\mu_{1-q_j-\delta}(\mathcal{F}) < \epsilon$.*

3 Multi-Layered PCP

In this section we describe the Multi-Layered PCP constructed in [8] and its useful properties. An instance Φ of the Multi-Layered PCP is parametrized by integers $L, R > 1$. The PCP consists of L sets of variables X_1, \dots, X_L . The label set (or range) of the variables in the l^{th} set X_l is a set R_{X_l} where $|R_{X_l}| = R^{O(L)}$. For any two integers $1 \leq l < l' \leq L$, the PCP has a set of constraints $\Phi_{l,l'}$ in which each constraint depends on one variable $x \in X_l$ and one variable $x' \in X_{l'}$. The constraint (if it exists) between $x \in X_l$ and $x' \in X_{l'}$ ($l < l'$) is denoted and characterized by a projection $\pi_{x \rightarrow x'} : R_{X_l} \rightarrow R_{X_{l'}}$. A labeling to x and x' satisfies the constraint $\pi_{x \rightarrow x'}$ if the projection (via $\pi_{x \rightarrow x'}$) of the label assigned to x coincides with the label assigned to x' .

The following useful ‘weak-density’ property of the Multi-Layered PCP was defined in [8].

Definition 13. *An instance Φ of the Multi-Layered PCP with L layers is weakly-dense if for any $\delta > 0$, given $m \geq \lceil \frac{2}{\delta} \rceil$ layers $l_1 < l_2 < \dots < l_m$ and given any sets $S_i \subseteq X_{l_i}$, for $i \in [m]$ such that $|S_i| \geq \delta |X_{l_i}|$; there always exist two layers $l_{i'}$ and $l_{i''}$ such that the constraints between the variables in the sets $S_{i'}$ and $S_{i''}$ is at least $\frac{\delta^2}{4}$ fraction of the constraints between the sets $X_{l_{i'}}$ and $X_{l_{i''}}$.*

The following inapproximability of the Multi-Layered PCP was proven by Dinur et al. [8] based on the PCP Theorem [4,3] and Raz’s Parallel Repetition Theorem [26].

Theorem 14. *There exists a universal constant $\gamma > 0$ such that for any parameters $L > 1$ and R , there is a weakly-dense L -layered PCP $\Phi = \cup \Phi_{l,l'}$ such that it is NP-hard to distinguish between the following two cases:*

- **YES Case:** *There exists an assignment of labels to the variables of Φ that satisfies all the constraints.*
- **NO Case:** *For every $1 \leq l < l' \leq L$, not more than $1/R^\gamma$ fraction of the constraints in $\Phi_{l,l'}$ can be satisfied by any assignment.*

4 Hardness Reduction for HYPVC-PARTITE

4.1 Construction of the Hypergraph

Fix a $k \geq 3$, an arbitrarily small parameter $\varepsilon > 0$ and let $r = \lceil 10\varepsilon^{-2} \rceil$. We shall construct a $(k+1)$ -uniform $(k+1)$ -partite hypergraph as an instance of $(k+1)$ -HYPVC-PARTITE. Our construction will be a reduction from an instance Φ of the Multi-Layered PCP with number of layers $L = 32\varepsilon^{-2}$ and parameter R which shall be chosen later to be large enough. It involves creating, for each variable of the PCP, several copies of the Long Code endowed with different biased measures as explained below.

Over any domain T , a Long Code \mathcal{H} is a collection of all subsets of T , i.e., $\mathcal{H} = 2^T$. A bias $p \in [0, 1]$ defines a measure μ_p on \mathcal{H} such that $\mu_p(v) = p^{|v|}(1-p)^{|T \setminus v|}$ for any $v \in \mathcal{H}$. In our construction we need several different biased measures defined as follows. For all $j = 1, \dots, r$, define $q_j := \frac{2j}{rk}$, and biases $p_j := 1 - q_j - \varepsilon$. Each p_j defines a biased measure μ_{p_j} over a Long Code over any domain. Next, we define the vertices of the hypergraph.

Vertices. We shall denote the set of vertices by V . Consider a variable x in the layer X_l of the PCP. For $i \in [k+1]$ and $j \in [r]$, let \mathcal{H}_{ij}^x be a Long Code on the domain R_{X_l} endowed with the bias μ_{p_j} , i.e., $\mu_{p_j}(v) = p_j^{|v|}(1-p_j)^{|R_{X_l} \setminus v|}$ for all $v \in \mathcal{H}_{ij}^x = 2^{R_{X_l}}$. The set of vertices corresponding to x is $V[x] := \bigcup_{i=1}^{k+1} \bigcup_{j=1}^r \mathcal{H}_{ij}^x$. We define the weights on vertices to be proportional to its biased measure in the corresponding Long Code. Formally, for any $v \in \mathcal{H}_{ij}^x$,

$$\text{wt}(v) := \frac{\mu_{p_j}(v)}{L|X_l|r(k+1)}.$$

The above conveniently ensures that for any $l \in [L]$, $\sum_{x \in X_l} \text{wt}(V[x]) = 1/L$, and $\sum_{l \in [L]} \sum_{x \in X_l} \text{wt}(V[x]) = 1$. In addition to the vertices for each variable of the PCP, the instance also contains $k+1$ dummy vertices d_1, \dots, d_{k+1} each with a very large weight given by $\text{wt}(d_i) := 2$ for $i \in [k+1]$. Clearly, this ensures that the total weight of all the vertices in the hypergraph is $2(k+1) + 1$. The edges shall be defined in such a way so as to ensure that the maximum sized independent set contains all the dummy vertices. We define the $(k+1)$ partition (V_1, \dots, V_{k+1}) of V to be:

$$V_i = \left(\bigcup_{l=1}^L \bigcup_{x \in X_l} \bigcup_{j=1}^r \mathcal{H}_{ij}^x \right) \cup \{d_i\},$$

for all $i = 1, \dots, k+1$.

We now define the hyperedges of the instance. In the rest of the section, the vertices shall be thought of as subsets of their respective domains.

Hyperedges. For every pair of variables x, y in Φ such that there is a constraint $\pi_{x \rightarrow y}$, we construct edges as follows:

- (1.) Consider all permutations $\sigma : [k+1] \rightarrow [k+1]$ and sequences $(j_1, \dots, j_k, j_{k+1})$ such that, $j_1, \dots, j_k \in [r] \cup \{0\}$ and $j_{k+1} \in [r]$ such that: $\sum_{i=1}^k \mathbb{1}_{\{j_i \neq 0\}} q_{j_i} \geq 1$.
- (2.) Add all possible hyperedges e such that for all $i \in [k]$:

- (2.a) If $j_i \neq 0$ then $e \cap V_{\sigma(i)} =: v_{\sigma(i)} \in \mathcal{H}_{\sigma(i),j_i}^x$, and,
- (2.b) If $j_i = 0$ then $e \cap V_{\sigma(i)} = d_{\sigma(i)}$ and,
- (2.c) $e \cap V_{\sigma(k+1)} =: u_{\sigma(k+1)} \in \mathcal{H}_{\sigma(k+1),j_{k+1}}^y$, which satisfy,

$$\pi_{x \rightarrow y} \left(\bigcap_{i: i \in [k], j_i \neq 0} v_{\sigma(i)} \right) \cap u_{\sigma(k+1)} = \emptyset. \tag{1}$$

Let us denote the hypergraph constructed above by $G(\Phi)$. From the construction it is clear that $G(\Phi)$ is $(k + 1)$ -partite with partition $V = \cup_{i \in [k+1]} V_i$.

The role of the dummy vertices $\{d_i\}_i$ is to ensure that each hyperedge contains exactly $k + 1$ vertices – without them we would have hyperedges with fewer than $k + 1$ vertices. Also note that the hyperedges are defined in such a way that the set $\{d_1, \dots, d_{k+1}\}$ is an independent set. Moreover, since the weight of each d_i is 2, while the total weight of all vertices except $\{d_i\}_i$ is 1, this implies that any maximum independent set \mathcal{I} contains all the dummy vertices. Thus, $V \setminus \mathcal{I}$ is a minimum vertex cover that does not contain any dummy vertices. For convenience, the analysis of our reduction shall focus on the weight of $(\mathcal{I} \cap V) \setminus \{d_1, \dots, d_{k+1}\}$.

The rest of this section proves the following theorem which implies Thm. 4.

Theorem 15. *Let Φ be the instance of Multi-Layered PCP from which the hypergraph $G(\Phi)$ is derived as an instance of $(k + 1)$ -HYPVC-PARTITE. Then,*

- *Completeness: If Φ is a YES instance, then there is an independent set \mathcal{I}^* in $G(\Phi)$ such that,*

$$\text{wt}(\mathcal{I}^* \cap (V \setminus \{d_1, \dots, d_{k+1}\})) \geq 1 - \frac{1}{k} - 2\varepsilon.$$

- *Soundness: If Φ is a NO instance, then for all maximum independent sets \mathcal{I} in $G(\Phi)$,*

$$\text{wt}(\mathcal{I} \cap (V \setminus \{d_1, \dots, d_{k+1}\})) \leq 1 - \frac{k}{2(k+1)} + \varepsilon.$$

The completeness is proved in Sec. 4.2 and the soundness in Sec. 4.3.

4.2 Completeness

In the completeness case, the instance Φ is a YES instance, i.e., there is a labeling A which maps each variable x in layer X_l to an assignment in R_{X_l} for all $l = 1, \dots, L$, such that all the constraints of Φ are satisfied.

Consider the set of vertices \mathcal{I}^* which satisfies the following properties:

1. $d_i \in \mathcal{I}^*$ for all $i = 1, \dots, k + 1$.
2. For all $l \in [L]$, $x \in X_l$, $i \in [k + 1]$, $j \in [r]$,

$$\mathcal{I}^* \cap \mathcal{H}_{ij}^x = \{v \in \mathcal{H}_{ij}^x : A(x) \in v\}. \tag{2}$$

Suppose x and y are two variables in Φ with a constraint $\pi_{x \rightarrow y}$ between them. Consider any $v \in \mathcal{I}^* \cap V[x]$ and $u \in \mathcal{I}^* \cap V[y]$. Since the labeling A satisfies the constraint $\pi_{x \rightarrow y}$, we have that $A(x) \in v$ and $A(y) \in u$ and $A(y) \in \pi_{x \rightarrow y}(v) \cap u$. Therefore,

Eq. (1) is not satisfied by the vertices in \mathcal{I}^* , and so \mathcal{I}^* is an independent set. By Eq. (2), the fraction of the weight of the Long Code \mathcal{H}_{ij}^x which lies in \mathcal{I}^* is p_j , for any variable x , $i \in [k + 1]$ and $j \in [r]$. Therefore,

$$\frac{\text{wt}(\mathcal{I}^* \cap V[x])}{\text{wt}(V[x])} = \frac{1}{r} \sum_{j=1}^r p_j = 1 - \frac{1}{k} \left(1 + \frac{1}{r}\right) - \varepsilon,$$

by our setting of p_j in Sec. 4.1. The above yields that

$$\text{wt}(\mathcal{I}^* \cap (V \setminus \{d_1, \dots, d_{k+1}\})) = 1 - \frac{1}{k} \left(1 + \frac{1}{r}\right) - \varepsilon \geq 1 - \frac{1}{k} - 2\varepsilon, \quad (3)$$

for a small enough value of $\varepsilon > 0$ and our setting of the parameter r .

4.3 Soundness

For the soundness analysis we have that Φ is a NO instance as given in Thm. 1.4 and we wish to prove that the size of the maximum independent set in $G(\Phi)$ is appropriately small. For a contradiction, assume that there is a maximum independent set \mathcal{I} in $G(\Phi)$ such that, $\text{wt}(\mathcal{I} \cap (V \setminus \{d_1, \dots, d_{k+1}\})) \geq 1 - \frac{k}{2(k+1)} + \varepsilon$. Define the set of variables X' as:

$$X' := \left\{ x \text{ a variable in } \Phi : \frac{\text{wt}(\mathcal{I} \cap V[x])}{\text{wt}(V[x])} \geq 1 - \frac{k}{2(k+1)} + \frac{\varepsilon}{2} \right\}. \quad (4)$$

An averaging argument shows that $\text{wt}(\cup_{x \in X'} V[x]) \geq \varepsilon/2$. A further averaging implies that there are $\frac{\varepsilon}{4}L = \frac{8}{\varepsilon}$ layers of Φ such that $\frac{\varepsilon}{4}$ fraction of the variables in each of these layers belong to X' . Applying the Weak Density property of Φ given by Definition 1.3 and Thm. 1.4 yields two layers $X_{l'}$ and $X_{l''}$ ($l' < l''$) such that $\frac{\varepsilon^2}{64}$ fraction of the constraints between them are between variables in X' . The rest of the analysis shall focus on these two layers and for convenience we shall denote $X' \cap X_{l'}$ by X and $X' \cap X_{l''}$ by Y , and denote the respective label sets by R_X and R_Y .

Consider any variable $x \in X$. For any $i \in [k + 1], j \in [r]$, call a Long Code \mathcal{H}_{ij}^x significant if $\mu_{p_j}(\mathcal{I} \cap \mathcal{H}_{ij}^x) \geq \frac{\varepsilon}{2}$. From Eq. (4) and an averaging argument we obtain,

$$|\{(i, j) \in [k + 1] \times [r] : \mathcal{H}_{ij}^x \text{ is significant}\}| \geq \left(1 - \frac{k}{2(k+1)}\right) r(k+1). \quad (5)$$

An analogous statement holds for every variable $y \in Y$ and corresponding Long Codes \mathcal{H}_{ij}^y . The following structural lemma now follows. We omit the proof.

Lemma 16. *Consider any variable $x \in X$. Then there exists a sequence (j_1, \dots, j_{k+1}) with $j_i \in [r] \cup \{0\}$ for $i \in [k + 1]$; such that the Long Codes $\{\mathcal{H}_{i,j_i}^x \mid i \in [k + 1] \text{ where } j_i \neq 0\}$, are all significant. Moreover, $\sum_{i=1}^{k+1} j_i \geq \frac{rk}{2} + r$.*

Next we define the decoding procedure to define a label for any given variable $x \in X$.

Labeling for variable $x \in X$. The label $A(x)$ for each variable $x \in X$ is chosen independently via the following three step (randomized) procedure.

Step 1. Choose a sequence (j_1, \dots, j_{k+1}) yielded by Lem. 16 applied to x .

Step 2. Choose an element i_0 uniformly at random from $[k + 1]$.

Before describing the third step of the procedure, we require the following lemma that is proved using Lem. 16 and Lem. 12. We omit the proof.

Lemma 17. *There exist vertices $v_i \in \mathcal{I} \cap \mathcal{H}_{i_j^x}^x$ for every $i : i \in [k + 1] \setminus \{i_0\}, j_i \neq 0$, and an integer $t := t(\varepsilon)$ satisfying:*

$$\left| \bigcap_{i:i \neq i_0; j_i \neq 0} v_i \right| < t. \tag{6}$$

The third step of the labeling procedure is as follows:

Step 3. Apply Lem. 17 to obtain the vertices $v_i \in \mathcal{I} \cap \mathcal{H}_{i_j^x}^x$ for every $i : i \in [k + 1] \setminus \{i_0\}, j_i \neq 0$ satisfying Eq. (6). Define $B(x)$ as,

$$B(x) := \bigcap_{i:i \neq i_0; j_i \neq 0} v_i,$$

noting that $|B(x)| < t$. Assign x a random label from $B(x)$ and call the label $A(x)$.

Labeling for variable $y \in Y$. After labeling the variables $x \in X$ via the procedure above, we construct a labeling $A(y)$ for any variable $y \in Y$ by defining,

$$A(y) := \operatorname{argmax}_{a \in R_Y} |\{x \in X \cap N(y) \mid a \in \pi_{x \rightarrow y}(B(x))\}|,$$

where $N(y)$ is the set of all variables that have a constraint with y . The above process selects a label for y which lies in maximum number of projections of $B(x)$ for variables $x \in X$ which have a constraint with y .

The rest of this section lower bounds the number of constraints satisfied by the labeling process, and thus obtains a contradiction to the fact that Φ is a NO instance.

Lower bounding the number of satisfied constraints. Fix a variable $y \in Y$. Let $U(y) := X \cap N(y)$, i.e., the variables in X which have a constraint with y . Further, define the set $P(y) \subseteq [k + 1]$ as follows,

$$P(y) = \{i \in [k + 1] \mid \exists j \in [r] \text{ such that } \mu_{p_j}(\mathcal{I} \cap \mathcal{H}_{i_j^y}^y) \geq \varepsilon/2\}.$$

In other words, $P(y)$ is the set of all those indices in $[k+1]$ such that there is a *significant* Long Code corresponding to each of them. Applying Eq. (5) to y we obtain that there at least $\frac{r(k+2)}{2}$ *significant* Long Codes corresponding to y , and therefore $|P(y)| \geq \frac{k+2}{2} \geq 1$. Next we define subsets of $U(y)$ depending on the outcome of Step 2 in the labeling procedure for variables $x \in U(y)$. For $i \in [k + 1]$ define,

$$U(i, y) := \{x \in U(y) \mid i \text{ was chosen in Step 2 of the labeling procedure for } x\},$$

Also, define $U^*(y) := \bigcup_{i \in P(y)} U(i, y)$. Note that $\{U(i, y)\}_{i \in [k+1]}$ is a partition of $U(y)$. Also, since $|P(y)| \geq \frac{k+1}{2}$ and the labeling procedure for each variable x chooses the index in Step 2 uniformly and independently at random we have,

$$\mathbb{E}[|U^*(y)|] \geq \frac{|U(y)|}{2}, \tag{7}$$

where the expectation is over the random choice of the indices in Step 2 of the labeling procedure for all $x \in U(y)$. Before continuing we need the following simple lemma (proved as Claim 5.4 in [8]).

Lemma 18. *Let A_1, \dots, A_N be a collection of N sets, each of size at most $T \geq 1$. If there are not more than D pairwise disjoint sets in the collection, then there is an element that is contained in at least $\frac{N}{TD}$ sets.*

Now consider any $i' \in P(y)$ such that $U(i', y) \neq \emptyset$ and a variable $x \in U(i', y)$. Since $i' \in P(y)$ there is a *significant* Long Code $\mathcal{H}_{i',j'}^y$ for some $j' \in [r]$. Furthermore, since \mathcal{I} is an independent set there cannot be a $u \in \mathcal{I} \cap \mathcal{H}_{i',j'}^y$ such that $\pi_{x \rightarrow y}(B(x)) \cap u = \emptyset$, otherwise the following set of $k + 1$ vertices,

$$\{v_i \mid i \in [k + 1] \setminus \{i'\}, j_i \neq 0\} \cup \{d_i \mid i \in [k + 1] \setminus \{i'\}, j_i = 0\} \cup \{u\}$$

form an edge in \mathcal{I} , where $\{v_i\}, \{j_i\}$ were picked in the labeling procedure for x , and the vertices $\{d_i\} \in \mathcal{I}$ since it is a maximum independent set.

Consider the collection of sets $\pi_{x \rightarrow y}(B(x))$ for all $x \in U(i', y)$. Clearly each set is of size less than t . Let D be the maximum number of disjoint sets in this collection. Each disjoint set independently reduces the measure of $\mathcal{I} \cap \mathcal{H}_{i',j'}^y$ by a factor of $(1 - (1 - p_{j'})^t)$. However, since $\mu_{p_{j'}}(\mathcal{I} \cap \mathcal{H}_{i',j'}^y)$ is at least $\frac{\varepsilon}{2}$, this implies that D is at most $\log(\frac{\varepsilon}{2}) / \log(1 - (2/rk)^t)$, since $p_{j'} \leq 1 - \frac{2}{rk}$. Moreover, since t and r depends only on ε , the upper bound on D also depends only on ε .

Therefore by Lem. [18] there is an element $a \in R_Y$ such that $a \in \pi_{x \rightarrow y}(B(x))$ for at least $\frac{1}{Dt}$ fraction of $x \in U(i', y)$. Noting that this bound is independent of j' and that $\{U(i', y)\}_{i' \in P(y)}$ is a partition of $U^*(y)$, we obtain that there is an element $a \in R_Y$ such that $a \in \pi_{x \rightarrow y}(B(x))$ for $\frac{1}{(k+1)Dt}$ fraction of $x \in U^*(y)$. Therefore, in Step 3 of the labeling procedure when a label $A(x)$ is chosen uniformly at random from $B(x)$, in expectation, $a = \pi_{x \rightarrow y}(A(x))$ for $\frac{1}{(k+1)Dt^2}$ fraction of $x \in U^*(y)$. Combining this with Eq. (7) gives us that there is a labeling to the variables in X and Y which satisfies $\frac{1}{2(k+1)Dt^2}$ fraction of the constraints between variables in X and Y which is in turn at least $\frac{\varepsilon^2}{64}$ fraction of the constraints between the layers $X_{l'}$ and $X_{l''}$. Since D and t depend only on ε , choosing the parameter R of Φ to be large enough we obtain a contradiction to our supposition on the lower bound on the size of the independent set. Therefore in the Soundness case, any for any independent set \mathcal{I} ,

$$\text{wt}(\mathcal{I} \cap (V \setminus \{d_1, \dots, d_{k+1}\})) \leq 1 - \frac{k}{2(k+1)} + \varepsilon.$$

Combining the above with Eq. (3) of the analysis in the Completeness case yields a factor $\frac{k^2}{2(k+1)} - \delta$ (for any $\delta > 0$) hardness for approximating $(k+1)$ -HYPVC-PARTITE. Thus, we obtain a factor $\frac{k}{2} - 1 + \frac{1}{2k} - \delta$ hardness for approximating k -HYPVC-PARTITE.

References

1. Aharoni, R., Holzman, R., Krivelevich, M.: On a theorem of Lovász on covers in r -partite hypergraphs. *Combinatorica* 16(2), 149–174 (1996)
2. Alon, N., Lubetzky, E.: Uniformly cross intersecting families. *Combinatorica* 29(4), 389–431 (2009)

3. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and the hardness of approximation problems. *J. ACM* 45(3), 501–555 (1998)
4. Arora, S., Safra, S.: Probabilistic checking of proofs: A new characterization of NP. *J. ACM* 45(1), 70–122 (1998)
5. Austrin, P., Khot, S., Safra, M.: Inapproximability of vertex cover and independent set in bounded degree graphs. *Theory of Computing* 7(1), 27–43 (2011)
6. Bansal, N., Khot, S.: Inapproximability of hypergraph vertex cover and applications to scheduling problems. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) *ICALP 2010. LNCS*, vol. 6198, pp. 250–261. Springer, Heidelberg (2010)
7. Dinur, I., Guruswami, V., Khot, S.: Vertex cover on k -uniform hypergraphs is hard to approximate within factor $(k-3-\epsilon)$. *ECCC Technical Report TR02-027* (2002)
8. Dinur, I., Guruswami, V., Khot, S., Regev, O.: A new multilayered pcp and the hardness of hypergraph vertex cover. *SIAM J. Comput.* 34, 1129–1146 (2005)
9. Dinur, I., Safra, S.: On the hardness of approximating minimum vertex cover. *Annals of Mathematics* 162, 439–485 (2004)
10. Feder, T., Motwani, R., O’Callaghan, L., Panigrahy, R., Thomas, D.: Online distributed predicate evaluation. Technical Report, Stanford University (2003)
11. Gottlob, G., Senellart, P.: Schema mapping discovery from data instances. *J. ACM* 57(2) (January 2010)
12. Graham, R.L., Grötschel, M., Lovász, L. (eds.): *Handbook of combinatorics*, vol. 2. MIT Press, Cambridge (1995)
13. Guruswami, V., Saket, R.: On the inapproximability of vertex cover on k -partite k -uniform hypergraphs. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) *ICALP 2010. LNCS*, vol. 6198, pp. 360–371. Springer, Heidelberg (2010)
14. Håstad, J.: Some optimal inapproximability results. *J. ACM* 48(4), 798–859 (2001)
15. Halperin, E.: Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. *SIAM J. Comput.* 31(5), 1608–1623 (2002)
16. Holmerin, J.: Improved inapproximability results for vertex cover on k -uniform hypergraphs. In: Widmayer, P., Triguero, F., Morales, R., Hennessy, M., Eidenbenz, S., Conejo, R. (eds.) *ICALP 2002. LNCS*, vol. 2380, p. 1005. Springer, Heidelberg (2002)
17. Ilie, L., Solis-Oba, R., Yu, S.: Reducing the size of NFAs by using equivalences and preorders. In: Apostolico, A., Crochemore, M., Park, K. (eds.) *CPM 2005. LNCS*, vol. 3537, pp. 310–321. Springer, Heidelberg (2005)
18. Karakostas, G.: A better approximation ratio for the vertex cover problem. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) *ICALP 2005. LNCS*, vol. 3580, pp. 1043–1050. Springer, Heidelberg (2005)
19. Karp, R.M.: Reducibility among combinatorial problems. *Complexity of Computer Computations*, 85–103 (1972)
20. Khot, S.: On the power of unique 2-prover 1-round games. In: *Proc. 34th ACM STOC*, pp. 767–775 (2002)
21. Khot, S., Regev, O.: Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.* 74(3), 335–349 (2008)
22. Kumar, A., Manokaran, R., Tulsiani, M., Vishnoi, N.K.: On LP-based approximability for strict CSPs. In: *Proc. SODA* (2011)
23. Lovász, L.: On minimax theorems of combinatorics. *Doctoral Thesis, Mathematiki Lapok* 26, 209–264 (1975)
24. Mossel, E.: Gaussian bounds for noise correlation of functions and tight analysis of long codes. In: *Proc. 49th IEEE FOCS*, pp. 156–165 (2008)
25. Raghavendra, P.: Optimal algorithms and inapproximability results for every CSP? In: *Proc. 40th ACM STOC*, pp. 245–254 (2008)
26. Raz, R.: A parallel repetition theorem. *SIAM J. Comput.* 27(3), 763–803 (1998)
27. Trevisan, L.: Non-approximability results for optimization problems on bounded degree instances. In: *Proc. 33rd ACM STOC*, pp. 453–461 (2001)

A Linear Time Approximation Scheme for Maximum Quartet Consistency on Sparse Sampled Inputs

Sagi Snir¹ and Raphael Yuster²

¹ Dept. of Evolutionary Biology, University of Haifa,
Haifa 31905, Israel

² Dept. of Mathematics, University of Haifa,
Haifa 31905, Israel

Abstract. Phylogenetic tree reconstruction is a fundamental biological problem. Quartet amalgamation - combining a set of trees over four taxa into a tree over the full set - stands at the heart of many phylogenetic reconstruction methods. However, even reconstruction from a *consistent* set of quartet trees, i.e. all quartets agree with some tree, is NP-hard, and the best approximation ratio known is $1/3$. For a dense input of $\Theta(n^4)$ quartets (not necessarily consistent), the problem has a polynomial time approximation scheme.

When the number of taxa grows, considering such dense inputs is impractical and some sampling approach is imperative. In this paper we show that if the number of quartets sampled is at least $\Theta(n^2 \log n)$, there is a randomized approximation scheme, that runs in linear time in the number of quartets. The previously known polynomial approximation scheme for that problem required a very dense sample of size $\Theta(n^4)$. We note that samples of size $\Theta(n^2 \log n)$ are sparse in the full quartet set.

1 Introduction

The study of evolution and the construction of phylogenetic (evolutionary) trees are classical subjects in biology. Existing accurate phylogenetic techniques are capable of coping with a relatively small amount of data. DNA sequences from a variety of organisms are rapidly accumulating, challenging current approaches of phylogenetics. The *supertree* approach works by constructing small trees over overlapping sets of taxa, and subsequently, amalgamating these trees into a big tree over the full set.

We distinguish between *rooted* and *unrooted* phylogenetic trees. In the rooted setting a rooted triplet tree (Figure 1a) is the basic informative unit. We denote a triplet over the taxa a, b, c by $ab|c$ meaning that, in the underlying tree, the least common ancestor of a and b ($lca(a, b)$) is a descendant of $lca(a, c) = lca(b, c)$. Given a set of rooted triplets, there exists a polynomial time algorithm that constructs a tree consistent with the given set, or reports that no such tree exists [13]. In the unrooted setting, the notion of lca is meaningless and therefore the basic informative unit is a quartet tree (Figure 1b) - $ab|cd$ - meaning that there

is a path in the underlying tree separating a and b from c and d . Here however, the decision problem of whether there exists a tree satisfying all the quartets in an arbitrary given set, is NP-hard [7]. This raises the problem of finding a tree maximizing the number of consistent quartets - *maximum quartet consistency* (MQC) [7].

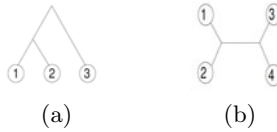


Fig. 1. (a) - a rooted triplet tree 12|3. (b) - an unrooted quartet tree 12|34.

The MQC problem is central in many phylogenetic problems that reduce to solving MQC at some stage (see [5], Chapter 6, for an introduction). The approximation complexity of MQC is a longstanding open problem. At present, the best known polynomial time approximation algorithm has an approximation ratio of $1/3$ (see Section 1.1). There are also a few results that assume some constraint either on the correctness or the density of the input. Most notably, Jiang et al. [4] designed a polynomial time approximation scheme (PTAS) for MQC when the input consists of all $\binom{n}{4}$ possible quartets. This was later generalized by the authors in [6] for inputs of size $\Theta(n^4)$.

The requirement that the input consists of $\Theta(n^4)$ quartets as in [4,6] becomes prohibitive when the number of taxa grows even to moderate sizes. A faster approach is to sample a relatively small number of $m \ll \binom{n}{4}$ four-taxa sets, providing as input the corresponding m quartets they define, and try to solve MQC on this input. This version of the problem is *sampled-MQC*.

In a recent paper [6], the authors have devised a new polynomial time approximation algorithm for sampled-MQC. Given a set of m quartets sampled uniformly from the set of all $\binom{n}{4}$ quartets, the algorithm achieves an approximation ratio of roughly 0.425. The result is obtained by constructing a *quartet graph* and approximating a MaxCut in that graph.

The main result of this paper is that sampled-MQC admits a linear time randomized approximation scheme for sparse inputs. We prove that already for $m = \Theta(n^2 \log n)$, sampled-MQC admits an EPRAS (efficient polynomial time randomized approximation scheme [8]) that runs in $O(m)$ time. In other words, we compute, in $O(m)$ time, an n -taxa phylogenetic tree that satisfies, with high probability, at least $(1 - \epsilon)m$ of the input quartets. This is an improvement over the input density of the other PTAS algorithms of [4,6], but at the cost of assuming a uniformly sampled input. It also improves very significantly over the 0.425 previous approximation, but at the cost of $\Omega(n^2 \log n)$ quartets. Our algorithm also allows for the possibility that a small fraction (smaller than ϵ) of the supplied sampled quartets are erroneous.

1.1 Preliminaries

An (unrooted) phylogenetic tree is a tree whose internal vertices each have degree 3, and whose leaves are labeled by some taxa set (representing existing species). Throughout this paper all trees are phylogenetic trees, unless stated otherwise. For a tree T , we denote by $\mathcal{L}(T)$ the taxa set corresponding to the leaves of T .

Let T be a tree and $A \subseteq \mathcal{L}(T)$ a subset of the leaves of T . We denote by T_A , the subtree of T induced by A . Namely, T_A is the tree obtained from T by removing all leaves in $\mathcal{L}(T) \setminus A$ and paths leading exclusively to them, and subsequently internal vertices with degree two are contracted.

For two trees T and T' , we say that T' is *satisfied* by T , if $\mathcal{L}(T') \subseteq \mathcal{L}(T)$ and $T_{\mathcal{L}(T')} = T'$. Otherwise, T' is *violated* by T . For a set of trees $\mathcal{T} = \{T_1, \dots, T_k\}$ with possibly overlapping leaves, we denote by $\mathcal{T}_s(T)$ the set of trees in \mathcal{T} that are satisfied by T . We say that \mathcal{T} is *consistent* if there exists a tree T^* over the set of leaves $\cup_i \mathcal{L}(T_i)$ that satisfies every tree $T_i \in \mathcal{T}$. Otherwise, \mathcal{T} is *inconsistent*. When \mathcal{T} is inconsistent, it is desirable to find a tree T^* over $\cup_i \mathcal{L}(T_i)$ that maximizes some objective function. T^* is called a *supertree* and the problem of finding T^* is the *supertree problem*.

A *quartet* tree (or simply *quartet*), is an undirected phylogenetic tree over four leaves $\{a, b, c, d\}$. We write a quartet over $\{a, b, c, d\}$ as $ab|cd$ if the removal of the unique edge connecting the two internal vertices partitions the quartet into two components, one containing a, b and the other containing c, d . An important case of the supertree problem is when the set of input trees is a set of quartet trees \mathcal{Q} and the task is to find a tree T such that $|\mathcal{Q}_s(T)|$ is maximized. The problem is denoted as *maximum quartet consistency* (MQC). We note that MQC is NP-hard even when \mathcal{Q} is consistent [7].

Every tree T with $|\mathcal{L}(T)| = n$ defines its full quartet set, $\mathcal{Q}(T)$. Since any four leaves define a unique quartet satisfied by T , we always have $\mathcal{Q}(T) = \binom{n}{4}$. Consider the following trivial approximation algorithm for MQC. Take any tree T^* with n leaves, and randomly label them with the elements of $\mathcal{L}(T)$. As any four leaves a, b, c, d define one of three possible quartets (either $ab|cd$ or $ac|bd$ or $ad|bc$), only one of which is satisfied by T , we have that T^* satisfies an expected number of $1/3$ of the input quartet set. Surprisingly, no algorithm is known that improves upon the naive $1/3$ approximation, although the problem has been raised over two decades ago. In fact, even if we are guaranteed that the input \mathcal{Q} satisfies $\mathcal{Q} \subset \mathcal{Q}(T)$ (namely, we are guaranteed that the optimal solution to MQC is $|\mathcal{Q}|$), no algorithm is known to achieve an outcome that is asymptotically better than $|\mathcal{Q}|/3$. As mentioned in the introduction, the MQC problem has a PTAS when $|\mathcal{Q}| = \Theta(n^4)$ [4,6].

We now turn to sampled MQC. As described in the introduction, we know the set of taxa $\mathcal{L}(T)$ of some unknown tree T , and given any four taxa we can (using biological information) infer the correct quartet. Clearly, if we have unlimited time and resources, we can generate all $\binom{n}{4}$ elements of $\mathcal{Q}(T)$ and solve the problem. This, however, is unrealistic for very large n .

Motivated by this problem, sampled-MQC consists of an input \mathcal{Q} of $m \ll \binom{n}{4}$ quartets sampled uniformly (say, with replacement), from $\mathcal{Q}(T)$. Recently, the

authors [6] obtained an approximation algorithm for sampled-MQC that improves upon the naive $1/3$ approximation. They describe a randomized approximation algorithm that constructs a tree T^* satisfying an expected number of more than $0.425m$ elements of \mathcal{Q} .

The main result of this paper is a significant strengthening of the result of [6], in case the sample size is at least $\Theta(n^2 \log n)$. Notice that such a sample is very sparse in $\mathcal{Q}(T)$, as the size of the latter is $\binom{n}{4}$. We construct a linear time randomized approximation scheme for sampled-MQC. The exact statement of our result follows.

Theorem 1. *Let $\epsilon > 0$ be fixed. Suppose that \mathcal{Q} is a set of $m \geq \Theta(n^2 \log n)$ quartets sampled uniformly from the full quartet set $\mathcal{Q}(T)$ of some unknown tree T . Then there is an $O(m)$ time randomized algorithm that constructs a tree T^* that satisfies an expected number of $(1 - \epsilon)m$ elements of \mathcal{Q} .*

As mentioned earlier, the proof of Theorem 1 is slightly more general. One does not need to assume that all m sampled quartets are correct. The proof stays intact as long as each sampled quartet is erroneously generated with small probability (smaller than ϵ). The general (extremely high level) idea of the algorithm is to consider relatively small *models* of phylogenetic trees. By scanning constantly many such models, we prove that one of them is guaranteed to be a model M of our unknown tree T . Given that model M , we prove how to expand it to a tree T^* which is also modeled by M . We prove that with small *constant* probability, T^* satisfies many quartets of \mathcal{Q} . By repeating this process constantly many times, we eventually obtain, with high probability, a tree T^* which indeed satisfies many quartets of \mathcal{Q} .

The rest of this paper is organized as follows. In the next section we state and prove several notions and lemmas that are useful for the description of the algorithm. The algorithm and its proof of correctness are given in Section 3.

2 Tree Models

For the remainder of this paper, we assume that T is some (unknown) phylogenetic tree with n leaves. The leaves are labeled by a set $\mathcal{L}(T)$ of known labels, and $\mathcal{Q} \subset \mathcal{Q}(T)$ is a set of quartets obtained by sampling uniformly (with replacement) m elements of $\mathcal{Q}(T)$. The error parameter $\epsilon > 0$ is fixed, and we assume that $m \geq Cn^2 \log n$ where C is some suitably large constant depending on ϵ . Our goal is to construct a tree T^* that satisfies, with high probability, at least $(1 - \epsilon)m$ quartets of \mathcal{Q} .

2.1 Tree Models of Constant Size

Since all internal vertices of T have degree 3, it is a well-known easy fact (see, e.g., [5]) that there is always an edge of T , whose removal partitions T into two components, each having at least $n/3$ of the leaves. So assume $e = (x, y)$ is such an edge of T . It will be convenient to view T as a *rooted* tree. Subdivide e by

introducing a new vertex r in its middle and make r the root. Hence, now T is a full binary tree, the children of r are x and y , and each of them is an ancestor of at least $n/3$ leaves. Unless otherwise specified, we refer to this rooted version of T .

Let $\mathcal{I}(T)$ denote the set of internal vertices of T (including the root r), and notice that since $|\mathcal{L}(T)| = n$, and T is a full binary tree, then we have $|\mathcal{I}(T)| = n - 1$.

Recall that $lca(x, y)$ denotes the unique lowest common ancestor of two vertices x, y in a rooted tree. In particular, if x is an ancestor of y then $lca(x, y) = x$. We say that a subset $K \subset \mathcal{I}(T)$ is *LCA-closed* if whenever $x, y \in K$ then also $lca(x, y) \in K$. We further demand that $r \in K$. It is a straightforward exercise that every set of k vertices of a rooted tree can be made LCA-closed by adding to it at most k additional vertices.

There is a natural correspondence between an LCA-closed subset K , a topological minor of T it defines, and a partition it defines on $\mathcal{L}(T)$. We now state this correspondence formally.

Definition 1. *Let $K \subset \mathcal{I}(T)$ be LCA-closed. Let $M_T(K)$ be the rooted binary tree whose vertex set is K , and u is the parent of v in $M_T(K)$ if and only if u is the lowest among all proper ancestors of v in K . We call $M_T(K)$ a tree model of T .*

Notice that r is the root of $M_T(K)$, since it is the only vertex of K with no proper ancestor in K . Observe also that $M_T(K)$ is a contraction (in fact, a topological minor) of T .

For $v \in K$, let $A_v \subset \mathcal{L}(T)$ denote the set of leaves of T having the property that $x \in A_v$ if and only if v is the lowest among all proper ancestors of x in K . Let v_0 and v_1 be the two children of v in T (and notice that v_0, v_1 are not necessarily in K and may or may not be in $\mathcal{L}(T)$). Then A_v is further divided into two parts, $A_{v,0}$ are those leaves that have v_0 as their ancestor while $A_{v,1}$ have v_1 as their ancestor.

Definition 2. *The set $\mathcal{P}_T(K) = \{A_{v,0}, A_{v,1} \mid v \in K\}$ is the leaf partition of the model.*

Notice that $\mathcal{P}_T(K)$ is a partition of $\mathcal{L}(T)$ into $2|K|$ parts. It may be the case that some parts are empty; for example, if $v \in K$ and its child $v_0 \in K$ then $A_{v,0} = \emptyset$.

Definition 3. *We say that $M_T(K)$ is a δ -model of T if every element of $\mathcal{P}_T(K)$ has size at most δn .*

The next lemma proves that there are δ -models with $O(1/\delta)$ vertices.

Lemma 1. *There is a δ -model of T with at most $4/\delta$ vertices.*

Proof. We prove that there exists a (not necessarily LCA-closed) subset $K' \subset \mathcal{I}(T)$ with $r \in K'$, so that for each $v \in K'$, the set of leaves A_v has $\delta n/2 \leq |A_v| \leq \delta n$. (Notice that since $r \in K'$ then A_v is well-defined even if K' is

not LCA-closed.) Since $\mathcal{P}' = \mathcal{P}_T(K')$ is a partition of $\mathcal{L}(T)$, this implies that $|K'| \leq n/(\delta n/2) = 2/\delta$. Now, since K' can be made into an LCA-closed set K by adding to K' at most $|K'|$ additional vertices, we have that $|K| \leq 4/\delta$. Since $\mathcal{P} = \mathcal{P}_T(K)$ is a refinement of \mathcal{P}' , then every element of \mathcal{P} also has size at most δn , and the lemma follows.

We prove the existence of K' as follows. We initially set all the leaves in $\mathcal{L}(T)$ as *unmarked*. Next, we perform a postorder traversal of T . Whenever we reach a vertex $v \in \mathcal{I}(T)$, let U_v denote the set of yet unmarked leaves in the subtree rooted at v . If $|U_v| \geq \delta n/2$ then we add v to K' , mark all elements of U_v , and notice that $A_v = U_v$. Notice that we must have $|U_v| \leq \delta n$. Otherwise, one of the two children of v , say w , would have had at least $\delta n/2$ unmarked leaves in U_w . But since w has already been traversed, we should have already added w to K' and marked all elements of U_w , a contradiction. Finally, when we reach r we add it to K' in any case. □

2.2 Constant Size Nets for Constant Size Models

Let $M_T(K)$ be a δ -model, and let $\mathcal{P}_T(K)$ be its leaf partition, consisting of subsets $A_{v,j}$ for $v \in K$ and $j = 0, 1$.

Definition 4. *The function $f_T(K) : K \times \{0, 1\} \rightarrow [0, \delta]$ defined by $f_T(K)(v, j) = |A_{v,j}|/n$ is called the size vector of the model.*

We say that a function $f' : K \times \{0, 1\} \rightarrow [0, \delta]$ is a δ^4 -approximation of $f_T(K)$ if $f'(v, j) \leq f_T(K)(v, j) \leq f'(v, j) + \delta^4$ for all $v \in K$ and $j = 0, 1$.

Given $|K|$ and δ , a family of functions \mathcal{F} is called a $(|K|, \delta^4)$ -net if for every possible function $f : K \times \{0, 1\} \rightarrow [0, \delta]$ one can find in \mathcal{F} a δ^4 -approximation of f .

For constants $|K|$ and δ , it is not difficult to construct a $(|K|, \delta^4)$ -net of constant size, and in constant time. This is analogous to constructing the set of all vectors of length $2|K|$ whose coordinates are of the form $i\delta^4$ for $i = 0, \dots, \lfloor \delta^{-3} \rfloor$. As there are at most $(1 + \delta^{-3})^{2|K|}$ such vectors, the claim follows.

3 Proof of the Main Result

In our proof we will use $\delta = \epsilon/5000$. For the *proof* of our algorithm we need to fix and reference the following objects.

1. A rooting of T from some vertex r as described in Section 2.1. Recall that this makes T into a full binary tree, and each child of r is an ancestor of at least $n/3$ leaves.
2. A δ -model $M_T(K)$ of T with at most $4/\delta$ vertices, guaranteed to exist by Lemma 1. Label the vertices of $M_T(K)$ with $\{1, \dots, |K|\}$.
3. The leaf partition $\mathcal{P}_T(K)$ of the model $M_T(K)$. Recall that $\mathcal{P}_T(K)$ is a partition of $\mathcal{L}(T)$ into $2|K|$ parts, denoted by $A_{v,j}$ for $v \in K$ and $j = 0, 1$.
4. The size vector $f_T(K)$ of the model. Recall that $f_T(K)(v, j) = |A_{v,j}|/n$.

Generally speaking, our algorithm will “guess” a δ -model, it will “guess” a close approximation of $f_T(K)$, and it will try to generate a partition of $\mathcal{L}(T)$ defined by that guess. We will show that after guessing constantly many times (constant depending on δ and hence on ϵ) we are guaranteed that one of our guesses is correct. We will then show that, under the assumption of a correct guess, the generated partition is *close* to the actual partition $\mathcal{P}_T(K)$. We will then show how to construct a phylogenetic tree T^* using the generated partition, which satisfies many quartets of \mathcal{Q} .

We describe a general procedure $\text{PARTITION}(M, f)$. The first parameter is a labeled binary tree M with k vertices, and the second parameter is a function $f : \{1, \dots, k\} \times \{0, 1\} \rightarrow [0, \delta]$. We call PARTITION constantly many times. For all $k = 1/\delta, \dots, 4/\delta$ we construct a (k, δ^4) -net \mathcal{F}_k . We also construct the set \mathcal{M}_k of all possible labeled binary tree with k vertices. There are constantly many such trees. For each k , for each $f \in \mathcal{F}_k$ and for each $M \in \mathcal{M}_k$ we call $\text{PARTITION}(M, f)$. Hence, at some point we are *guaranteed* to call it with parameters (M_0, f_0) where M_0 is label-isomorphic to $M_T(K)$ and f_0 is a δ^4 -approximation of $f_T(K)$.

What follows is a description of PARTITION *assuming* that the parameters are instantiated by (M_0, f_0) . Its behavior in other calls is of no interest to us (it may return “fail” or a partition that is not close to $\mathcal{P}_T(K)$).

3.1 PARTITION(M_0, f_0)

PARTITION tries, using f_0 and M_0 , to construct a partition \mathcal{P}^* of $\mathcal{L}(T)$ that is *close* (in a well defined sense) to $\mathcal{P}_T(K)$. We will show that with *constant positive probability*, it is guaranteed to succeed.

The main problem, of course, is that although we have M_0 and f_0 (and thus we assume from now on that we know $M_T(K)$ and have a good approximation of $f_T(K)$), we do *not know* the actual leaf partition $\mathcal{P}_T(K)$. However, we *do know* a close approximation of the *cardinality* of each element of $\mathcal{P}_T(K)$, since f_0 is close to $f_T(K)$. We define:

1. $y_{v,j}$ to be the child of v in T that is the ancestor of all elements of $A_{v,j}$;
2. $S_{v,j} \subset \mathcal{L}(T)$ to be all the leaves that have $y_{v,j}$ as their ancestor. Notice that $A_{v,j} \subset S_{v,j}$.

We say that (u, i) is *below* (v, j) (or, equivalently, denote that $(u, i) \leq (v, j)$) if v is an ancestor of u . We will perform a postorder traversal of the (v, j) . Namely, when we process (v, j) we already know that all (u, i) properly below it have been processed. Since M_0 is label isomorphic to $M_T(K)$, a postorder traversal of M_0 achieves this. Whenever we reach a vertex v of M_0 we process both $(v, 0)$ and $(v, 1)$.

Definition 5. *We say that a partition \mathcal{P}^* of $\mathcal{L}(T)$ is close to $\mathcal{P}_T(K)$ if the following conditions hold.*

1. $\mathcal{P}^* = \{B_{v,j} \mid v \in K, j = 0, 1\} \cup \{B^*\}$. We call B^* the exceptional part. Hence, $B^* = \mathcal{L}(T) \setminus \bigcup_{(v,j) \in K \times \{0,1\}} B_{v,j}$.

2. $|A_{v,j} \setminus B_{v,j}| \leq 50\delta^2 n$.
3. $B_{v,j} \subset S_{v,j}$.
4. $|S_{v,j} \setminus \bigcup_{(u,i) \leq (v,j)} B_{u,i}| \leq 50\delta^2 n$.

In fact, notice that the second requirement (which is the one we are after) is actually a consequence of the third and fourth requirements. We will show how to construct, with constant positive probability, a partition \mathcal{P}^* that is close to $\mathcal{P}_T(K)$.

We assume that whenever we reach (v, j) , we have already defined sets $B_{u,i}$ for all (u, i) that are properly below (v, j) , and, furthermore, the sets already defined satisfy the desired properties. We will show how to define $B_{v,j}$ so that with *constant* probability, it also satisfies the properties above. Since the number of possible (v, j) is only $2|K|$, this will yield that with constant positive probability, the constructed \mathcal{P}^* is close to $\mathcal{P}_T(K)$.

So assume that we have reached step (v, j) in our postorder traversal and wish to construct $B_{v,j}$. Consider the set of leaves $X = S_{v,j} \setminus \bigcup_{(u,i) < (v,j)} B_{u,i}$. Namely, X consists of the elements of $A_{v,j}$ together with all other elements of $S_{v,j}$ that have not been assigned to sets $B_{u,i}$. Although the algorithm does not *know* the set X (since it does not know $S_{v,j}$), it does know a good approximation for its cardinality. Since f_0 is a δ^4 -approximation, we know each $|A_{u,i}|$ up to $\delta^4 n$. As there are at most $2|K| \leq 8/\delta$ possible (u, i) , the overall error in estimating $|S_{v,j}|$ is $8\delta^3 n$. Hence, we know $|X|$ up to an error of $8\delta^3 n$.

The first case to look at is when our estimate for $|X|$ is less than $49\delta^2 n$. In particular, we are guaranteed that $|X| \leq 49\delta^2 n + 8\delta^3 n \leq 50\delta^2 n$. In this case, we simply define $B_{v,j} = \emptyset$. Notice that since X contains $A_{v,j}$, this still satisfies the conditions required of $B_{v,j}$.

So, we may now assume that $|X| \geq 49\delta^2 n$. Now, consider the tree T_X whose root is $y_{v,j}$ and whose leaf set is X . Again, the algorithm does not know T_X , but it can guess, with constant positive probability, some important information regarding its structure.

Each vertex t of T_X , when removed from T_X , partitions $T_X - t$, and hence also partitions X , into three parts (some of which may be empty). One part is the component containing the parent of t (if $t = y_{v,j}$ then this part is empty). The two other parts contain each a child of t (if t does not have two children then these parts could possibly be empty). So, denote the corresponding partition of X by $X_0(t), X_1(t), X_2(t)$ where $X_0(t)$ are the leaves of the part of $T_X - t$ that contain $y_{v,j}$. In particular $X_0(y_{v,j}) = \emptyset$ and if $t \in X$ (namely t a leaf of T_X) then $X_1(t) = X_2(t) = \emptyset$ while $X_0(t) = X - t$.

Lemma 2. *There exists $t \in T_X$ for which $|X_0(t)| \leq 16\delta^2 n$ but $|X_0(t) \cup X_1(t)| > 16\delta^2 n$ and $|X_0(t) \cup X_2(t)| > 16\delta^2 n$.*

Proof. Let t be the furthest vertex from $y_{v,j}$ in T_X for which $|X_0(t)| \leq 16\delta^2 n$. Clearly t is not a leaf of T_X since for leaves we have $|X_0(t)| = |X| - 1 \geq 49\delta^2 n - 1 > 48\delta^2 n$. Also, t must have two children in T_X since otherwise, if t_1 is its only child then $X_0(t_1) = X_0(t)$, and t_1 is further than t from $y_{v,j}$. So, let t_1 and t_2 be the two children of t , where t_j belongs to the subtree of $T_X - t$

whose leaves are $X_j(t)$ for $j = 1, 2$. If $|X_0(t) \cup X_1(t)| \leq 16\delta^2 n$ then observe that since $X_0(t_2) = X_0(t) \cup X_1(t)$, then t_2 would have been furthest. Therefore, $|X_0(t) \cup X_1(t)| > 16\delta^2 n$, and analogously, $|X_0(t) \cup X_2(t)| > 16\delta^2 n$. \square

We call a vertex t satisfying Lemma 2 a *center* of T_X . So fix some center t , and consider the cardinalities, $|X_j(t)| = \alpha_j n$ for $j = 0, 1, 2$. Our algorithm guesses values α_j^* for $j = 0, 1, 2$ that approximate the α_j . We say that the guess is *valid* if $|\alpha_j^* - \alpha_j| \leq \delta^3$ for $j = 0, 1, 2$. Clearly, with constant positive probability (polynomial in the constant $1/\delta$) we arrive at a valid guess.

So we may now assume that the α_j^* are a valid guess for $j = 0, 1, 2$. Now, if $\alpha_1^* \geq 16\delta^2$, we also guess a leaf $w_1 \in X_1(t)$. The probability that we have guessed correctly is at least $16\delta^2 - \delta^3$, hence a constant positive probability. Similarly, if $\alpha_2^* \geq 16\delta^2$, we guess a leaf $w_2 \in X_2(t)$. So, assume that we have guessed correctly.

Now there are three cases to consider. The first case is when $\alpha_1^* < 16\delta^2$. The second case is when $\alpha_2^* < 16\delta^2$. The third case is when both are at least $16\delta^2$. Since the first and second case are symmetric, we consider, without loss of generality, only the first case and third case.

Before describing these cases, we fix some notation. Let $B = \bigcup_{(u,i) < (v,j)} B_{u,i}$. Let $D = \mathcal{L}(T) - S_{v,j}$. Namely, D is the set of leaves that do not have $y_{v,j}$ as their ancestor. In particular, D contains at least one third of the leaves of T , so $|D| \geq n/3$. Notice that D, B, X are pairwise disjoint and $D \cup B \cup X = \mathcal{L}(T)$. Consider a sample of $C'n^2 \log n$ quartets where C' is a sufficiently large constant. Eliminate from this sample all quartets that contain an element of B . As $|D| \geq n/3$, we still remain with a completely random sample Q of $q \geq Cn_0^2 \log n_0$ quartets over $D \cup X$ where $n_0 = |D \cup X| > n/3$, and C is a suitably large constant. Notice that for two leaves $a, b \in D \cup X$, the probability that they appear in a specific element of Q is *precisely* $p = \frac{12}{n_0(n_0-1)}$.

For simplicity, denote $|D| = \eta n_0$, $|X_i(t)| = \alpha_i n = \beta_i n_0$. Observe that

$$\eta + \beta_0 + \beta_1 + \beta_2 = 1 . \tag{1}$$

Finally, let $\beta_i^* = \alpha_i^* n/n_0$, and observe that as $n/n_0 < 3$ we have $|\beta_i - \beta_i^*| \leq 3\delta^3$.

The case $\alpha_1^* < 16\delta^2$. Since $|X| \geq 49\delta^2 n$, we have that $\alpha_0 + \alpha_1 + \alpha_2 \geq 49\delta^2$. As t is a center we have $\alpha_0 \leq 16\delta^2$. Since $|\alpha_j^* - \alpha_j| \leq \delta^3$ we surely have $\alpha_2^* > 16\delta^2$ so we have guessed $w_2 \in X_2(t)$.

We will show that, with high probability, we can construct a set $B_{v,j}$ that contains all the elements of $X_2(t)$, and that is contained in X . Notice that such a $B_{v,j}$ misses at most $|X_1(t) \cup X_0(t)| < 49\delta^2 n$ vertices of X , and in particular satisfies the requirements in the definition of a close partition.

We will show that for a $z \in D \cup X$, we can, with high probability, *differentiate* between the case $z \in D$ and the case $z \in X_2(t)$. For those $z \in X_0(t) \cup X_1(t)$ we will not be able to differentiate. Using this differentiation, we can find a subset $B_{v,j}$ so that $X_2(t) \subset B_{v,j} \subset X$, as required.

We will use w_2 as a pivot for our differentiation rule. Consider first the case $z \in D$. Given that w_2 and z are in the same quartet of Q , what is the probability that they are on opposite sides? Let a, b denote the other two elements of the quartet. Clearly, if $a \in X_2(t)$ and $b \notin X_2(t)$ then the quartet must be $aw_2|zb$. Similarly, if $b \in X_2(t)$ and $a \notin X_2(t)$ then the quartet must be $bw_2|za$. Also, if $a \in D$ and $b \in X_0(t) \cup X_1(t)$ we must have $bw_2|za$. Similarly, if $b \in D$ and $a \in X_0(t) \cup X_1(t)$ we must have $aw_2|zb$. It follows that, given that w_2 and z are in the same quartet of Q , they are on opposite sides with probability *at least*

$$2\beta_2\eta + 2\beta_2(\beta_0 + \beta_1) + 2\eta(\beta_0 + \beta_1) - o_n(1) = 2\beta_2 - 2\beta_2^2 + 2\eta(\beta_0 + \beta_1) - o_n(1)$$

where the r.h.s. uses (II) (the term $o_n(1)$ is due to the fact that a and b are sampled *without* replacement, as they must be distinct). Hence, the expected number of elements of Q in which w_2, z are on opposite sides is greater than

$$qp(2\beta_2 - 2\beta_2^2 + 2\eta(\beta_0 + \beta_1) - \delta^2/4) .$$

But $q \geq Cn_0^2 \log n_0$ and hence $qp > C \log n$. Since each element of Q is sampled uniformly and independently, we have, using a standard large deviation inequality (see [2], Corollary A.1.14), that the probability of deviating from the expectation by more than $qp\delta^2/4$ is smaller than $1/(9n)$, for a suitably large constant C . Hence, by the union bound, with probability at least $8/9$, for all $z \in D$ we have that the number of elements of Q in which w_2, z are on opposite sides is at least

$$qp(2\beta_2 - 2\beta_2^2 + 2\eta(\beta_0 + \beta_1) - \delta^2/2) . \tag{2}$$

Consider next the case $z \in X_2(t)$. As in the previous case, once can show that with probability at least $8/9$, for all $z \in X_2(t)$, the number of elements of Q in which w_2, z are on opposite sides is at most

$$qp(2\beta_2 - \beta_2^2 + \delta^2/2) . \tag{3}$$

Thus, our differentiation rule is the following. Given $z \in X \cup D$, we count the number of quartets of Q in which w_2, z are in opposite sides. If this number is at most $qp(2\beta_2^* - (\beta_2^*)^2 + 6\delta^3 + \delta^2/2)$ then we place z in $B_{v,j}$. Otherwise, we don't.

Lemma 3. *Let $B_{v,j}$ consist of all $z \in X \cup D$ for which the number of quartets containing w_2 and z in opposite sides is at most $qp(2\beta_2^* - (\beta_2^*)^2 + 6\delta^3 + \delta^2/2)$. Then, with probability at least $7/9$ we have $X_2(t) \subset B_{v,j} \subset X$.*

Proof. Recall that $|\beta_2^* - \beta_2| \leq 3\delta^3$. Hence,

$$qp(2\beta_2 - \beta_2^2 + \delta^2/2) \leq qp(2\beta_2^* - (\beta_2^*)^2 + 6\delta^3 + \delta^2/2) .$$

Thus, by (3), with probability at least $8/9$, we have that $X_2(t) \subset B_{v,j}$.

It remains to prove that with probability at least $8/9$ we have that $B_{v,j} \subset X$, or, equivalently, that $B_{v,j} \cap D = \emptyset$. By (2) it suffices to prove that

$$2\beta_2 - 2\beta_2^2 + 2\eta(\beta_0 + \beta_1) - \delta^2/2 > 2\beta_2^* - (\beta_2^*)^2 + 6\delta^3 + \delta^2/2 .$$

As $|\beta_2^* - \beta_2| \leq 3\delta^3$ it suffices to prove that $2\beta_2 - 2\beta_2^2 + 2\eta(\beta_0 + \beta_1) - \delta^2/2 > 2\beta_2 - \beta_2^2 + 12\delta^3 + \delta^2/2$ which is equivalent to showing that

$$2\eta(\beta_0 + \beta_1) - \beta_2^2 > \delta^2 + 12\delta^3. \quad (4)$$

Recall that $\eta = |D|/n_0$ and $|D| \geq n/3$ so $\eta \geq 1/3$. As t is a center we have, by Lemma 2, that $\alpha_0 + \alpha_1 \geq 16\delta^2$ so $\beta_0 + \beta_1 > 16\delta^2$. Since $|A_{v,j}| \leq \delta n$ and since X consists of $A_{v,j}$ and at most $50\delta^2 n$ additional vertices from sets below (v, j) we have, in particular, that $\alpha_2 \leq \delta + 50\delta^2$. Thus, $\beta_2 \leq 3\delta + 150\delta^2 < 3.1\delta$. Hence the l.h.s. of (4) is at least $\frac{2}{3} \cdot 16\delta^2 - 9.61\delta^2 > 1.056\delta^2$, proving (4). \square

The case $\alpha_1^* \geq 16\delta^2$ and $\alpha_2^* \geq 16\delta^2$. In this case we have selected $w_1 \in X_1(t)$ and $w_2 \in X_2(t)$. As in Section 3.1 one can show that, with high probability, we can construct a set $B_{v,j}$ that contains all the elements of $X_1(t) \cup X_2(t)$, and that is contained in X . Notice that such a $B_{v,j}$ misses at most $|X_0(t)| \leq 16\delta^2 n$ vertices of X , and in particular satisfies the requirements in the definition of a close partition. Due to space limitations, we omit these details.

3.2 Constructing a Tree from a Close Partition

In the previous section we have proved that PARTITION, when called with the parameters (M_0, f_0) , constructs, with small constant probability, a partition \mathcal{P}^* of $\mathcal{L}(T)$ that is close to $\mathcal{P}_T(K)$. Hence, if we run PARTITION(M_0, f_0) constantly many times, we are guaranteed that, with high probability, it will construct a \mathcal{P}^* that is close to $\mathcal{P}_T(K)$. To complete the proof of Theorem 1, it suffices to show that with high probability, a \mathcal{P}^* that is close to $\mathcal{P}_T(K)$ can be used to construct a tree T^* that satisfies a fraction of $(1 - \epsilon)$ elements of a random sample of size at least $Cn^2 \log n$.

So, for the remainder of this section we assume that \mathcal{P}^* is close to $\mathcal{P}_T(K)$. Recall that $\mathcal{P}^* = \{B_{v,j} \mid v \in K, j = 0, 1\} \cup B^*$. For each $B_{v,j}$ we construct a tree $T_{v,j}$ as follows. $T_{v,j}$ is an arbitrary rooted full binary tree, except for the root which has a unique child, and whose set of leaves is precisely $B_{v,j}$. In the event that $B_{v,j} = \emptyset$ then we also define $T_{v,j}$ to be an empty tree. Notice that $T_{v,j}$ has precisely $2|B_{v,j}|$ vertices.

We construct a tree T^* by attaching to M_0 the $2|K|$ trees $T_{v,j}$ at appropriate places as follows. There are three cases. If $B_{v,j} = \emptyset$ we do nothing with $T_{v,j}$ as it is an empty tree. So assume that $B_{v,j} \neq \emptyset$. If v is a leaf of M_0 then we attach $T_{v,j}$ to M_0 by identifying the root of $T_{v,j}$ with v . (Notice that both trees $T_{v,0}$ and $T_{v,1}$ are attached at v so v has two children in T^*). If v is an internal vertex of M_0 , then it has two emanating edges, e_0 and e_1 corresponding to $(v, 0)$ and $(v, 1)$. We subdivide the edge e_j introducing a new vertex and identify this new vertex with the root of $T_{v,j}$.

Notice that, considered as an unrooted tree (we can simply put an edge connecting the two children of the root of T^* , which is also the root of M_0 , and eliminate the root, thereby making T^* unrooted), T^* is a phylogenetic tree. Internal vertices have degree 3, and, furthermore $\mathcal{L}(T^*) = \mathcal{L}(T) - B^*$. We now prove that, with high probability, T^* satisfies a large fraction of the input quartet set.

Lemma 4. *For a random sample of m quartets, the expected number of quartets satisfied by T^* is at least $m(1 - \epsilon/3)$. Hence, with probability at least $2/3$ we have that T^* satisfies at least $(1 - \epsilon)m$ quartets.*

Proof. By linearity of expectation, it suffices to prove that a single randomly sampled quartet is satisfied by T^* with probability at least $1 - \epsilon/3$.

Let $E_{v,j} = A_{v,j} \cap B_{v,j}$. Call the sets $E_{v,j}$ the *essential sets*. The construction of T^* guarantees that if a, b, c, d are in pairwise *distinct* essential sets then the quartet they induce in T is identical to the quartet they induce in T^* . On the other hand, if one of a, b, c, d is not in an essential set, or if two of them are in the same essential set, this need not be the case.

Now, observe first that since \mathcal{P}^* is close to $\mathcal{P}_T(K)$ then we have $|E_{v,j}| \geq |A_{v,j}| - 50\delta^2 n$. As there are $2|K| \leq 8/\delta$ possible sets $A_{v,j}$ we have that

$$\left| \bigcup_{(v,j) \in K \times \{0,1\}} E_{v,j} \right| \geq \left| \bigcup_{(v,j) \in K \times \{0,1\}} A_{v,j} \right| - 400\delta n = n(1 - 400\delta).$$

Thus, a randomly chosen leaf is not in an essential set with probability at most 400δ . What is the probability that two leaves of a randomly sampled quartet $ab|cd$ are in the same part of $\mathcal{P}_T(K)$? As each part contains at most a δ fraction of the leaves, this probability is at most δ . As there are 6 pairs in a, b, c, d , with probability at least $1 - 6\delta$ each leaf of $ab|cd$ is in a distinct part. Overall, using $\delta = \epsilon/5000$, we have that with probability at least $1 - 6\delta - 4 \cdot (400\delta) > 1 - \epsilon/3$ each element of a randomly sampled quartet is in a distinct essential set. Hence, a randomly sampled quartet is also a quartet of T^* with probability at least $1 - \epsilon/3$. \square

References

1. Aho, A.V., Sagiv, Y., Szymanski, T.G., Ullman, J.D.: Inferring a tree from lowest common ancestors with an application to the optimization of relational expressions. *SIAM Journal on Computing* 10(3), 405–421 (1981)
2. Alon, N., Spencer, J.: *The Probabilistic Method*, 2nd edn. John Wiley, New York (2000)
3. Henzinger, M.R., King, V., Warnow, T.: Constructing a tree from homeomorphic subtrees, with applications to computational evolutionary biology. In: *Proceedings of the 7th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 333–340 (1996)
4. Jiang, T., Kearney, P.E., Li, M.: A polynomial time approximation scheme for inferring evolutionary trees from quartet topologies and its application. *SIAM Journal on Computing* 30(6), 1942–1961 (2000)
5. Semple, C., Steel, M.A.: *Phylogenetics*. Oxford University Press, Oxford (2003)
6. Snir, S., Yuster, R.: Reconstructing approximate phylogenetic trees from quartet samples. In: *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Austin, USA, pp. 1035–1044 (January 2010)
7. Steel, M.: The complexity of reconstructing trees from qualitative characters and subtrees. *Journal of Classification* 9(1), 91–116 (1992)
8. Vazirani, V.V.: *Approximation Algorithms*. Springer, Heidelberg (2004)

Viral Processes by Random Walks on Random Regular Graphs

Mohammed Abdullah¹, Colin Cooper¹, and Moez Draief^{2,*}

¹ Department of Informatics, King's College London
{mohammed.abdullah,colin.cooper}@kcl.ac.uk

² Department of Electrical and Electronic Engineering, Imperial College London
m.draief@imperial.ac.uk

Abstract. We study the SIR epidemic model with infections carried by k particles making independent random walks on a random regular graph. We give an edge-weighted graph reduction of the dynamics of the process that allows us to apply standard results of Erdős–Renyi random graphs on the particle set. In particular, we show how the parameters of the model produce two phase transitions: In the subcritical regime, $O(\ln k)$ particles are infected. In the supercritical regime, for a constant C determined by the parameters of the model, Ck get infected with probability C , and $O(\ln k)$ get infected with probability $(1 - C)$. Finally, there is a regime in which all k particles are infected. Furthermore, the edge weights give information about when a particle becomes infected. We demonstrate how this can be exploited to determine the completion time of the process by applying a result of Janson on randomly edge weighted graphs.

Keywords: Random walks, random graphs, viral processes, SIR model.

1 Introduction

The spread of an infection throughout a population, often referred to loosely as an *epidemic*, has come to be modelled in various ways in the literature, spurred by the richness of the domains in which the abstract notion of a virus has gone beyond the traditional biological phenomenon. Electronic viruses over computer networks are not the only extension, others include *rumour spreading* [20] or *broadcasting* [3] and *viral marketing* [13, ch. 9]. Models may vary over domains, but the underlying principle is one of spread of some unit of information or change of state through interaction between individuals.

In much of the literature on the spread of epidemics as well as the dissemination of information, individuals reside at fixed locations connected by a graph and the evolution of the state of an individual depends on the state of its neighbours in the graph. In particular if the graph is complete, mean-field (homogeneous mixing) models have been exploited to study the outcome of the diffusion

* Moez Draief gratefully acknowledges the Leverhulme trust funding through the Research Fellowship RF/9/RFG/2010/02/08.

process [9]. More recently, there has been an increasing interest in understanding the impact of the network topology on the spread of epidemics in networks with fixed nodes, see [13] for a review of such results. There has however been little analytical work to date on models where the possible interactions between the nodes are dynamic, i.e., evolves in time.

We explore a particular instance of dynamicity in interaction by assuming that individuals are mobile particles and can only infect each other if they are in sufficiently close proximity. The model is motivated both by certain kinds of biological epidemics, whose transmission may be dominated by sites at which individuals gather in close proximity (e.g., workplaces or public transport for a disease like SARS, cattle markets for foot-and-mouth disease, etc.) and by malware. Furthermore, it is relevant to studying the dissemination of information in opportunistic networks [5] where the information is transmitted between users who happen to be in each others range. As in the case of static networks [20] one may be interested in the time it takes for the rumour to be known to all users.

In our model (elaborated upon below) the virus can be carried by k particles making independent random walks on an n -vertex r -regular random graph G which represents the environment. Each particle is in one of three states: susceptible (S), infected (I), recovered (R). An infected particle can infect a susceptible particle, which remains infected for a fixed *infectious period* ξ before recovering permanently. This is known as the *SIR epidemic model* and is extensively studied in the literature. When $\xi = \infty$ (the *SI model*) particles never go from I to R. Two questions can be asked: (1) When $\xi < \infty$, how many particles ever get infected? (2) How long does the process take to complete? We address both of these questions by reducing the dynamics of the process to what we term an *interaction graph*. This is an Erdős–Renyi (E–R) random graph $\mathcal{G}_{k,\hat{q}}$ on the set of particles, where the edge probability \hat{q} is a function of the parameters of the model. Infected particles are connected components in $\mathcal{G}_{k,\hat{q}}$, and so well known results from the literature on E–R random graphs can be directly applied using our reduction to answer question (1). In particular, we show how the parameters of the model produce two phase transitions: In the subcritical regime, $O(\ln k)$ particles are infected. In the supercritical regime, for a constant C determined by the parameters of the model, Ck get infected with probability C , and $O(\ln k)$ get infected with probability $(1 - C)$. Finally, there is a regime in which all k particles are infected. Statements are with high probability (**whp**), that is, with probability tending to 1 as $n \rightarrow \infty$. Furthermore, the graph reduction assigns weights on the edges that give information about when a particle becomes infected. This information can be used for addressing question (2). As an example, in the case of $\xi = \infty$, we apply a result of Janson [16] on randomly-weighted graphs to deduce that completion time converges in probability to $\frac{2\theta_r n}{k} \ln k$, where $\theta_r = \frac{r-1}{r-2}$. This matches the expectation determined in [8] for the same example.

Whilst the metaphor of an epidemic is a motivating and illustrative one, this work is part of the more general scope of *interacting particle systems* (see, e.g., [2, ch. 14]).

Note, due to space restrictions, we refer to [11] for a more detailed exposition and proofs of lemmas and theorems not presented in this paper.

2 Model

Let $r \geq 3$ be fixed. Let \mathcal{G}_r denote the set of r -regular graphs with vertex set $V = \{1, 2, \dots, n\}$ and the uniform measure. Let $G = (V_G, E_G)$ be chosen uniformly at random (**uar**) from \mathcal{G}_r . The results in this paper are always asymptotic in $n = |V_G|$. The notation $A_n \sim B_n$ means that $\lim_{n \rightarrow \infty} A_n/B_n = 1$.

At step t , let $\mathcal{S}(t), \mathcal{I}(t), \mathcal{R}(t)$ be the set of susceptible, infected and recovered particles respectively. These form a partition of the particle set \mathcal{P} . Thus, $|\mathcal{P}| = k$. Let $I_0 = |\mathcal{I}(0)|$.

When two particles x and y meet at some vertex v at time step t , an *interaction* takes place between them at that time step with probability $\rho \in (0, 1]$, which is a fixed constant parameter of the model. We term such an event an *xy interaction* and call ρ the *interaction probability*. If one of the particles is infected and the other is susceptible, the infection takes place upon interaction. We define the *completion time* of the process to be the time step at which the last infection takes place. Consider that the counter has just changed from time step $t - 1$ to t . The sequence of actions is as follows: (i) Every particle makes an independent move in its random walk. (ii) A particle $x \in \mathcal{I}(t - 1)$ that had reached the end of its infectious period by time t changes from state I to state R, so $x \notin \mathcal{I}(t)$ and $x \in \mathcal{R}(t)$. Otherwise, it remains in state I, i.e., $x \in \mathcal{I}(t)$. (iii) Each pair of particles x, y at the same vertex v interact with probability ρ ; the particle-pair interactions are independent of each other. If one of x or y is in $\mathcal{I}(t)$, and the other was in $\mathcal{S}(t - 1)$, the latter becomes infected, and it becomes a member of $\mathcal{I}(t)$. New infections are considered to have started at time t ; we say a particle is infected at step t , and step t counts as one unit in the infectious period. (iv) The counter changes from t to $t + 1$, and the sequence is repeated.

A note on notation and terminology: When we refer to a ‘period’ $[t_1, t_2]$, we mean the set of time steps $\{t_1, t_1 + 1, t_1 + 2 \dots, t_2\}$. When we refer to “time t ” we are referring to step t on the counter - this is a discrete time process. The first time step is $t = 1$, and $t = 0$ is not a time step, just a useful convention to express the state of the system before the first time step.

Observe the pair-wise independence of the interactions: If x, y, z meet at a vertex v at time step t , there is an interaction between each pair ($\{xy, yz, xz\}$) with probability ρ , independently of the other interactions. From the sequence of actions, it can be seen that it is also the case that an infection is not transitive in a single time step. For example, say they meet at the same vertex at time step t , x is infected but y and z are not. If x interacts with y but not z , then y does not pass on the infection to z at that time step, even if they interact at that time step. Let M_k be the total number of particles that ever get infected in the course of the process, and let T_k be the completion time.

3 Results

Theorem 1. *Suppose $\rho = 1$ and $k \rightarrow \infty$ as $n \rightarrow \infty$. Define*

$$\phi \equiv k \left(1 - \left(1 - \frac{1}{\theta_r n} \right)^\xi \right) \tag{1}$$

where $\theta_r = \frac{r-1}{r-2}$. Then

- (i) If $\phi < 1$ then **whp**, $M_k = O(\ln k) + I_0$
- (ii) If $\phi \rightarrow c$ where $c > 1$ is some constant, then if $I_0 = o(k)$,
 - (a) with probability α , $M_k = C(k + o(k)) + I_0$,
 - (b) with probability $1 - \alpha$, $M_k = O(\ln k) + I_0$,
 where C is the unique solution in $(0, 1)$ of the equation $1 - x = e^{-\phi x}$ and $\alpha = (1 + o(1))(1 - (1 - C)^{I_0})$.
- (iii) If $\phi > (1 + \varepsilon) \ln k$ where $\varepsilon > 1$ is some constant, then **whp** $M_k = k$ (i.e., all the particles get infected).

Theorem 1 is for finite ξ , but observe that taking the convention that $x^\infty = 0$ for $|x| < 1$ means that part (iii) is consistent with the SI model, for which all particles get infected almost surely. The theorem effectively gives conditions for transitions between different possible “regimes” of behaviour. The most interesting is the regime of part (ii), which is entered as ϕ transitions from $\phi < 1$ to $\phi > 1$. When $I_0 = 1$, roughly speaking, in this transition the number of infected particles goes from very few ($O(\ln k)$) **whp**, to a constant fraction Ck with probability C , or $O(\ln k)$ with probability $1 - C$. Hence, it’s “all or nothing”, with a constant probability C of being “all”.

Observe further if $\xi = O(n/k)$, then we have $\phi \approx \frac{k\xi}{\theta_r n}$. So $\phi = c > 1$ if $\xi \approx \frac{c\theta_r n}{k}$. This is unsurprising since it can be seen from the tools we use in our analysis that the hitting time of one particle to any of the k others is distributed approximately $Geom(\frac{k}{\theta_r n})$, and hence has expected value $\frac{\theta_r n}{k}$.

Concerning the completion times, we will demonstrate in Sect. 7.2 that for $\xi = \infty$, (that is, the SI model) and $\rho = 1$, how the edge weightings can be exploited by a fairly straightforward application of a theorem of [16] to get

$$\frac{T_k}{\ln k/k} \xrightarrow{p} 2\theta_r n, \tag{2}$$

where T_k is the completion time for k particles. This is not a new result since it is implied by the matching expectation result from [8], however, our weighting scheme generalises to $\rho < 1$ and/or $\xi < \infty$ and in principle can be exploited in the same manner. In fact, for $\xi = \infty$, $\rho < 1$ we claim that (2) holds as $\frac{T_k}{\ln k/k} \xrightarrow{p} 2\frac{\theta_r n}{\psi}$.

For $\rho < 1$, we claim that Theorem 1 holds when the term $\frac{1}{\theta_r n}$ in (1) is replaced with $\frac{\psi}{\theta_r n}$ where

$$\psi = \frac{\rho(r - 1)}{r - 2 + \rho}. \tag{3}$$

See [1].

4 Related Work

In this section, we briefly describe some of the relevant related work on diffusion processes like epidemic spreading and the dissemination of information in mobile environments.

There has been a growing body of work in the interacting particle systems community analysing epidemic models with mobile particles. In [10] the authors provide a review of the results, techniques and conjectures when the graph is an infinite lattice. In [21], the authors explore by means of mean-field approximations the evolution of the number of infected individuals when individuals perform random motions on the plane. Recent papers by Peres et al [19] and Pettarin et al [22] both analyse mobile networks modeled as multiple random walks; the former as Brownian motion on \mathbb{R}^d and the latter as walks on the grid. In each case, there is a parameter r within which distance a pair of walks can communicate, producing a communication graph (which is disconnected below a certain threshold r_c). [19] studies various aspects of the communication graph, such as how long it takes a particular point to become part of the giant component. [22] studies the broadcast time T_B of a piece of information and originating in one agent in relation to r . Setting $r = 0$ means information is passed only when particles are coincident. In this case, T_B is equivalent to our completion time and [22] give $T_B = \tilde{O}(n/\sqrt{k})$.

Of closer relevance to this work are [12] and [11]. Both of these papers study infections carried by random walks on graphs. In particular [11] analyses an SI model similar to ours; multiple random walks on a graph G that carry a virus, and the quantity of interest is the completion time. They give a general upper bound $\mathbf{E}[T_k] = O(m^* \ln k)$ for any graph G , where m^* is the expected meeting time of a pair of random walks maximised over all pairs of starting vertices. Special cases are analysed too, in particular, they give an upperbound of $\mathbf{E}[T_k] = O(\frac{nr}{k} \ln k \ln n)$ for random r -regular graphs. This is a factor $\ln n$ larger than the precise value of the process considered in this paper.

Finally, we mention [3], which studies flooding on dynamic random networks. A fixed set of n vertices is part of a dynamic random graph process where each edge is an independent two-state Markov chain, either existing or not existing. A single initial vertex initially holds a message and any vertex which receives this message broadcasts it on existing edges for the next k steps. Although flooding is a different process to multiple random walks, the authors develop a reduction to a weighted random graph with some similarity to the interaction graphs we present. It allows them to derive relations between the edge-Markov probabilities and state asymptotic bounds on the number of vertices that receive the message, as well as the broadcast (equivalently, completion) time.

5 Assumptions and Approach

If each particle is distance at least $\omega(k, n) \equiv \Omega(\ln \ln n + \ln k)$ from every other then we say the particles are in *general position* (g.p.). We assume the following:

- (i) The number of particles $k \leq n^\epsilon$ where ϵ is some sufficiently small constant.
- (ii) G is typical. (iii) Particles start in general position. (iv) $I_0 = |\mathcal{I}(0)| \geq 1$ and $|\mathcal{R}(0)| = 0$.

A *typical* graph (formally defined in the appendix) is one that has a number of specific properties. It is connected and non-bipartite, which implies that a random walk will converge to a stationary distribution π on the vertex set. Because the graph is regular, π will be the uniform distribution. A typical graph also has good expansion properties, meaning that a random walk will converge quickly to the stationary distribution; it will be *rapidly mixing*. In fact, for all the graph we consider, $O(k \ln n)$ steps is sufficient for our results. A typical graph also has most of its vertices treelike. A vertex v is *treelike* if there is no cycle in the subgraph $G[v, L_1]$ induced by the set of vertices within distance $L_1 = \lfloor \epsilon_1 \log_r n \rfloor$ of v , where $\epsilon_1 > 0$ is a sufficiently small constant. The near-uniformity and treelike structure throughout the graph allows us to make precise calculations for the behaviour of a random walk from a treelike vertex within the period of the mixing time. This helps us separate the analysis of the period within the mixing time from the period after it. The rapid mixing means that the dynamics of the process is dominated by the long-term behaviour very quickly, and that essentially, short-term behaviour is restricted to groups of local interactions that don't interfere with each other. This provides a degree of homogeneity to the process.

Assumptions (ii) and (iii) are not unreasonable: G is typical **whp** (Lemma [10](#) in appendix), and it is straightforward to verify that if the positions of each of the k particles are chosen **uar** from the vertices of G , then **whp** they will be in g.p. with respect to each other if ϵ is small enough. We will require additional assumptions beyond those stated in this section and previous ones, and they will be shown to hold **whp** too. Assumption (iv) implies $|\mathcal{S}(0)| = k - I_0$.

The analysis of multiple random walks can be reduced to a single random walk, in a way we informally describe here based on techniques developed in [8](#) and extended in [11](#). We refer to the latter for details and rigorous justification for this section.

We define a *product graph* $H = (V_H, E_H)$ with vertex set $V_H = V^k$ and edge set $E_H = E^k$. The vertices \mathbf{v} of H consist of k -tuples $\mathbf{v} = (v_1, v_2, \dots, v_k)$ of vertices $v_i \in V(G), i = 1, \dots, k$, with repeats allowed. Two vertices \mathbf{v}, \mathbf{w} are adjacent if $(v_1, w_1), \dots, (v_k, w_k)$ are edges of G . The graph H replaces k random walks $\mathcal{W}_{u_i}(t)$ on G with current positions v_i and starting positions u_i by a single walk $\mathcal{W}_{\mathbf{u}}^H(t)$.

If $S \subseteq V_H$, then $\Gamma = \Gamma(S)$ is obtained from H by contracting S to a single vertex $\gamma(S)$. All edges, including loops and parallel edges are retained, producing a multigraph. Thus $d_\Gamma(\gamma) = d_H(S)$, where d_F denotes vertex degree in graph F . Moreover Γ and H have the same total degree $(nr)^k$, and the degree of any vertex of Γ , except γ , is r^k .

We can calculate the times for a pair of particles (or one particle and a set of others, or any pair from a set of pairs, or whatever) to meet by calculating the time for the single random walk $\mathcal{W}_{\mathbf{u}}^H(t)$ to visit the relevant set of vertices

$S \subseteq V_H$. This is turn is done by the contraction described above, and calculating the time for the walk $\mathcal{W}_{\mathbf{u}}^T(t)$ on Γ to visit $\gamma(S)$. These two times asymptotically equal.

The dynamics of the system, is as follows. The random walks have a *mixing time* $T = O(k \ln n)$, and the distribution of a particle at some time $t \geq T$ is almost the stationary distribution. In this case, because the graph is regular, the stationary distribution is uniform across the vertices of G .

Consider a set A of particle pairs. For a particle pair $(x, y) \in A$, let $\mathcal{B}_{(x,y)}(t)$ be the event {no pair in A meet in the period $[T, t-1]$ and x and y meet at time t }. Note, (x, y) is unordered. Subject to some restrictions (which are satisfied by the type of interactions relevant to this paper), it is established in [1] that

$$\Pr(\mathcal{B}_{(x,y)}(t)) \sim p(1 - |A|p)^{t-1}, \tag{4}$$

where

$$p = \frac{1}{\theta_r n} \left(1 + O\left(\frac{k \ln n}{n}\right) \right), \tag{5}$$

During the mixing time, we cannot be precise in general. However, if the particles begin in general position with respect to each other then **whp**, the particles don't meet in the mixing time. This is a consequence of Lemma 5. We say a visit to vertex v , or a particle-pair meeting is *T-distinct* if it occurs at least T steps after a previous T -distinct visit/meeting, or the start of the walk. As suggested by the above, we can view the dynamics within the mixing time separately to the dynamics after the mixing time, up until the visit to a vertex or a particle-pair meeting.

6 Two-Particle System

In this section, we discuss how the system behaves when $k = 2$. Let s and x be the two particles, with s being the initial infective. Suppose that the assumptions stated above hold. We allow $\xi < \infty$ and/or $\rho < 1$. The former conditions means that x may never get infected, the latter condition means that it may take more than one meeting between s and x before an interaction takes place. Note, that if s and x were at the same vertex at time t , and happen to move to the same neighbouring vertex in the next step, then this counts as another meeting.

By (4) with $A = \{(s, x)\}$,

$$\Pr(\text{first meeting between } s \text{ and } x \text{ after } T \text{ is at step } t) \sim p(1 - p)^{t-1}, \tag{6}$$

The RHS of (6) is the probability of a *Geom*(p) random variable taking value t .

Now, suppose s and x have just stepped to the same vertex v . With probability ρ there will be an interaction. After this, they will move again, either to the same neighbour of v with probability $1/r$ or to different neighbours with probability $(r - 1)/r$. Let

$$\phi = \Pr(\text{No } sx \text{ interaction before they move apart}) \tag{7}$$

$$= \sum_{i \geq 1} (1 - \rho)^i \left(\frac{1}{r}\right)^{i-1} \left(1 - \frac{1}{r}\right) = \frac{(1 - \rho)(r - 1)}{r - 1 + \rho}. \tag{8}$$

The following lemma is from [8]:

Lemma 2. *Let G be a typical r -regular graph, and let v be a vertex of G , treelike to depth $L_1 = \lfloor \epsilon_1 \log_r n \rfloor$. Suppose that at time zero, two independent random walks $(\mathcal{W}_1, \mathcal{W}_2)$ start from v . Let (a, b) denote the position of the particles at any step. Let $S = \{(u, u) : u \in V\}$. Let f be the probability of a first return to S within $T = O(k \ln n)$ steps given that the walks leave v by different edges at time zero. Then*

$$f = \frac{1}{(r - 1)^2} + O\left(n^{-\Omega(1)}\right).$$

Using this lemma, let

$$\phi_T = \Pr(\text{No } sx \text{ interaction before being apart more than } T \text{ time steps}) \tag{9}$$

$$= \sum_{i \geq 1} \phi^i f^{i-1} (1 - f) = \frac{\phi(1 - f)}{1 - \phi f} \tag{10}$$

Now, assuming s and x start at the same vertex,

$$\Pr(sx \text{ interaction occurs within } T \text{ time steps}) \tag{11}$$

$$\sim \Pr(sx \text{ interaction occurs before } s \text{ and } x \text{ have been apart more than } T \text{ steps}) \tag{12}$$

$$= 1 - \phi_T = 1 - \frac{\phi(1 - f)}{1 - \phi f} \sim \frac{\rho(r - 1)}{r - 2 + \rho} = \psi. \tag{13}$$

Recall ψ was defined in [3], and observe that $\rho \leq \psi \leq 1$ with $\psi = 1$ iff $\rho = 1$.

Clearly, the number of T -distinct meetings in $[0, t]$ is at most t/T . Subject to slightly more stringent restrictions, it can be shown (see [1]) that we have

$$\Pr(\text{there are } i \text{ } T\text{-distinct meetings in } [0, t]) \sim \binom{t}{i} p^i (1 - p)^{t-i}, \tag{14}$$

i.e., it is approximately distributed $Bin(t, p)$. The probability that there are no interactions in any of the i intervals $[t_j, t_j + T]$ where t_j is the time of the j 'th T -distinct meeting is $(1 - \psi)^i$. Thus

$$\begin{aligned} \Pr(\text{there are no interactions in the period } [0, t]) &\sim \sum_{i=0}^{t/T} \binom{t}{i} (p(1 - \psi))^i (1 - p)^{t-i} \\ &= \sim (1 - \psi p)^t \end{aligned}$$

Hence,

$$\Pr(x \text{ gets infected within time } \xi) \sim 1 - (1 - \psi p)^\xi. \tag{15}$$

When $\rho = \psi = 1$, (15) looks similar to the bracketed terms in (11). This is, of course, not a coincidence since the bracketed term in (11) is essentially the probability that an infection is passed between a pair of particles if one of them had been infected, and therefore, ϕ is effectively the expected number of other particles that are infected by a particular particle.

7 Interaction Graph Framework: SI Model

We remind the reader that proofs of lemmas and theorems not present in what follows are given in [1].

We can study the SI model (the case $\xi = \infty$) by analysing the edge weights of the *interaction graph* $\mathcal{Y} = (V_{\mathcal{Y}}, E_{\mathcal{Y}})$. This is a complete graph on the particle set \mathcal{P} , thus $V_{\mathcal{Y}} = \mathcal{P}$ and $E_{\mathcal{Y}} = \{(x, y) : x, y \in \mathcal{P}, x \neq y\}$. For a particle x , let $t(x)$ be the time at which x got infected, or ∞ if it never gets infected. For an *interaction edge* $e = (x, y) \in E_{\mathcal{Y}}$, let $t(e) = \min\{t(x), t(y)\}$. Then the weight $w_{\mathcal{Y}}(e)$ of the edge is a random variable defined as

$$w_{\mathcal{Y}}(e) = \min\{t - t(e) : t > t(e) \text{ and there was an } xy \text{ interaction at step } t\}. \tag{16}$$

If the other particle was susceptible at the time of interaction, it becomes infected at the point of interaction. \mathcal{Y} , therefore, is a space of randomly-weighted complete graphs, and the distribution on \mathcal{Y} is the joint distribution of the edge weight random variables $w_{\mathcal{Y}}(e)$. A member Z of this space - an instance of \mathcal{Y} - is a specific set of values for each random variable $w_{\mathcal{Y}}(e)$. We may write $Z_{\mathcal{Y}} \in \mathcal{Y}$ to denote a particular instance drawn from the space \mathcal{Y} with probability $\Pr(Z_{\mathcal{Y}})$.

We claim that the joint distribution of edge weights, is well approximated by the joint distribution of random edge weights of another complete graph on \mathcal{P} , $\Lambda = (V_{\Lambda}, E_{\Lambda})$. The weight $w_{\Lambda}(e)$ of each edge $e \in E_{\Lambda}$ is an independent random variable with distribution $Geom(q)$, where $q = \frac{\psi}{\theta_r n}$.

Labelling the edges in Λ as e_i with $1 \leq i \leq \binom{k}{2}$, observe

$$\Pr \left(\bigwedge_{i=1}^{\binom{k}{2}} w_{\Lambda}(e_i) = z_i \right) = \prod_{i=1}^{\binom{k}{2}} q(1 - q)^{z_i - 1} \tag{17}$$

Observe that with probability one, neither graph has an edge of infinite weight.

7.1 Interaction Graph Approximation

In what follows we give a proof of the following theorem for the special case in which $\rho = 1$ (implying $\psi = 1$). We claim that it holds when $\rho < 1$ as well.

Theorem 3. *Let*

$$Z_F = \left\{ \bigwedge_{i=1}^{\binom{k}{2}} w_F(e_i) = z_i \right\}. \tag{18}$$

where $F \in \{\mathcal{Y}, \Lambda\}$. Then

$$\Pr(Z_{\mathcal{Y}}) = (1 + o(1))\Pr(Z_A). \tag{19}$$

Edge weightings in the interaction graph are pair-wise independent by virtue of the independence of the random walks. However there is a correlation when more than two edges are considered. For example, if particles x and y interacted at some time t and x and z interacted at some time close to t , then y and z are more likely to have interacted at some time close to t .

We say an edge e is *active* in the period $[t(e), t(e) + w_{\mathcal{Y}}(e) - 1]$. The event $Z_{\mathcal{Y}}$ can be restated as follows: For each edge $e_i = (x, y)$, the first xy interaction after time $t(e_i)$ is at time $t(e_i) + z_i$.

For a pair of particles x, y and some weighted graph F on the particle set, let $D_F(x, y)$ represent the weighted distance between x and y in F . Let $d_F(x) = \min\{D_F(s, x) : s \in \mathcal{I}(0)\}$. Furthermore, for an edge $e = (x, y)$, let $d_F(e) = \min\{d_F(x), d_F(y)\}$. \square

Lemma 4. *For a particle x , $t(x) = d_{\mathcal{Y}}(x)$, and so for an edge $e \in E_{\mathcal{Y}}$, $t(e) = d_{\mathcal{Y}}(e)$.*

Therefore,

$$Z_{\mathcal{Y}} = \{\text{For each edge } e_i = (x, y), \text{ there is no } xy \text{ interaction in the period } [d_{\mathcal{Y}}(e_i) + 1, d_{\mathcal{Y}}(e_i) + z_i - 1] \text{ and there is an } xy \text{ interaction at time } d_{\mathcal{Y}}(e_i) + z_i\}. \tag{20}$$

Let A_0 be the set of active edges at time $\tau_0 = 0$. For example, in the case $\mathcal{I}(0) = \{s\}$, $A_0 = \{(s, x) : x \in \mathcal{P}, x \neq s\}$ whence $|A_0| = k - 1$. Let $\tau_i, 1 \leq i \leq R$ for some R , denote the times at which the set of active edges changes. Let A_i be the set of edges active in the period $[\tau_i, \tau_{i+1} - 1]$. We call the τ_i 's *epochs*. A particular edge $e_i = (x, y)$ will become active at some epoch $\tau = d_{\mathcal{Y}}(e_i)$ and remain so until $\tau' - 1$ (inclusive), where $\tau' = d_{\mathcal{Y}}(e_i) + z_i$ is some epoch after τ . Its period of activation may, therefore, cross a number of epochs, in which case it will be a member of a number of edge sets A_j .

We analyse each period $[\tau_j + 1, \tau_{j+1}]$ individually through application of (4). In the product graph H , we identify the set of vertices $S(A_j) \subseteq V_H$ such that $\mathbf{v} \in S(A_j)$ iff for some $(x, y) \in A_j$ \mathbf{v} represents x and y being at the same vertex in G . Thus, if $\mathbf{v} = (v_1, v_2, \dots, v_k)$, then $v_r = v_s$ where r and s are the vector indices for x and y respectively. Since each pair is not to interact in the period $[\tau_j + 1, \tau_{j+1} - 1]$, and since we assume $\rho = 1$, the walk \mathcal{W} on H must avoid the set $S(A_j)$ until $\tau_{j+1} - 1$ (inclusive). Then, at τ_{j+1} it must visit $S_j \subseteq S(A_j)$. $\mathbf{v} \in S_j$ iff it represents any of the $(x, y) \in A_j$ that interact at time τ_{j+1} being at the same vertex in G . We deal with (non-)visits to $S(A_j)$ and S_j by applying (4) to their contractions $\gamma(S(A_j))$ and $\gamma(S_j)$.

We do not prove (19) for all possible $Z_{\mathcal{Y}}$, only those that are termed *good*. Let $T = O(k \ln n)$ be a mixing time and let $L = T^3$, and let $\ell = 2(T + L)$.

Call $Z_{\mathcal{Y}}$ *good* if it satisfies these criteria: (a) $k^3 \ln n \sum_{i=1}^{\binom{k}{2}} z_i = o(n^2)$ (b) None of the $\binom{k}{2}$ interactions that form the edges of \mathcal{Y} take place within ℓ steps of each other.

Lemma 5. *With high probability, $Z_{\mathcal{Y}} \in \mathcal{Y}$ is good.*

We shall assume the $Z_{\mathcal{Y}}$ that is drawn is good. Note part (b) implies $R = \binom{k}{2}$ and so we only need to consider cases of $Z_{\mathcal{Y}}$ where, for each j , S_j is a single particle pair. It also implies that during the period $[\tau_j + 1, \tau_j + \ell]$, we do not need to account for interactions. The following lemma applies (4) sequentially for each period $[\tau_j + 1, \tau_{j+1}]$. By the Markov property, we can multiply the probabilities for each period. Theorem 3 follows from this.

Lemma 6. *Let $p_j = |A_j|p(1 - O(k^3 \ln n/n))$ where p is from (5),*

$$\Pr(Z_{\mathcal{Y}}) = \prod_{j=0}^R (1 + O(L\pi_j) + O(\epsilon_j))p(1 - p_j)^{\tau_{j+1} - \tau_j - 1} \tag{21}$$

$$= (1 + o(1))\Pr(Z_A). \tag{22}$$

In (21) we have included correction factors which we left out in (4) for clarity.

7.2 Completion Time

In [8] an expectation of $\frac{2\theta_x n}{k} \ln k$ was determined for the completion time of a broadcasting model on k particles that is equivalent to the SI model with $I_0 = 1$ and $\rho = 1$. We apply a theorem from [16] to get a convergence in probability to the same value. Assign each edge (i, j) of a complete graph on n vertices a random weight Y_{ij} . The weights are assumed to be independent and identically distributed, non-negative and satisfying $\Pr(Y_{ij} \leq t) = t + o(t)$ as $t \rightarrow 0$. Let X_{ij} be the minimal total weight of a path between a given pair of vertices i, j . The theorem states $\frac{\max_{j \leq n} X_{ij}}{\ln n/n} \xrightarrow{P} 2$. By scaling, we can apply the result to an exponential analogue of the edge probabilities of Λ , and show that the distribution is well approximated, thereby giving (2). See [1] for details.

8 Interaction Graph Framework: SIR Model

To analyse the SIR model, we build an interaction graph but we modify the process by considering two phases. Phase 1 assigns edge weights as before: an edge $e = (x, y)$ is weighted by the time it takes for an xy interaction after one of them has become infected. It is possible, of course, that neither ever get infected when ξ is finite, in which case the edge (x, y) does not become active (and is not given a weight) in this phase. Phase 1 ends at τ_{end} when there are no more active edges. At this point, there remains a set of susceptible particles $\mathcal{S}(\tau_{end})$ that were never infected in phase 1. Phase 2 begins with an arbitrary $z \in \mathcal{S}(\tau_{end})$

being given a pseudo-infection with an infinite infectious period, and we proceed with an SI process on the particle set $\mathcal{S}(\tau_{end})$, giving them edge weights, as per the normal SI process. A pseudo-infection is the same process as an infection but we do not count pseudo-infected particles as infected; only particles infected in phase 1 are counted as infect. Phase 2 exists to maintain the same probability space as in the SI model.

Call the resulting (complete) graph Ψ . We remove an edge e in Ψ if $w_\Psi(e) > \xi$ and call the resulting graph Ψ' .

Let $\mathcal{C}(x)$ be the component of Ψ' containing a particle x . Let $D_{\Psi'}(x, y)$ be the weighted distance between particles x and y if $y \in \mathcal{C}(x)$, otherwise let $D_{\Psi'}(x, y) = \infty$. Let $d_{\Psi'}(y) = \min\{D_{\Psi'}(x, y) : x \in \mathcal{I}(0)\}$.

Theorem 7. *In Ψ' , (i) A particle y is infected if and only if $y \in \mathcal{C}(x)$ for some initial infective $x \in \mathcal{I}(0)$, (ii) If a particle y gets infected, the infection time is $t(y) = d_{\Psi'}(y)$.*

We show that Theorem 3 holds for Ψ . Let Λ and Z_F for some graph space F be as before.

Lemma 8.

$$\Pr(Z_\Psi) = (1 + o(1))\Pr(Z_\Lambda). \tag{23}$$

As with Ψ , Ψ' is a space of weighted graphs. Furthermore, we can define an equivalence relation \sim on Ψ' such that for $Y, Z \in \Psi'$, we have $Y \sim Z$ iff $E(Y) = E(Z)$, that is, they have the same edge set. Let $\Psi'_{/\sim}$ be the graph space induced by Ψ' and the equivalence relation \sim . Thus, the probability a graph G is drawn from $\Psi'_{/\sim}$, that is, $\Pr(G \in \Psi'_{/\sim}) = \Pr(\text{some } Z \text{ is drawn from } \Psi' \text{ such that } E(Z) = E(G))$. We show $\Psi'_{/\sim}$ is approximated by $\mathcal{G}_{k, \hat{q}}$, the space of Erdős-Renyi random graphs with edge probability $\hat{q} = 1 - (1 - q)^\xi$, where $q = \frac{\psi}{\theta_{r,n}}$ for the special case $\rho = \psi = 1$.

Lemma 9. *Let G be a graph on the particle set \mathcal{P} .*

$$\Pr(G \in \Psi'_{/\sim}) = (1 + o(1))\Pr(G \in \mathcal{G}_{k, \hat{q}}). \tag{24}$$

Proof of Theorem 1. For an Erdős-Renyi random graph space $\mathcal{G}_{n,p}$ on n vertices and edge probability p , the connectivity results are well known (see, e.g., [13]). By Lemma 9, the whp statements carry through to $\Psi'_{/\sim}$. Since Theorem 1 deals with the case of one initial infective s , the infected particles will be those in $\mathcal{C}(s)$. Cases (i) and (iii) of Theorem 1 are straightforward to see from the above. For case (ii), there is a unique giant component g of size C_1 in $G \in \Psi'_{/\sim}$. By the symmetry of the model, the probability any particular particle x being in g is $|g|/k = (1 + o(1))C$. Thus, the giant component is infected with this probability. Otherwise, s will be placed in a component of size at most $O(\ln k)$. \square

References

1. Abdullah, M., Cooper, C., Draief, M.: Viral Processes by Random Walks on Random Regular Graphs (2011), <http://arxiv.org/abs/1104.3789>

2. Aldous, D., Fill, J.: Reversible Markov Chains and Random Walks on Graphs, <http://stat-www.berkeley.edu/pub/users/aldous/RWG/book.html> (in preparation)
3. Baumann, H., Crescenzi, P., Fraigniaud, P.: Parsimonious Flooding in Dynamic Graphs. In: Proceedings of the 28th ACM Symposium on Principles of Distributed Computing, pp. 260–269 (2009)
4. Buscarino, A., Fortuna, L., Frasca, M., Latora, V.: Disease Spreading in Populations of Moving Agents. *EPL (Europhysics Letters)* 82(3) (2008)
5. Chaintreau, A., Hui, P., Scott, J., Gass, R., Crowcroft, J., Diot, C.: Impact of Human Mobility on Opportunistic Forwarding Algorithms. *IEEE Transactions Mobile Computing* 6(6), 606–620 (2007)
6. Cooper, C., Frieze, A.: The Cover Time of Random Regular Graphs. *SIAM Journal on Discrete Mathematics* 18, 728–740 (2005)
7. Cooper, C., Frieze, A.M.: The Cover Time of the Giant Component of a Random graph. *Random Structures and Algorithms* 32, 401–439 (2008)
8. Cooper, C., Frieze, A.M., Radzik, T.: Multiple Random Walks in Random Regular Graphs. *SIAM Journal on Discrete Mathematics* 23, 1738–1761 (2009)
9. Daley, D., Gani, J.: Epidemic Modelling: An Introduction. *Studies in Mathematical Biology*. Cambridge University Press, Cambridge (2001)
10. Dickman, R., Rolla, L., Sidoravicius, V.: Activated Random Walkers: Facts, Conjectures and Challenges. *Journal of Statistical Physics* 138, 126–142 (2010)
11. Dimitriou, T., Nikolettseas, S., Spirakis, P.: The Infection Time of Graphs. *Discrete Applied Mathematics* 154, 18 (2006)
12. Draief, M., Ganesh, A.: A random Walk Model for Infection on Graphs: Spread of Epidemics and Rumours with Mobile Agents. *Discrete Event Dynamic Systems* 21, 1 (2011)
13. Draief, M., Massoulié, L.: Epidemics and Rumours in Complex Networks. *London Mathematical Society Series*, vol. 369. Cambridge University Press, Cambridge (2010)
14. Feller, W.: An Introduction to Probability Theory, 2nd edn., vol. I. Wiley, New York (1960)
15. Friedman, J.: A Proof of Alon’s Second Eigenvalue Conjecture and Related Problems, vol. 195. *Memoirs of the American Mathematical Society*, Providence (2008)
16. Janson, S.: One, Two and Three Times $\log n/n$ for Paths in a Complete Graph with Random Weights. *Combinatorics, Probability and Computing* 8, 347–361 (1999)
17. Kurtz, T.G., Lebensztayn, E., Leichsenring, A.R., Machado, F.P.: Limit Theorems for an Epidemic Model on the Complete Graph. *Latin American Journal of Probability and Mathematical Statistics* 4, 45–55 (2008)
18. Lovász, L.: Random Walks on Graphs: A Survey, *Combinatorics*. In: Paul Erdős is Eighty, vol. 2, pp. 1–46. *Bolyai Society Mathematical Studies* (1993)
19. Peres, Y., Sinclair, A., Sousi, P., Stauffer, A.: Mobile Geometric Graphs: Detection, Coverage and Percolation. In: Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 412–428 (2011)
20. Pittel, B.: On spreading a rumor. *SIAM Journal of Applied Mathematics* 47(1), 213–223 (1987)
21. Rhodes, C., Nekovee, M.: The Opportunistic Transmission of Wireless Worms Between Mobile Devices. *Physica A: Statistical Mechanics and Its Applications* 387(27), 6837–6840 (2008)
22. Pettarin, A., Pietracaprina, A., Pucci, G., Upfal, E.: Tight Bounds on Information Dissemination in Sparse Mobile Networks. In: Proceedings of the 30th Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (2011)

23. Zhou, J., Liua, Z.: Epidemic Spreading in Communities with Mobile Agents. *Physica A: Statistical Mechanics and its Applications* 388(7), 1, 1228–1236 (2009)

Appendix

Typical r -regular Graphs

We say an r -regular graph G is *typical* if it satisfies the properties P1–P4 listed below. We first give some definitions.

Let

$$L_1 = \lfloor \epsilon_1 \log_r n \rfloor, \tag{25}$$

where $\epsilon_1 > 0$ is a sufficiently small constant.

A vertex v is *treelike* if there is no cycle in the subgraph $G[v, L_1]$ induced by the set of vertices within distance L_1 of v .

A cycle C is *small* if $|C| \leq L_1$.

P1. G is connected and not bipartite.

P2. The second eigenvalue of the adjacency matrix of G is at most $2\sqrt{r-1} + \epsilon$, where $\epsilon > 0$ is an arbitrarily small constant.

P3. There are at most $n^{2\epsilon_1}$ vertices on small cycles.

P4. No pair of cycles C_1, C_2 with $|C_1|, |C_2| \leq 100L_1$ are within distance $100L_1$ of each other.

Note that P3 implies that at most n^{ϵ_C} vertices of a typical r -regular graph are not treelike, where

$$n^{\epsilon_C} = O(r^{L_1} n^{2\epsilon_1}) = O(n^{3\epsilon_1}). \tag{26}$$

Lemma 10. *Let $\mathcal{G}'_r \subseteq \mathcal{G}_r$ be the set of typical r -regular graphs. Then $|\mathcal{G}'_r| \sim |\mathcal{G}_r|$.*

For Lemma 10, that P2 holds **whp** is a very difficult result of Friedman 15. The other properties are straightforward to establish; see, e.g., 6.

Quantum Property Testing for Bounded-Degree Graphs

Andris Ambainis¹, Andrew M. Childs², and Yi-Kai Liu³

¹ Faculty of Computing, University of Latvia
ambainis@lu.lv

² Department of Combinatorics & Optimization and
Institute for Quantum Computing, University of Waterloo
amchilds@uwaterloo.ca

³ Department of Computer Science, University of California, Berkeley
yikailiu@eecs.berkeley.edu

Abstract. We study quantum algorithms for testing bipartiteness and expansion of bounded-degree graphs. We give quantum algorithms that solve these problems in time $\tilde{O}(N^{1/3})$, beating the $\Omega(\sqrt{N})$ classical lower bound. For testing expansion, we also prove an $\tilde{\Omega}(N^{1/4})$ quantum query lower bound, thus ruling out the possibility of an exponential quantum speedup. Our quantum algorithms follow from a combination of classical property testing techniques due to Goldreich and Ron, derandomization, and the quantum algorithm for element distinctness. The quantum lower bound is obtained by the polynomial method, using novel algebraic techniques and combinatorial analysis to accommodate the graph structure.

1 Introduction

In *property testing*, one is asked to distinguish between objects that satisfy a property P and objects that are far from satisfying P . The goal is to design algorithms that test properties in sublinear or even constant time, without reading the entire input—a task that is nontrivial even for properties that can be computed in polynomial time. This is motivated by the practical question of how to extract meaningful information from massive data sets that are too large to fit in a single computer’s memory and can only be handled in small pieces.

Testing properties of graphs is an interesting special case. Many graph properties, such as connectivity and planarity, can be tested in constant time, independent of the number of vertices N [19, 22]. However, some graph properties are much harder to test. For bounded-degree graphs in the adjacency-list representation, the best classical algorithms for testing *bipartiteness* [20] and *expansion* [21, 16, 25, 29] use $\tilde{O}(\sqrt{N})$ queries. In fact, this is nearly optimal, as there are $\Omega(\sqrt{N})$ query lower bounds for both problems [22]. As a natural extension, we

¹ Here, the graph can be specified by an adjacency matrix (suitable for dense graphs) or by a collection of adjacency lists (for bounded-degree graphs).

² We use tilde notation to suppress logarithmic factors.

consider whether these problems can be solved more efficiently using *quantum* queries.

There has been some previous work on quantum property testing. In particular, there are examples of exponential separations between quantum and classical property testing [12], and there are quantum algorithms for testing juntas [9], solvability of black-box groups [24], uniformity and orthogonality of distributions [13, 14], and certain properties related to the Fourier transform [2, 14]. However, aside from concurrent work on testing graph isomorphism [14], we are not aware of previous work on quantum algorithms for testing properties of graphs³

Here, we give quantum algorithms for testing bipartiteness and expansion of bounded-degree graphs in time only $\tilde{O}(N^{1/3})$, beating the $\Omega(\sqrt{N})$ classical lower bounds [22]. Moreover, we prove that any quantum algorithm for testing expansion must use $\tilde{\Omega}(N^{1/4})$ queries, showing that quantum computers cannot achieve a superpolynomial speedup for this problem.

Why might quantum computers offer an advantage for testing bipartiteness and expansion? The classical algorithms for these problems use random walks to explore the graph, so one might hope to do better by using quantum walks, which are a powerful tool for searching graphs [32]. In fact, our algorithms use quantum walks indirectly. The classical algorithm for testing bipartiteness is based on checking whether a pair of short random walks form an odd-length cycle in the graph, thereby certifying non-bipartiteness [20]. The algorithm for testing expansion looks for collisions between the endpoints of short random walks, with a large number of collisions indicating that the walk is not rapidly mixing [21]. In both cases, the property is tested by looking for collisions among a set of $\tilde{O}(\sqrt{N})$ items. By using the quantum walk algorithm for element distinctness [7, 27] to look for these collisions, we can solve the problem using $\tilde{O}(N^{1/3})$ quantum queries. In addition, we show that the above classical algorithms can be derandomized, using $O(\log N)$ -wise independent bits. This yields quantum algorithms that run in time $\tilde{O}(N^{1/3})$.

While we have shown a polynomial quantum speedup, one may ask whether an exponential speedup is possible. Quantum computers can give at most a polynomial speedup for total functions [10], but this limitation does not apply to property testing (and indeed, examples of exponential speedup are known [12]). On the other hand, superpolynomial speedup is impossible for symmetric functions [3], even in the case of partial functions such as those arising in property testing. It is an interesting question whether exponential speedups are possible for testing *graph* properties, which may have significantly less symmetry.

Here we prove that testing expansion requires $\tilde{\Omega}(N^{1/4})$ quantum queries, thus ruling out the possibility of an exponential speedup. We use the polynomial method [10]—specifically, a technique of Aaronson based on reduction to a bivariate polynomial [1]. We define a distribution over N -vertex graphs with ℓ

³ Quantum speedups are known for *deciding* certain graph properties, without the promise that the graph either has the property or is far from having it [17, 26, 15]. This turns out to be a fairly different setting, and the results there are not directly comparable to ours.

connected components (and with another parameter $M \approx N$), such that each component is an expander with high probability. With $\ell = 1$ component, such graphs are almost surely expanders, whereas graphs with $\ell \geq 2$ components are very far from expanders. Our main technical contribution is to show that the acceptance probability of any T -query quantum algorithm, when presented with this distribution, is well-approximated by a bivariate polynomial in M and ℓ of degree $O(T \log T)$. This requires a somewhat involved calculation of a closed-form expression for the acceptance probability as a function of M and ℓ , using algebraic techniques and the combinatorics of partitions. Then it follows by known results on polynomial approximation that $\Omega(N^{1/4}/\log N)$ queries are necessary to test expansion.

This proof may be of independent interest since there are very few techniques available to prove quantum lower bounds for property testing. In particular, the standard quantum adversary method [6] is subject to a “property testing barrier” [23]. Furthermore, graph structure makes it difficult to apply the polynomial method, so our lower bound for testing expansion requires substantial new machinery. These techniques may be applicable to other problems with graph structure. Note also that our approach is an alternative to the classical lower bounds for testing bipartiteness and expansion [22].

We are only aware of a few previous lower bounds for quantum property testing: the result that not all languages can be tested efficiently [12] (which is nonconstructive, using a counting argument), and lower bounds for testing orthogonality and uniformity of distributions [13, 14] and for testing graph isomorphism [14] (which follow by reduction from the collision problem).

Despite this progress, there remain many unanswered questions about quantum testing of graph properties. So far, we have been unable to prove a superconstant lower bound for testing bipartiteness. More generally, is there any graph property testing problem that admits an exponential quantum speedup?

In the remainder of this section, we define the model of quantum property testing. We use the adjacency-list model for graphs with bounded (i.e., constant) degree d . A graph $G = (V, E)$ is represented by a function $f_G: V \times \{1, \dots, d\} \rightarrow V \cup \{*\}$, where $f_G(v, i)$ returns the i^{th} neighbor of v in G , or $*$ if v has fewer than i neighbors. A quantum computer is provided with a unitary black box that reversibly computes f_G as $|v, i, z\rangle \mapsto |v, i, z \oplus f_G(v, i)\rangle$. The query complexity of an algorithm is the number of calls it makes to the black box for f_G .

We say that G is ε -far from satisfying a property P if one must change at least εnd edges of G in order to satisfy P . We say that an algorithm ε -tests P if it accepts graphs that satisfy P with probability at least $2/3$, and rejects graphs that are ε -far from satisfying P with probability at least $2/3$. (More generally, we may consider algorithms that determine whether a graph satisfies P or is ε -far from satisfying a related property P' .)

We say that a graph G is an α -expander if for every $U \subseteq V$ with $|U| \leq |V|/2$, we have $|\partial(U)| \geq \alpha|U|$, where $\partial(U)$ is the set of vertices in $V - U$ adjacent to at least one vertex of U .

2 Quantum Algorithms for Bipartiteness and Expansion

First, recall the classical algorithm for testing bipartiteness [20]. This algorithm performs $T = \Theta(1/\varepsilon)$ repetitions, where during each repetition it chooses a random starting vertex s , then does $K = \sqrt{N} \text{poly}(\frac{\log N}{\varepsilon})$ random walks from s , each of length $L = \text{poly}(\frac{\log N}{\varepsilon})$, and looks for “collisions” where two walks from s reach the same vertex v , one after an even number steps, the other after an odd number of steps.

We derandomize each of the T repetitions separately. Each repetition uses $n = O(KL \log d)$ bits of randomness. We claim that it suffices to use k -wise independent random bits for some $k = O(L \log d)$. To see this, consider the analysis given in [20]. Lemma 4.5 of [20] states sufficient conditions for the algorithm to find an odd cycle, and hence reject, with high probability. The proof considers the random variable $X = \sum_{i < j} \eta_{ij}$, where η_{ij} is a Boolean random variable that indicates whether walk i collides with walk j while having different parity. The probability that $X = 0$ is upper bounded using Chebyshev’s inequality together with bounds on $E[X]$ and $\text{Var}[X]$. Note that $E[X]$ and $\text{Var}[X]$ are linear and quadratic in the η_{ij} , respectively, so they only depend on sets of at most $O(L \log d)$ random bits. Thus they are unchanged by substituting k -wise independent random bits for some $k = O(L \log d)$. This reduces the number of random bits required by the algorithm to $O(k \log n) = O(\text{poly}(\frac{\log N \log d}{\varepsilon}))$.

We then combine this derandomized classical algorithm with Ambainis’ quantum algorithm for element distinctness [7, 27, 35]. (For details, see the full version of this paper [8].) This shows

Theorem 1. *There is a quantum algorithm that always returns “true” when G is bipartite, returns “false” with constant probability when G is ε -far from bipartite, and runs in time $O(N^{1/3} \text{poly}(\frac{\log N}{\varepsilon}))$.*

Using similar ideas, we can also give an $\tilde{O}(N^{1/3})$ -time quantum algorithm for testing expansion. We start with the classical algorithm of [21], derandomize it using k -wise independent random variables, and apply the quantum algorithm for element distinctness. There is a slight complication, because we need to count collisions, not just detect them. However, the number of collisions is small—roughly $O(N^{2\mu})$ where μ is chosen to be a small constant—so we can count the collisions using brute force. See [8] for details.

3 Quantum Lower Bound for Testing Expansion

3.1 Overview

We now turn to lower bounds for testing expansion. Specifically, we prove

Theorem 2. *Any quantum algorithm for testing expansion of bounded-degree graphs must use $\Omega(N^{1/4} / \log N)$ queries.*

Proof. We consider random graphs G on N vertices, sampled from the following distribution $P_{M,l}$ (where $M \geq N$ and l divides M):

1. We start by constructing a random graph G' on M vertices, as follows: First, we partition the vertices into l sets V_1, \dots, V_l , with each set V_i containing M/l vertices. Then, on each set V_i , we create a random subgraph by randomly choosing c perfect matchings on V_i and taking their union. (Here c is some sufficiently large constant.)
2. We then construct G as follows: First, we pick a subset of vertices v_1, \dots, v_N from G' . To pick v_1 , we choose one of the sets V_1, \dots, V_l uniformly at random, call it V_j , and we let v_1 be a random vertex from V_j . For each subsequent vertex v_i , we again select a set V_j uniformly at random, and choose v_i uniformly at random among those vertices of V_j that were not chosen in the previous steps. Then we let G be the induced subgraph of G' on v_1, \dots, v_N .

The process above fails if we try to choose more than M/l vertices from the same V_j . However, the probability of that happening is small—on average, N/l vertices are chosen in each V_j . We choose $M = (1 + \Theta(N^{-0.1}))N$. Then a straightforward application of Chernoff bounds implies that the process fails with probability at most $e^{-\Omega(N^{0.55})}$. For more detail, see Section C.1 in [8].

Note that the resulting graph G has degree at most c . The reason for choosing G as a subgraph of G' (rather than constructing G directly) is that this leads to simpler formulas for the probabilities of certain events, e.g., the probability that vertices v_1, v_2 and v_3 all belong to the same component of G is $1/l^2$. This seems essential for our use of the polynomial method.

If $l = 1$, then this process generates an expander with high probability. It is well known [31, 28] that the graph on M vertices generated by taking c perfect matchings is an expander with high probability. In Section C.2 in [8], we show that the subgraph that we choose is also an expander. (Informally, the main reason is that only a $\Theta(N^{-1/4})$ fraction of the vertices of G' are not included in G . This allows us to carry out the proof of [31, 28] without substantial changes.)

If $l = 2$, then this process generates a disconnected graph with two connected components, each of size roughly $N/2$. Such a graph is very far from any expander graph—specifically, for any α' , it is at least about $(\alpha'/2d)$ -far from an α' -expander of maximum degree d .

Therefore, if a quantum algorithm tests expansion, it must accept a random graph generated according to $P_{M,1}$ with probability at least $2/3$, and a random graph generated according to $P_{M,2}$ with probability at most $1/3$. (Graphs drawn from $P_{M,l}$ with $l > 2$ must also be accepted with probability at most $1/3$, although this fact is not used in the analysis.)

The strategy of the proof is as follows. We show that for any quantum algorithm run on a random graph from the distribution $P_{M,l}$, the acceptance probability of the algorithm can be approximated by a bivariate polynomial in M and l , where the number of queries used by the algorithm corresponds to the degree of this polynomial. (This is our main technical contribution.) We then lower bound the degree of this polynomial.

In more detail, we will prove the following lemma (see [Section 3.2](#)):

Lemma 1. *Let A be a quantum algorithm using T queries. The acceptance probability of A (for the probability distribution $P_{M,l}$) is approximated (up to an additive error of $e^{-\Omega(N^{0.55})}$) by a fraction $\frac{f(M,l)}{g(M,l)}$, where $f(M,l)$ and $g(M,l)$ are polynomials of degree $O(T \log T)$ and $g(M,l)$ is a product of factors $(M - (2k - 1)l)$ for $k \in \{1, \dots, T\}$, with $(M - (2k - 1)l)$ occurring at most $2T/k$ times.*

Now choose $a = 1 + \Theta(N^{-0.1})$ such that aN is even. We say that a pair (M, l) is δ -good if $M \in [aN - \delta^{3/2}, aN + \delta^{3/2}]$, $l \leq \delta$, and l divides M .

We then approximate the fraction $\frac{f(M,l)}{g(M,l)}$ (from [Lemma 1](#)) by $\frac{f(M,l)}{(aN)^{\deg g(M,l)}}$. For each term $M - (2k - 1)l$, we first replace it by M and then by aN . The first step introduces multiplicative error of $1 - \frac{(2k-1)l}{M} \geq 1 - \frac{2kl}{N} \approx e^{-2kl/N}$. For all terms together, the error introduced in this step is at most $\prod_{k=1}^T (e^{-2kl/N})^{2T/k} = e^{-4T^2 l/N}$. If $T = O(N^{1/4}/\log N)$ and $l = O(N^{1/2})$, the multiplicative error is $1 - o(1)$.

The second approximation step introduces multiplicative error of

$$\left(\frac{M}{aN}\right)^{O(T \log T)} \approx (e^{(M-aN)/aN})^{O(T \log T)} \leq (e^{\delta^{3/2}/aN})^{O(T \log T)}.$$

If $\delta = O(N^{1/2})$ and $T = O(N^{1/4}/\log N)$, this can be upper bounded by $1 + \epsilon$ for arbitrarily small $\epsilon > 0$, by appropriately choosing the big- O constant in $T = O(N^{1/4}/\log N)$.

Next, we prove a second lemma, which lower bounds the degree of a bivariate polynomial:

Lemma 2. *Let $f(M, l)$ be a polynomial such that $|f(aN, 1) - f(aN, 2)| \geq \epsilon$ for some fixed $\epsilon > 0$ and, for any δ -good (M, l) , $|f(M, l)| \leq 1$. Then the degree of $f(M, l)$ is $\Omega(\sqrt{\delta})$.*

The proof of this lemma follows the collision lower bounds of Aaronson and Shi [\[1, 33\]](#) and is included in Section C.3 in [\[8\]](#) for completeness.

We now set $\delta = \Theta(N^{1/2})$ and apply [Lemma 2](#) to $\frac{f(M,l)}{2(aN)^{\deg g(M,l)}}$. This is a polynomial in M and l , because the denominator is a constant. With $M = aN$, its values at $l = 1$ and $l = 2$ are bounded away from each other by at least $1/3$ since the algorithm works. Its values at δ -good pairs (M, l) have magnitude at most 1 because the acceptance probability of the algorithm is in $[0, 1]$, so $|\frac{f(M,l)}{2(aN)^{\deg g(M,l)}}| \leq \frac{1}{2} + o(1)$. Thus we find that the degree of $f(M, l)$ must be $\Omega(N^{1/4})$. It follows that $T = \Omega(N^{1/4}/\log N)$ queries are necessary.

3.2 Proof of [Lemma 1](#)

Here we assume that the process generating a graph G from the probability distribution $P_{M,l}$ does not fail. (The effect of this process possibly failing is considered in Section C.1 in [\[8\]](#).) The acceptance probability of A is a polynomial P_A of degree at most $2T$ in Boolean variables $x_{u,v,j}$, where $x_{u,v,j} = 1$ iff (u, v) is an edge in the j^{th} matching.

P_A is a weighted sum of monomials. It suffices to show that the expectation of every such monomial has the rational form described in [Lemma 1](#). If this is shown, then $E[P_A]$ is a sum of such fractions: $E[P_A] = \frac{f_1(M,l)}{g_1(M,l)} + \frac{f_2(M,l)}{g_2(M,l)} + \dots$. We put these fractions over a common denominator, obtaining $E[P_A] = \frac{f(M,l)}{g(M,l)}$ where $g(M,l) = \text{lcm}(g_1(M,l), g_2(M,l), \dots)$. In this common denominator, $(M - (2k - 1)l)$ occurs at most $2T/k$ times. Therefore, the degree of $g(M,l)$ is at most $2T \sum_{k=1}^{2T} \frac{1}{k} = O(T \log T)$. Similarly, the degree of $f(M,l)$ is at most $O(T \log T) + \deg g(M,l) = O(T \log T)$.

Now consider a particular monomial $P = x_{u_1, v_1, j_1} x_{u_2, v_2, j_2} \dots x_{u_d, v_d, j_d}$, where $d = \deg P$. Let G_P be the graph with edges $(u_1, v_1), \dots, (u_d, v_d)$ (i.e., with the edges relevant to P) where the edge (u_a, v_a) comes from the j_a^{th} matching. Let C_1, \dots, C_k be the connected components of G_P . For each component C_i , let X_i be the event that every edge (u_a, v_a) in C_i (viewed as a subgraph of G_P) is present in the random graph G as part of the j_a^{th} matching. We have to find an expression for the expectation

$$E[P] = \Pr[X_1 \cap X_2 \cap \dots \cap X_k].$$

We first consider $\Pr[X_i]$. Let v_i be the number of vertices in C_i , and for each matching j , let $d_{i,j}$ be the number of variables $x_{u,v,j}$ in P that have $u, v \in C_i$ and label j . Note that

$$d_{i,1} + d_{i,2} + \dots + d_{i,c} \geq v_i - 1 \tag{1}$$

because a connected graph with v_i vertices must have at least $v_i - 1$ edges. We have

$$\Pr[X_i] = \frac{1}{l^{v_i-1}} \prod_{j=1}^c \prod_{j'=1}^{d_{i,j}} \frac{1}{M/l - (2j' - 1)} = \frac{1}{l^{v_i-1}} \prod_{j=1}^c \prod_{j'=1}^{d_{i,j}} \frac{l}{M - (2j' - 1)l}. \tag{2}$$

Here $l^{-(v_i-1)}$ is the probability that all v_i vertices are put into the same set V_j (for some $1 \leq j \leq l$) (which is a necessary condition for having edges among them), and $\prod_{j'=1}^{d_{i,j}} \frac{1}{M/l - (2j' - 1)}$ is the probability that $d_{i,j}$ particular edges from the j^{th} matching are present. (For the first edge (u, v) in the j^{th} matching, the probability that it is present is $\frac{1}{M/l-1}$, since u is equally likely to be matched with any of M/l vertices in V_j except for u itself; for the second edge (u', v') in the j^{th} matching, the probability that it is present is $\frac{1}{M/l-3}$, since u' can be matched with any of M/l vertices except u, v, u' ; and so on. Note that without loss of generality, we can assume that the edges in P from the j^{th} matching are distinct. If P contains the same edge twice from the same matching, then we can remove one of the duplicates without changing the value of P .)

We can rewrite [\(2\)](#) as $\Pr[X_i] = \frac{1}{l^{v_i-1}} \prod_{j=1}^c R_{d_{i,j}}$, where we define

$$R_d = \prod_{j'=1}^d \frac{l}{M - (2j' - 1)l}. \tag{3}$$

We now extend this to deal with multiple components C_i at once, i.e., we want to evaluate $\Pr[\bigcap_{i \in S} X_i]$, where $S \subseteq \{1, \dots, k\}$. Let E_S be the event that the vertices in $\bigcup_{i \in S} C_i$ (i.e., in any of the components indicated by S) are all put into one set V_j . Then $\Pr[\bigcap_{i \in S} X_i | E_S] = \prod_{j=1}^c R_{\sum_{i \in S} d_{i,j}}$. The event E_S happens with probability $l^{-(\sum_{i \in S} v_i)+1}$, since the total number of vertices in $\bigcup_{i \in S} C_i$ is $\sum_{i \in S} v_i$.

Let $L = (S_1, \dots, S_t)$ be a partition of $\{1, 2, \dots, k\}$. We call S_1, \dots, S_t *classes* of the partition L . We say that $S \in L$ if S is one of S_1, \dots, S_t . Let $|L| = t$. We say that L is a refinement of L' (denoted $L < L'$) if L can be obtained from L' by splitting some of the classes of L' into two or more parts. We write $L \leq L'$ if $L < L'$ or $L = L'$. When $L < L'$, let $c_{L,L'}$ be the number of sequences $L = L_0 < L_1 < \dots < L_j = L'$, with sequences of even length j counting as $+1$ and sequences of odd length j counting as -1 . We define $c_{L,L'} = 1$ when $L = L'$. We have the following partition identity, which will be useful later; the proof is given in Section C.4 in [8].

Proposition 1. *Suppose $L'' < L$. Then $\sum_{L': L'' \leq L' \leq L} c_{L',L} = 0$.*

We define the expressions

$$f_L(M, l) = \prod_{S \in L} \prod_{j=1}^c R_{\sum_{i \in S} d_{i,j}} \tag{4}$$

$$f'_L(M, l) = \sum_{L': L' \leq L} c_{L',L} f_{L'}(M, l). \tag{5}$$

We can now evaluate $\Pr[X_1 \cap X_2 \cap \dots \cap X_k]$ as follows. For any partition L of $\{1, 2, \dots, k\}$, let E_L be the event $\bigcap_{S \in L} E_S$. Let E'_L be the event that E_L happens but no $E_{L'}$ with $L < L'$ happens (i.e., L is the least refined partition that describes the event). Then

$$\Pr[X_1 \cap X_2 \cap \dots \cap X_k] = \sum_L \Pr[E'_L] f_L(M, l).$$

By inclusion-exclusion, $\Pr[E'_L] = \sum_{L': L \leq L'} c_{L,L'} \Pr[E_{L'}]$. Now substitute into the previous equation, reorder the sums, and use the definition of $f'_L(M, l)$:

$$\Pr[X_1 \cap X_2 \cap \dots \cap X_k] = \sum_{L'} \Pr[E_{L'}] \sum_{L: L \leq L'} c_{L,L'} f_L(M, l) = \sum_L \Pr[E_L] f'_L(M, l).$$

Note that $\Pr[E_L] = \prod_{S \in L} \Pr[E_S] = \prod_{S \in L} l^{-(\sum_{i \in S} v_i)+1} = l^{-(\sum_{i=1}^k v_i)+|L|}$. Thus we have

$$\Pr[X_1 \cap X_2 \cap \dots \cap X_k] = \sum_L l^{-(\sum_{i=1}^k v_i)+|L|} f'_L(M, l). \tag{6}$$

We have now written $\Pr[X_1 \cap X_2 \cap \dots \cap X_k]$ as a sum of rational functions of M and l . We can combine these into a single fraction $\frac{f(M,l)}{g(M,l)}$. It remains to show that this fraction has the properties claimed in Lemma 1.

First, we claim that the denominator $g(M, l)$ contains at most $2T/k$ factors of $M - (2k - 1)l$. Observe that each $f_L(M, l)$ is a fraction whose denominator consists of factors $M - (2k - 1)l$. The number of factors in the denominator is equal to the number of variables in the monomial P , which is at most $2T$. By the form of (3), for each $M - (2k - 1)l$ in the denominator, we also have $M - l, M - 3l, \dots, M - (2k - 3)l$ in the denominator. Therefore, if we have t factors of $M - (2k - 1)l$ in the denominator, then the total degree of the denominator is at least tk . Since $tk \leq 2T$, we have $t \leq 2T/k$. This statement holds for $f_L(M, l)$ for every L . Thus, when we sum the $f_L(M, l)$ to obtain first $f'_L(M, l)$ and then $\Pr[X_1 \cap X_2 \cap \dots \cap X_k]$, and put all the terms over a common denominator $g(M, l)$, this statement also holds for $g(M, l)$.

In $\Pr[X_1 \cap X_2 \cap \dots \cap X_k]$, when we sum the $f'_L(M, l)$ in (6), we also have factors of $l^{\sum_{i=1}^k v_i - |L|}$ in the denominator. Proposition 2 shows that these factors are cancelled out by corresponding factors in the numerator.

Proposition 2. $f'_L(M, l)$ is equal to a fraction whose denominator is a product of factors $(M - (2k - 1)l)$ and whose numerator is divisible by $l^{\sum_{i=1}^k v_i - |L|}$.

When we combine the different $f'_L(M, l)$ in (6) into a single fraction $\frac{f(M, l)}{g(M, l)}$, we see that f and g have the desired form. Also note that f and g have degree $O(T \log T)$, by repeating the same argument used earlier to combine the different monomials P . This completes the proof of Lemma 1; it remains to show Proposition 2.

Proof (of Proposition 2). Note that R_d contains an obvious factor of l^d . We define

$$R'_d = \frac{R_d}{l^d} = \prod_{j'=1}^d \frac{1}{M - (2j' - 1)l}$$

and we redefine $f_L(M, l)$ and $f'_L(M, l)$ (equations (4) and (5)) using R'_d instead of R_d . This removes a factor of l^d from the numerator of R_d and a factor of $l^{\sum_{i,j} d_{i,j}}$ from the numerator of $f_L(M, l)$. By equation (11), this factor is at least $l^{\sum_i v_i - k}$. Therefore, it remains to show that the numerator of the redefined $f'_L(M, l)$ is divisible by $l^{k - |L|}$.

Recall that $f'_L(M, l)$ is a sum of terms $f_{L'}(M, l)$ for all $L' \leq L$. Let us write each term as $f_{L'}(M, l) = 1 / \prod_{k \in K(L')} (M - kl)$, where $K(L')$ is a multiset. We put these terms over a common denominator $\beta_L(M, l) = \prod_{k \in B(L)} (M - kl)$, where $B(L) \supseteq K(L')$ for all $L' \leq L$. Then we have

$$f_{L'}(M, l) = \frac{\alpha_{L'}(M, l)}{\beta_L(M, l)}, \quad \alpha_{L'}(M, l) = \prod_{k \in B(L) - K(L')} (M - kl),$$

$$f'_L(M, l) = \frac{\alpha'_L(M, l)}{\beta_L(M, l)}, \quad \alpha'_L(M, l) = \sum_{L': L' \leq L} c_{L', L} \alpha_{L'}(M, l).$$

Let $m = |B(L)|$. Also, let $\tilde{m} = |K(L')| = \sum_{S \in L'} \sum_{j=1}^c \sum_{i \in S} d_{i,j} = \sum_{i=1}^k \sum_{j=1}^c d_{i,j}$, which is independent of L' . Let $m' = |B(L) - K(L')| = m - \tilde{m}$, which depends on L but not on L' .

We want to show that $\alpha'_L(M, l)$ is divisible by $l^{k-|L|}$. First, we multiply out each term $\alpha_{L'}(M, l)$ to get $\alpha_{L'}(M, l) = \sum_{i=0}^{m'} e_i(B(L) - K(L')) M^{m'-i} (-l)^i$, where e_i is the i^{th} elementary symmetric polynomial (i.e., $e_i(B(L) - K(L'))$ is the sum of all products of i variables chosen without replacement from the multiset $B(L) - K(L')$). We can then write $\alpha'_L(M, l)$ as

$$\alpha'_L(M, l) = \sum_{i=0}^{m'} \theta_{L,i} M^{m'-i} (-l)^i, \quad \theta_{L,i} = \sum_{L': L' \leq L} c_{L',L} e_i(B(L) - K(L')).$$

It suffices to show that, for all $0 \leq i \leq k - |L| - 1$, the coefficient $\theta_{L,i}$ is 0. Note that if L is the finest possible partition L_* , then $|L| = k$ and the above claim is vacuous, so we can assume that $L_* < L$. Also note that $\theta_{L,0} = 0$ by **Proposition 1** with $L'' = L_*$, so it suffices to consider $i > 0$.

For any set of variables E and any $a \geq 0$, define the power-sum polynomial $T_a(E) = \sum_{k \in E} k^a$. We can write $e_i(B(L) - K(L'))$ in terms of power sums:

$$e_i(B(L) - K(L')) = A_{i,L}[T_a(B(L) - K(L')): a = 0, 1, 2, \dots, i],$$

where $A_{i,L}$ is a polynomial function of the power sums $T_a(B(L) - K(L'))$ of total degree i in the variables $k \in B(L) - K(L')$. Note that the polynomial $A_{i,L}$ only depends on the size of the set $B(L) - K(L')$, hence it only depends on L , and not on L' . To simplify things, we can write $T_a(B(L) - K(L')) = T_a(B(L)) - T_a(K(L'))$ and absorb the $T_a(B(L))$ term into the polynomial $A_{i,L}$ to get a new polynomial $\tilde{A}_{i,L}$. Then we have $e_i(B(L) - K(L')) = \tilde{A}_{i,L}[T_a(K(L')): a = 0, 1, 2, \dots, i]$, and

$$\theta_{L,i} = \sum_{L': L' \leq L} c_{L',L} \tilde{A}_{i,L}[T_a(K(L')): a = 0, 1, 2, \dots, i].$$

It suffices to show that, for all $0 \leq i \leq k - |L| - 1$, the above sum vanishes term-by-term, i.e., for all sequences $\{a_j\}$ such that $a_j \geq 0$ and $\sum_j a_j \leq i$, we have

$$\sum_{L': L' \leq L} c_{L',L} \prod_j T_{a_j}(K(L')) = 0. \tag{7}$$

We have $T_a(K(L')) = \sum_{S \in L'} \sum_{j=1}^c T_a(\{1, 3, 5, \dots, 2(\sum_{i \in S} d_{i,j}) - 1\})$, by the definition of $K(L')$. Note that, for any integer s , $T_a(\{1, 3, 5, \dots, 2s - 1\}) = T_a(\{1, 2, 3, \dots, 2s\}) - 2^a T_a(\{1, 2, 3, \dots, s\})$, and by Faulhaber's formula, this equals a polynomial $Q_a(s)$ of degree $a + 1$, with rational coefficients and no constant term. We have $T_a(K(L')) = \sum_{S \in L'} \sum_{j=1}^c Q_a(\sum_{i \in S} d_{i,j})$. Let $q_{a,\alpha}$ ($\alpha = 1, \dots, a + 1$) be the coefficients of Q_a . Then we can rewrite this as

$$T_a(K(L')) = \sum_{\alpha=1}^{a+1} q_{a,\alpha} S_\alpha(L'), \quad \text{where } S_\alpha(L') = \sum_{S \in L'} \sum_{j=1}^c \left(\sum_{i \in S} d_{i,j} \right)^\alpha.$$

It suffices to show that the sum in equation (7) vanishes term-by-term, i.e., for all $0 \leq i \leq k - |L| - 1$ and for all sequences $\{\alpha_j\}$ such that $\alpha_j \geq 1$ and $\sum_j (\alpha_j - 1) \leq i$, we have

$$\sum_{L': L' \leq L} c_{L',L} \prod_j S_{\alpha_j}(L') = 0.$$

This final claim is shown in Section C.5 in [8]. This completes the proof of [Proposition 2](#).

Acknowledgments. We thank the anonymous referees for several helpful comments. AA was supported by ESF project 1DP/1.1.1.2.0/09/APIA/VIAA/044, FP7 Marie Curie Grant PIRG02-GA-2007-224886 and FP7 FET-Open project QCS. AMC and YKL acknowledge the hospitality of the Kavli Institute for Theoretical Physics, where this research was supported in part by the National Science Foundation under Grant No. PHY05-51164. AMC received support from MITACS, NSERC, QuantumWorks, and the US ARO/DTO. YKL received support from an NSF postdoctoral fellowship and ARO/NSA. This work was done in part while YKL was at the Institute for Quantum Information at Caltech.

References

- [1] Aaronson, S.: Quantum lower bound for the collision problem. In: STOC, pp. 635–642 (2002)
- [2] Aaronson, S.: BQP and the polynomial hierarchy. In: STOC, pp. 141–150 (2010)
- [3] Aaronson, S., Ambainis, A.: The need for structure in quantum speedups. In: Innovations in Computer Science, pp. 338–352 (2011)
- [4] Alon, N., Babai, L., Itai, A.: A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. of Algorithms* 7(4), 567–583 (1986)
- [5] Alon, N., Goldreich, O., Hastad, J., Peralta, R.: Simple constructions of almost k-wise independent random variables. *Random Structures and Algorithms* 3(3), 289–304 (1992)
- [6] Ambainis, A.: Quantum lower bounds by quantum arguments. *J. of Computer and System Sciences* 64(4), 750–767 (2002)
- [7] Ambainis, A.: Quantum walk algorithm for element distinctness. *SIAM J. on Computing* 37(1), 210–239 (2007)
- [8] Ambainis, A., Childs, A.M., Liu, Y.-K.: Quantum property testing for bounded-degree graphs, arXiv:1012.3174 (2010)
- [9] Atici, A., Servedio, R.: Quantum algorithms for learning and testing juntas. *Quantum Information Processing* 6(5), 323–348 (2007)
- [10] Beals, R., Buhrman, H., Cleve, R., Mosca, M., de Wolf, R.: Quantum lower bounds by polynomials. *J. of the ACM* 48(4), 778–797 (2001)
- [11] Buhrman, H., Durr, C., Heiligman, M., Hoyer, P., Magniez, F., Santha, M., de Wolf, R.: Quantum algorithms for element distinctness. *SIAM J. on Computing* 34(6), 1324–1330 (2005)
- [12] Buhrman, H., Fortnow, L., Newman, I., Rohrig, H.: Quantum property testing. *SIAM J. on Computing* 37(5), 1387–1400 (2008)

- [13] Bravyi, S., Harrow, A.W., Hassidim, A.: Quantum algorithms for testing properties of distributions. In: STACS, pp. 131–142 (2010)
- [14] Chakraborty, S., Fischer, E., Matsliah, A., de Wolf, R.: New results on quantum property testing. In: FSTTCS, pp. 145–156 (2010)
- [15] Childs, A.M., Kothari, R.: Quantum query complexity of minor-closed graph properties. To appear in STACS (2011)
- [16] Czumaj, A., Sohler, C.: Testing expansion in bounded-degree graphs. In: FOCS, pp. 570–578 (2007)
- [17] Durr, C., Heiligman, M., Hoyer, P., Mhalla, M.: Quantum query complexity of some graph problems. *SIAM J. on Computing* 35(6), 1310–1328 (2006)
- [18] Goldreich, O.: *Randomized Methods in Computation*, Lecture 2 (2001), <http://www.wisdom.weizmann.ac.il/~oded/rnd.html>
- [19] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *J. of the ACM* 45(4), 653–750 (1998)
- [20] Goldreich, O., Ron, D.: A sublinear bipartiteness tester for bounded degree graphs. *Combinatorica* 19(3), 335–373 (1999)
- [21] Goldreich, O., Ron, D.: On testing expansion in bounded-degree graphs. ECCC report TR00-020 (2000)
- [22] Goldreich, O., Ron, D.: Property testing in bounded degree graphs. *Algorithmica* 32(2), 302–343 (2002)
- [23] Hoyer, P., Lee, T., Spalek, R.: Negative weights make adversaries stronger. In: STOC, pp. 526–535 (2007)
- [24] Inui, Y., Le Gall, F.: Quantum property testing of group solvability. In: LATIN 2008. LNCS, vol. 4957, pp. 772–783. Springer, Heidelberg (2008)
- [25] Kale, S., Seshadhri, C.: Testing expansion in bounded-degree graphs, ECCC report TR07-076 (2007)
- [26] Magniez, F., Santha, M., Szegedy, M.: Quantum algorithms for the triangle problem. *SIAM J. on Computing* 37(2), 413–424 (2007)
- [27] Magniez, F., Nayak, A., Roland, J., Santha, M.: Search via quantum walk. In: STOC, pp. 575–584 (2007)
- [28] Motwani, R., Raghavan, P.: *Randomized Algorithms*. Cambridge University Press, Cambridge (1995)
- [29] Nachmias, A., Shapira, A.: Testing the expansion of a graph. *Information and Computation* 208, 309–314 (2010)
- [30] Paturi, R.: On the degree of polynomials that approximate symmetric Boolean functions (preliminary version). In: STOC, pp. 468–474 (1992)
- [31] Pinsker, M.: On the complexity of a concentrator. In: Proceedings of the 7th International Teletraffic Conference, pp. 318/1–318/4 (1973)
- [32] Santha, M.: Quantum walk based search algorithms. In: *Theory and Applications of Models of Computation*, pp. 31–46 (2008)
- [33] Shi, Y.: Quantum lower bounds for the collision and the element distinctness problems. In: FOCS, pp. 513–519 (2002)
- [34] Simon, D.R.: On the power of quantum computation. *SIAM J. on Computing* 26(5), 1474–1483 (1997)
- [35] Szegedy, M.: Quantum speed-up of Markov chain based algorithms. In: FOCS, pp. 32–41 (2004)

Lower Bounds on the Query Complexity of Non-uniform and Adaptive Reductions Showing Hardness Amplification

Sergei Artemenko and Ronen Shaltiel*

University of Haifa

Abstract. Hardness amplification results show that for every function f there exists a function $\text{Amp}(f)$ such that the following holds: if every circuit of size s computes f correctly on at most a $1 - \delta$ fraction of inputs, then every circuit of size s' computes $\text{Amp}(f)$ correctly on at most a $1/2 + \epsilon$ fraction of inputs. All hardness amplification results in the literature suffer from “size loss” meaning that $s' \leq \epsilon \cdot s$. In this paper we show that proofs using “non-uniform reductions” must suffer from size loss. To the best of our knowledge, all proofs in the literature are by non-uniform reductions. Our result is the first lower bound that applies to non-uniform reductions that are *adaptive*.

A reduction is an oracle circuit $R^{(\cdot)}$ such that when given oracle access to any function D that computes $\text{Amp}(f)$ correctly on a $1/2 + \epsilon$ fraction of inputs, R^D computes f correctly on a $1 - \delta$ fraction of inputs. A *non-uniform* reduction is allowed to also receive a short advice string α that may depend on both f and D in an arbitrary way. The well known connection between hardness amplification and list-decodable error-correcting codes implies that reductions showing hardness amplification cannot be uniform for $\epsilon < 1/4$. A reduction is *non-adaptive* if it makes non-adaptive queries to its oracle. Shaltiel and Viola (STOC 2008) showed lower bounds on the number of queries made by non-uniform reductions that are *non-adaptive*. We show that every non-uniform reduction must make at least $\Omega(1/\epsilon)$ queries to its oracle (even if the reduction is *adaptive*). This implies that proofs by non-uniform reductions must suffer from size loss.

We also prove the same lower bounds on the number of queries of non-uniform and adaptive reductions that are allowed to rely on arbitrary specific properties of the function f . Previous limitations on reductions were proven for “function-generic” hardness amplification, in which the non-uniform reduction needs to work for every function f and therefore cannot rely on specific properties of the function.

1 Introduction

1.1 Background on Hardness Amplification

Hardness amplification results transform functions that are hard on the worst case (or sometimes mildly hard on average) into functions that are very hard on

* This research was supported by BSF grant 2004329 and ISF grant 686/07.

average. These results play an important role in computational complexity and cryptography. There are many results of this kind in the literature depending on the precise interpretation of “hard”. In this paper we focus on hardness against Boolean circuits and use the following notation.

Definition 1. Let $g : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$.

- Let $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$. We say that C has agreement p with g if $\Pr_{X \leftarrow U_n}[C(X) = g(X)] \geq p$.
- Let $C : \{0, 1\}^n \rightarrow \{0, 1\}^\ell \cup \{\perp\}$. We say that C has errorless agreement p with g if C has agreement p with g and for every $x \in \{0, 1\}^n$, if $C(x) \neq \perp$ then $C(x) = g(x)$.
- We say that g is p -hard for size s if no circuit C of size s has agreement p with g . We say that g is p -hard for errorless size s if no circuit C of size s has errorless agreement p with g .

Typical hardness amplification results start from a function $f : \{0, 1\}^k \rightarrow \{0, 1\}$ that is p -hard for size s and show that some function $g : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ is p' -hard for size s' . (The reader should think of k, n, p, p', s, s' and ℓ as parameters). These results “amplify hardness” in the sense that p' is typically much smaller than p (meaning that g is harder on average than f). We now briefly survey some of the literature on hardness amplification.

Worst-case to average-case. Here $p = 1$ (meaning that f is hard on the worst case for circuits of size s), $\ell = 1$ (meaning that g is Boolean), and $p' = 1/2 + \epsilon$ for a small parameter ϵ (meaning that circuits of size s' have advantage at most ϵ over random guessing when attempting to compute g). Many such results are known [Lip91, BFNW93, IW97, IW98, STV01, TV07, GGH07] see [Tre04] for a survey article.

Mildly-average-case to average case. This setup is similar to the one above except that $p = 1 - \delta$ for some small parameter δ (meaning that f is mildly average-case hard for circuits of size s). In other words, the setup of worst-case to average-case above can be seen as a special case in which $\delta < 1/2^k$. An extensively studied special case is Yao’s XOR-Lemma in which $g(x_1, \dots, x_t) = f(x_1) \oplus \dots \oplus f(x_t)$ [Lev87, Imp95, IW97, KS03, Tre03] see [GNW95] for a survey article. Other examples are [O’D04, HVV06, Tre05, GK08].

Non-Boolean target function. The two setups mentioned above can also be considered when the target function $g : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ is not Boolean. In the Boolean case we set $p' = 1/2 + \epsilon$ as it is trivial to have agreement of $1/2$. We typically consider $\ell > \log(1/\epsilon)$ and set $p' = \epsilon$. Namely, it is required that no circuit of size s' has agreement ϵ with g . An extensively studied special case is direct-product theorems in which $g(x_1, \dots, x_t) = (f(x_1), \dots, f(x_t))$ [Imp95, IW97, GNW95, GG11, IJK09a, IJK09b, IJKW10].

Errorless amplification. The three notions above are also studied when the circuits attempting to compute f and g are errorless [BS07, Wat10].

We are interested in proving lower bounds on hardness amplification results. We want our lower bounds to hold for all the settings mentioned above. For this purpose we will focus on a specific setting (which we refer to as “basic hardness amplification”) that is implied by all the settings mentioned above.

Basic hardness amplification. Let $\epsilon, \delta > 0$ and $\ell \geq 1$ be parameters. The *basic* hardness amplification task is to show that if f is $(1 - \delta)$ -hard for size s then g is ϵ -hard for *errorless* size s' . Stated in the contra-positive, the basic hardness amplification task is to show that if there exists a circuit D of size s' that has errorless agreement $p' = \epsilon$ with g then there exists a circuit C of size s that has agreement $p = 1 - \delta$ with f .

It is easy to see that basic hardness amplification is indeed implied by all the settings considered above.¹ Therefore, lower bounds on basic hardness amplification immediately apply to all the aforementioned settings. We make this statement more precise in Section 1.2.

Generic Hardness Amplification and Error-correcting Codes. Most of the hardness amplification results in the literature are *function-generic*, meaning that they provide a map Amp mapping functions $f : \{0, 1\}^k \rightarrow \{0, 1\}$ into functions $g = Amp(f)$ where $g : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ and show that for every f that is p -hard for size s , the function $g = Amp(f)$ is p' -hard for size s' . In contrast, a *function-specific* hardness amplification result uses specific functions f, g and the proof of the hardness amplification result is allowed to use specific properties of these functions. Examples of function-specific hardness amplification are [Lip91, IW98, TV07, Tre03, Tre05].

It is known that function-generic hardness amplification from worst-case to strongly average-case is closely related to (locally) list-decodable codes [STV01]. We elaborate on this relationship in the full version.

Size Loss in Hardness Amplification. A common disadvantage of all hardness amplification results surveyed above is that when starting from a function that is hard for circuits of size s , one obtains a function that is hard for circuits of smaller size $s' \leq \epsilon \cdot s$. This is a major disadvantage as it means that if one starts from a function that is hard for size s , existing results cannot produce a function that is $(1/2 + \epsilon)$ -hard for $\epsilon < 1/s$. It is natural to ask whether such a loss is necessary. In order to make this question precise, we need to consider formal models for proofs of hardness amplification results.

¹ Note that the basic hardness amplification task is trivially implied by all the settings above in case that g is non-boolean. In case g is Boolean, if there exists a circuit D of size s' that has errorless agreement ϵ with g then we can easily convert this circuit into a circuit D of size $s' + O(1)$ that has agreement $1/2 + \epsilon/2$ with g . Given input x , circuit D applies circuit D on x and outputs the same value if it is not ‘ \perp ’, and a fixed bit $b \in \{0, 1\}$ otherwise. It is easy to see that there exists a choice of b for which D has agreement $1/2 + \epsilon/2$ with g .

1.2 Non-uniform Reductions for Hardness Amplification

We are interested in proving impossibility results on proofs for hardness amplification and therefore consider the weakest variant of hardness amplification (which is *basic* hardness amplification). The notion that we study in this paper is that of “non-uniform” reductions. As explained in Section 1.3, this notion (defined below) captures the proofs of almost all hardness amplification results in the literature.

Definition 2 (non-uniform reduction). *Let $f : \{0, 1\}^k \rightarrow \{0, 1\}$ and $g : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ be functions. Let ϵ, δ and a be parameters. A non-uniform reduction showing basic hardness amplification (for f, g, ϵ, δ and a) is an oracle circuit $R^{(\cdot)}$ which takes two inputs $x \in \{0, 1\}^k$ and $\alpha \in \{0, 1\}^a$. It is required that for every function $D : \{0, 1\}^n \rightarrow \{0, 1\}^\ell \cup \{\perp\}$ that has errorless agreement ϵ with g , there exists a string $\alpha \in \{0, 1\}^a$ (which we call an “advice string”) such that the function $C(x) = R^D(x, \alpha)$ has agreement $1 - \delta$ with f .*

We say that R is semi-uniform if $a = 0$ (in which case R does not receive an advice string α). The size of the reduction is the size of the oracle circuit $R^{(\cdot)}$. We say that R makes at most q queries if for every choice of oracle D and inputs $x \in \{0, 1\}^k, \alpha \in \{0, 1\}^a$, reduction $R^D(x, \alpha)$ makes at most q queries to its oracle. We say that R is non-adaptive if for every choice of oracle and inputs, R makes non-adaptive queries to its oracle.

In the discussion below we explain the choices made in Definition 2.

Usefulness of Non-uniform Reductions. We first note that a non-uniform reduction indeed implies a basic hardness amplification result in the following sense: If there exists a circuit D of size s' that has errorless agreement ϵ with g then we have that $C(x) = R^D(x, \alpha)$ has agreement $1 - \delta$ with f , and furthermore, C can be implemented by a circuit of size $s = r + a + q \cdot s'$ where r is the size of R and q is the number of queries made by R . It follows that the number of queries q made by the reduction is the dominant factor in the ratio between s and s' . In other words, if we show that every reduction R must use at least $q = \Omega(1/\epsilon)$ queries, then we get that $s = \Omega(s'/\epsilon)$ which gives that the size loss is $s' = O(s \cdot \epsilon)$.

What is Non-uniform in This Reduction? Reduction R has two sources of non-uniformity: First, R is a circuit and therefore may be hardwired with non-uniform

² We make a comment about terminology. The literature on impossibility results for reductions often uses the term “black-box” when referring to reductions. We do not use this term as the definition above allows the reduction R to get an advice string α that may be an arbitrary function of the “oracle function” D given to it. There is no requirement that α can be computed by using few black-box queries to D . In fact, the issue that R receives non-black-box information about its oracle is the main difficulty that we need to solve in this paper. In contrast, semi-uniform reductions are black-box (as they only have black-box access to D). They are not uniform as they are circuits (meaning that they may be hardwired with advice that depends on f and g).

advice (that may depend on f). Note that this is the case even for semi-uniform reductions. The second (and more interesting) source of non-uniformity is the advice string α . It is important to stress that the order of quantifiers in the definition above allows α to depend on the choice of D (in addition to the choice of f). This is in contrast to the non-uniformity of R that is fixed in advance and does not depend on D .

Lower Bounds for Semi-uniform Reductions. We now illustrate the difference between semi-uniform reductions and general non-uniform reductions. It is not hard to show that semi-uniform reductions have to use $q = \Omega(1/\epsilon)$ queries. This follows by a folklore argument (attributed to Steven Rudich in [GNW95]). Consider a probability distribution over oracles which is uniformly distributed over all functions D that have errorless agreement ϵ with g . A semi-uniform reduction that makes $q = o(1/\epsilon)$ queries has probability $1 - o(1)$ to see only ‘ \perp ’ on its q queries. Therefore, such a reduction cannot expect to get meaningful information from its oracle, and can be used to construct a small circuit (with no oracle) that has agreement $1 - \delta - o(1)$ with f . This shows that the existence of a reduction R unconditionally implies that f is not $(1 - \delta - o(1))$ -hard.

We stress that the argument above critically depends on the fact that R is semi-uniform. A non-uniform reduction is allowed to receive an advice string α that is a function of D . Such an advice string can encode queries $y \in \{0, 1\}^n$ such that $D(y) \neq \perp$. While this does not seem to help R in having large agreement with f , the argument of Rudich no longer applies. As we point out next, semi-uniform reductions are rare exceptions in the literature on hardness amplification, and the main contribution of this paper is developing techniques to extend Rudich’s argument for *non-uniform* and *adaptive* reductions.

Non-uniform Reductions for other Settings of Hardness Amplification. Definition 2 is tailored for basic hardness amplification. However, the same reasoning can be used to define all the hardness amplification setups surveyed in Section 1.1. More precisely, we define the notion of “non-uniform reduction showing mildly-average-case to average-case hardness amplification” similarly by replacing the requirement that “ D has errorless agreement ϵ with g ” with the requirement that “ D has agreement p with g ” where $p = 1/2 + \epsilon$ in case $\ell = 1$ and $p = \epsilon$ in case $\ell > 1$. The discussion above about usefulness of non-uniform reductions trivially applies to this setting as well. Moreover, it trivially follows that a non-uniform reduction showing mildly-average-case to average-case hardness amplification implies a non-uniform reduction showing basic hardness amplification with essentially same parameters. As a consequence proving a lower bound of $q = \Omega(1/\epsilon)$ on the number of queries used by reductions showing basic hardness amplification entails the same lower bound in all the settings described in Section 1.1.

Function-generic Hardness Amplification. Definition 2 considers *specific* functions f, g . Most of the hardness amplification results in the literature are *function generic* in the following sense:

Definition 3 (function-generic hardness amplification). *Let ϵ, δ, a and ℓ be parameters. A function-generic reduction showing basic hardness amplification (for parameters ϵ, δ, a and ℓ) is a pair (Amp, R) where Amp is a map from functions $f : \{0, 1\}^k \rightarrow \{0, 1\}$ to functions $\text{Amp}(f) : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$, and for every function $f : \{0, 1\}^k \rightarrow \{0, 1\}$, $R^{(\cdot)}$ is a non-uniform reduction showing basic hardness amplification for $f, g = \text{Amp}(f), \epsilon, \delta$ and a .*

We use Definition 3 to also define the analogous notion for mildly-average-case to average-case hardness amplification. For the special case of Boolean mildly-average-case to average-case hardness amplification Definition 3 is equivalent to the notion of “black-box hardness amplification” defined in [SV10]. It is known that function-generic hardness amplification is equivalent to certain variants of list-decodable error-correcting codes. We elaborate on this connection in the full version.

1.3 Our Results

Function-generic Hardness Amplification. The vast majority of hardness amplification in the literature are function-generic reductions showing worst-case to average-case hardness amplification (or mildly-average-case to average-case hardness amplification). To the best of our knowledge, all the proofs in the literature are captured by Definition 3. Moreover, by the aforementioned connection to error-correcting codes, the reductions in these settings cannot be semi-uniform in the “list-decoding regime” (that is for $\epsilon < 1/4$). Consequently, Rudich’s argument does not apply for showing lower bounds on these reductions. Theorem 1 below proves lower bounds on the number of queries made by function-generic reductions showing basic hardness amplification.

Theorem 1 (main theorem for function-generic reductions).

There exists a constant $c > 1$ such that the following holds. Let $k, n, \ell, \epsilon, \delta, r$ and a be parameters such that $a, \frac{1}{\epsilon}, \frac{1}{\delta}, n, r \leq 2^{k/c}$ and $\delta \leq 1/3$. Let (Amp, R) be a function-generic reduction showing basic hardness amplification (for $f, g, \epsilon, \delta, \ell$ and a) and assume that R is of size r . Then, R makes at least $\frac{1}{100\epsilon}$ queries.

We have stated Theorem 1 in a general form with many parameters. In typical hardness amplification results the parameter setting is $n = \text{poly}(k)$, $\ell = 1$, $\epsilon = 1/k^b$ for some constant b (or sometimes slightly super constant b), $\delta \leq 1/3$, and $r, a = \text{poly}(k)$. Note that Theorem 1 holds for this choices. (In fact, the theorem holds even when $\text{poly}(k)$ is replaced by $2^{\frac{k}{c}}$ for some small constant c . This is best possible in the sense that any function on k bits has a circuit of size 2^k .) We furthermore remark that the requirement on r can in fact be removed from Theorem 1 as explained in the proof. We also stress that the constant $1/3$ can be replaced by any constant smaller than $1/2$.

The bound in Theorem 1 is tight in the sense that there are function-generic reductions showing basic hardness amplification which for $\delta = \Omega(1)$ make $O(1/\epsilon)$ queries [GNW95, IJKW10, Wat10]. (In fact, some of these reductions are for

showing non-Boolean mildly-average-case to average-case hardness amplification). For general δ , these reductions make $O(\frac{\log(1/\delta)}{\epsilon})$ queries. We can improve the bound in Theorem 1 to $\Omega(\frac{\log(1/\delta)}{\epsilon})$ which is tight for every δ . However, we only know how to do this in the special case where the reduction is *non-adaptive*.

By the previous discussion on the relationship between reductions showing various notions of hardness amplification it follows that Theorem 1 applies also for Boolean mildly-average-case to average-case amplification and gives the same lower bound of $\Omega(1/\epsilon)$ on the number of queries. In this setup the best known upper bounds [Imp95, KS03] make $O(\frac{\log(1/\delta)}{\epsilon^2})$ queries. A matching lower bound of $\Omega(\frac{\log(1/\delta)}{\epsilon^2})$ was given in [SV10] for the special case where the reduction R is *non-adaptive*. The argument in [SV10] heavily relies on the non-adaptivity of the reduction. The main contribution of this paper is developing techniques to handle reductions that are both *non-uniform* and *adaptive*, and Theorem 1 is the first bound on such general reductions (of any kind). Most reductions in the literature are non-adaptive, however there are some examples in the literature of adaptive reductions for hardness amplification and related tasks [SU05, GGH07].

Finally, we remark that the technique of [SV10] (which is different than the one used in this paper) can be adapted to the setting of basic hardness amplification (as observed in [Wat10]) showing our aforementioned lower bounds for the special case where the reduction is *non-adaptive*.

Function-specific Hardness Amplification. In contrast to function-generic reductions, non-uniform reductions for specific functions f, g (as defined in Definition 2) are allowed to depend on the choice of functions f, g and their particular properties. It is therefore harder to show lower bounds against such reductions. Moreover, as we now explain, we cannot expect to prove that for every function f, g , every non-uniform reduction R showing basic hardness amplification must use $\Omega(1/\epsilon)$ queries. This is because if f is a function such that there exists a small circuit C that has agreement $1 - \delta$ with f , then there exists a trivial non-uniform reduction R that makes *no queries* as reduction R can ignore its oracle and set $R^{(\cdot)}(x) = C(x)$. Consequently, the best result that we can hope for in this setting is of the form: for every functions f, g and every non-uniform reduction $R^{(\cdot)}$ for f, g , if R makes $o(1/\epsilon)$ queries then there exists a circuit C (with no oracle) of size comparable to that of R that has agreement almost $1 - \delta$ with f . Theorem 2 stated below is of this form.

Theorem 2 (main theorem for function-specific reductions). *Let ϵ, δ and a be parameters. Let $f : \{0, 1\}^k \rightarrow \{0, 1\}$ and $g : \{0, 1\}^n \rightarrow \{0, 1\}^\ell$ be functions. Let $R^{(\cdot)}$ be a non-uniform reduction for f, g, ϵ, δ and a . If R is of size r and makes q queries then for every $\rho \geq 10\epsilon q$ there exists a circuit C of size $r + \text{poly}(a, q, n, 1/\rho)$ that has agreement $1 - \delta - \rho$ with f .*

Theorem 2 says that if $q = o(1/\epsilon)$ then the mere existence of reduction R implies the existence of a circuit C that has agreement $1 - \delta - o(1)$ with f . This can be interpreted as a lower bound on the number of queries in the following sense: Reductions making $o(1/\epsilon)$ queries are not useful as their existence implies that the hardness assumption does not hold.

Function-specific Hardness Amplification in the Literature. Function-specific hardness amplification results are less common than function-generic results. One motivation for developing such results is that function-specific reductions can bypass the coding theoretic objection and be semi-uniform (or even completely uniform). Examples are the reductions in [IW98, TV07, Tre03, Tre05]. Another example is in Cryptography where protocols are often constructed assuming the hardness of some *specific* function (e.g., factoring or discrete log) and properties of this function are used to improve either security or efficiency. Theorem 2 shows that in these settings, reductions must make $\Omega(1/\epsilon)$ queries even if they are non-uniform.

In the function-specific setting there are few examples in the literature of reductions for tasks related to hardness amplification that have proofs not captured by Definition 2. It was pointed out in [GTS07] that the techniques of [GSTS07, Ato06] (that show some worst-case to average-case reduction for NP) are not *black-box* in the sense that we now explain. Semi-uniform reductions are black-box in the sense that R has only black-box access to D . Non-uniform reductions allow R to also get some short advice string α about D . Note that there is no requirement that α is generated using black-box access to D (and this is why we refrain from using the term “black-box” when referring to non-uniform reductions). However, even non-uniform reductions make no assumption about the oracle D and are required to perform for every function D (even if D is not computable by a small circuit). The reductions used in [GSTS07, Ato06] are only guaranteed to perform in case D is efficient, and are therefore not captured by Definition 2. See [GTS07, GV08] for a discussion on such reductions.

1.4 Related Work

We have already surveyed many results on hardness amplification. We now survey some relevant previous work regarding limitations on proof techniques for hardness amplification. We focus on such previous work that is relevant to this paper and the reader is referred to [SV10] for a more comprehensive survey.

The complexity of reductions showing hardness amplification was studied in [SV10, GR08]. Both papers show that function-generic reductions for mildly-average-case to average-case hardness amplification cannot be computed by small constant depth circuits if ϵ is small. Both results fail to rule out general reductions. The result of [GR08] rules out *adaptive* reductions but only if they use very low non-uniformity (meaning that $a = O(\log(1/\epsilon))$ which is much smaller than k in typical settings). The result of [SV10] rules out non-uniform reductions with large non-uniformity (allowing $a = 2^{\Omega(k)}$) but only if they are *non-adaptive*. As mentioned earlier, our results extend previous lower bounds on the number of queries that were proven in [SV10] for *non-adaptive* reductions. This suggests that our techniques may be useful in extending the result of [SV10] regarding constant depth circuits to *adaptive* reductions. We stress however, that we are studying reductions showing *basic* hardness amplification and there are such reductions in the literature that can be computed by small constant depth circuits [LJKW10].

In this paper we are interested in the complexity of function-generic reductions showing hardness amplification. There is an orthogonal line of work [Vio05a, LTW08] that aims to show limitations on “fully-black-box constructions” of hardness amplifications. In our terminology, these are function-generic non-uniform reductions (Amp, R) with the restriction that there exists an oracle machine $M^{(\cdot)}$ called *construction* such that for every function f , $Amp(f)$ is implemented by M^f . The goal in this direction is to prove lower bounds on the complexity of M (which corresponds to encoding), whereas we focus on R (which corresponds to decoding).

There are many other results showing limitations on reductions for hardness amplification and related tasks in various settings. A partial list includes [FF93, TV07, BT06, RTV04, Vio05b, AGGM06, LTW07].

2 Technique

Our high level approach is to extend Rudich’s argument that is explained in the introduction to the case of non-uniform and adaptive reductions. Recall that Rudich’s argument applies to semi-uniform reductions but fails for non-uniform reduction (even if they are non-adaptive). We rely on some of the machinery developed in [SV10], however our overall approach is very different. The approach of [SV10] (which consider non-adaptive function-generic reductions) is to show that the existence of a “too good” function-generic reduction implies a “too good” statistical test that can distinguish between q independent fair coins and q independent biased coins. In contrast, our approach is to show that the existence of a “too good” function-specific reduction yields small circuits for the function f . We do not attempt to mimic the approach of [SV10], as it seems difficult to extend it to adaptive and non-uniform reductions.

The argument of Rudich works by showing that a semi-uniform reduction $R^{(\cdot)}(x)$ that makes $o(1/\epsilon)$ queries, and has access to an oracle D that is chosen uniformly at random amongst all functions that have errorless agreement ϵ with the function f , does not benefit much from querying the oracle. More precisely, that the expected fraction of inputs x which are “silent” (meaning that all queries asked on these inputs answer ‘ \perp ’) is $1 - o(1)$. Thus, by averaging there exists a function D on which $R^D(x)$ has agreement $1 - \delta$ with f , and on almost all inputs, R does not need to query its oracle. This means that R induces a small circuit (with no oracle) that has agreement $1 - \delta - o(1)$ with f .

In the general case of non-uniform and adaptive reductions the reduction R also accepts an advice string α (that may depend on the function D given as oracle). We show that for every such reduction which makes $o(1/\epsilon)$ queries, there exist (i) a fixed advice string α' , (ii) a subset E of all functions that have errorless agreement ϵ with the function f , and (iii) a small set B of possible queries such that the following holds: for every function D in E , the advice string used by R^D is α' . Furthermore, when R receives a uniformly chosen D in E , the expected fraction of inputs x which are “almost silent” (meaning that all queries asked on these inputs are either in B or they answer ‘ \perp ’) is $1 - o(1)$. This is sufficient to

construct a small circuit for f as before (by hardwiring α' , B and the values of g on B , to the circuit R). The main technical difficulty is that reduction R can use its advice string to decide what queries to ask, and then use the answers to these queries (and in particular whether the queries answer ‘ \perp ’ or not) to decide on the next queries it makes.

Due to space limitations this extended abstract does not contain proofs. The reader is referred to the full version for proofs.

3 Conclusion and Open Problem

Our results rule out certain proof techniques for showing hardness amplification results with small “size loss”. As we explain in Section [L.3](#), the framework of reductions that we study captures essentially all hardness amplification results in the literature. Nevertheless, it may be possible to bypass these limitations by developing alternative proof techniques. We remark that the techniques of [\[GSTS07, Ato06\]](#) are not captured in our framework (see Section [L.3](#)).

We now mention a few open problems (continuing the discussion of Section [L.4](#)). Extend the results of [\[SV10\]](#) regarding “necessity of majority” to *adaptive* reductions. More specifically, show that non-uniform and adaptive function-generic reductions for mildly-average-case to average-case hardness amplification cannot be computed by small constant depth circuits if ϵ is small.

Extend the results of [\[SV10\]](#) regarding “number of queries” to *adaptive* reductions. More specifically, show that non-uniform and adaptive function-generic reductions for mildly-average-case to average-case hardness amplification must use $q = \Omega(\frac{\log(1/\delta)}{\epsilon^2})$ queries. (Note that a lower bound of $q = \Omega(1/\epsilon)$ follows from our results on basic hardness amplification).

Our results on basic hardness amplification give a lower bound of $q = \Omega(1/\epsilon)$ for $\delta \leq 1/3$. This meets the known upper bounds for constant δ . However, it seems that the right lower bound should be $q = \Omega(\frac{\log(1/\delta)}{\epsilon})$ and match the known upper bounds of [\[KS03\]](#). We do not know how to show such a bound for non-uniform and adaptive reductions.

Finally, the framework of function-specific reductions suggested in this paper captures more proof techniques than those captured in earlier work. It is natural to study the questions above (as well as related questions in the area) using this more general framework.

Acknowledgements. The second author is grateful to Oded Goldreich, Avi Wigderson and Emanuele Viola for many interesting discussions on hardness amplification. We also thank anonymous referees for helpful comments and suggestions.

References

- [AGGM06] Akavia, A., Goldreich, O., Goldwasser, S., Moshkovitz, D.: On basing one-way functions on np-hardness. In: Proceedings of the 38th Annual ACM Symposium on Theory of Computing, pp. 701–710 (2006)

- [Ats06] Atserias, A.: Distinguishing sat from polynomial-size circuits, through black-box queries. In: IEEE Conference on Computational Complexity, pp. 88–95 (2006)
- [BFNW93] Babai, L., Fortnow, L., Nisan, N., Wigderson, A.: Bpp has subexponential time simulations unless exptime has publishable proofs. *Computational Complexity* 3, 307–318 (1993)
- [BS07] Bogdanov, A., Safra, M.: Hardness amplification for errorless heuristics. In: 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 418–426 (2007)
- [BT06] Bogdanov, A., Trevisan, L.: On worst-case to average-case reductions for np problems. *SIAM J. Comput.* 36(4), 1119–1159 (2006)
- [FF93] Feigenbaum, J., Fortnow, L.: Random-self-reducibility of complete sets. *SIAM J. Comput.* 22(5), 994–1005 (1993)
- [GG11] Gopalan, P., Guruswami, V.: Hardness amplification within NP against deterministic algorithms. *J. Comput. Syst. Sci.* 77(1), 107–121 (2011)
- [GGH07] Goldwasser, S., Gutfreund, D., Healy, A., Kaufman, T., Rothblum, G.N.: Verifying and decoding in constant depth. In: 39th Annual ACM Symposium on Theory of Computing (STOC), pp. 440–449 (2007)
- [GIL90] Goldreich, O., Impagliazzo, R., Levin, L.A., Venkatesan, R., Zuckerman, D.: Security preserving amplification of hardness. In: FOCS, pp. 318–326 (1990)
- [GK08] Guruswami, V., Kabanets, V.: Hardness amplification via space-efficient direct products. *Computational Complexity* 17(4), 475–500 (2008)
- [GNW95] Goldreich, O., Nisan, N., Wigderson, A.: On Yao’s XOR lemma. Technical Report TR95–050, Electronic Colloquium on Computational Complexity (March 1995), <http://www.eccc.uni-trier.de/>
- [GR08] Gutfreund, D., Rothblum, G.: The complexity of local list decoding. In: 12th Intl. Workshop on Randomization and Computation, RANDOM (2008)
- [GSTS07] Gutfreund, D., Shaltiel, R., Ta-Shma, A.: If np languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity* 16(4), 412–441 (2007)
- [GTS07] Gutfreund, D., Ta-Shma, A.: Worst-case to average-case reductions revisited. In: Charikar, M., Jansen, K., Reingold, O., Rolim, J.D.P. (eds.) RANDOM 2007 and APPROX 2007. LNCS, vol. 4627, pp. 569–583. Springer, Heidelberg (2007)
- [GV08] Gutfreund, D., Vadhan, S.P.: Limitations of hardness vs. Randomness under uniform reductions. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 469–482. Springer, Heidelberg (2008)
- [HVV06] Healy, A., Vadhan, S.P., Viola, E.: Using nondeterminism to amplify hardness. *SIAM J. Comput.* 35(4), 903–931 (2006)
- [IJK09a] Impagliazzo, R., Jaiswal, R., Kabanets, V.: Approximate list-decoding of direct product codes and uniform hardness amplification. *SIAM J. Comput.* 39(2), 564–605 (2009)
- [IJK09b] Impagliazzo, R., Jaiswal, R., Kabanets, V.: Chernoff-type direct product theorems. *J. Cryptology* 22(1), 75–92 (2009)
- [IJKW10] Impagliazzo, R., Jaiswal, R., Kabanets, V., Wigderson, A.: Uniform direct product theorems: Simplified, optimized, and derandomized. *SIAM J. Comput.* 39(4), 1637–1665 (2010)

- [Imp95] Impagliazzo, R.: Hard-core distributions for somewhat hard problems. In: FOCS, pp. 538–545 (1995)
- [IW97] Impagliazzo, R., Wigderson, A.: $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In: STOC, pp. 220–229 (1997)
- [IW98] Impagliazzo, R., Wigderson, A.: Randomness vs. time: De-randomization under a uniform assumption. In: 39th Annual Symposium on Foundations of Computer Science. IEEE, Los Alamitos (1998)
- [KS03] Klivans, A., Servedio, R.A.: Boosting and hard-core sets. *Machine Learning* 53(3), 217–238 (2003)
- [Lev87] Levin, L.A.: One-way functions and pseudorandom generators. *Combinatorica* 7(4), 357–363 (1987)
- [Lip91] Lipton, R.: New directions in testing. In: Proceedings of DIMACS Workshop on Distributed Computing and Cryptography, vol. 2, pp. 191–202. ACM/AMS (1991)
- [LTW07] Lu, C.-J., Tsai, S.-C., Wu, H.-L.: On the complexity of hard-core set constructions. In: 34th International Colloquium on Automata, Languages and Programming, pp. 183–194 (2007)
- [LTW08] Lu, C.-J., Tsai, S.-C., Wu, H.-L.: On the complexity of hardness amplification. *IEEE Transactions on Information Theory* 54(10), 4575–4586 (2008)
- [O'D04] O'Donnell, R.: Hardness amplification within np . *J. Comput. Syst. Sci.* 69(1), 68–94 (2004)
- [RTV04] Reingold, O., Trevisan, L., Vadhan, S.P.: Notions of reducibility between cryptographic primitives. In: Naor, M. (ed.) TCC 2004. LNCS, vol. 2951, pp. 1–20. Springer, Heidelberg (2004)
- [STV01] Sudan, M., Trevisan, L., Vadhan, S.P.: Pseudorandom generators without the xor lemma. *J. Comput. Syst. Sci.* 62(2), 236–266 (2001)
- [SU05] Shaltiel, R., Umans, C.: Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM* 52(2), 172–216 (2005)
- [SV10] Shaltiel, R., Viola, E.: Hardness amplification proofs require majority. *SIAM J. Comput.* 39(7), 3122–3154 (2010)
- [Tre03] Trevisan, L.: List-decoding using the xor lemma. In: 44th Symposium on Foundations of Computer Science, pp. 126–135 (2003)
- [Tre04] Trevisan, L.: Some applications of coding theory in computational complexity. In: Complexity of Computations and Proofs. *Quad. Mat.*, vol. 13, pp. 347–424. Dept. Math., Seconda Univ., Napoli (2004)
- [Tre05] Trevisan, L.: On uniform amplification of hardness in np . In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing, pp. 31–38 (2005)
- [TV07] Trevisan, L., Vadhan, S.: Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity* 16(4), 331–364 (2007)
- [Vio05a] Viola, E.: The complexity of constructing pseudorandom generators from hard functions. *Computational Complexity* 13(3-4), 147–188 (2005)
- [Vio05b] Viola, E.: On constructing parallel pseudorandom generators from one-way functions. In: IEEE Conference on Computational Complexity, pp. 183–197 (2005)
- [Wat10] Watson, T.: Query complexity in errorless hardness amplification. In: Electronic Colloquium on Computational Complexity (ECCC), vol. 17, p. 126 (2010)

Testing Graph Blow-Up

Lidor Avigad¹ and Oded Goldreich²

¹ 78 Arlozorov street, Rehovot, Israel
avigadl@gmail.com

² Department of Computer Science, Weizmann Institute of Science, Rehovot, Israel
oded.goldreich@weizmann.ac.il

Abstract. Referring to the query complexity of testing graph properties in the adjacency matrix model, we advance the study of the class of properties that can be tested non-adaptively within complexity that is inversely proportional to the proximity parameter. Arguably, this is the lowest meaningful complexity class in this model, and we show that it contains a very natural class of graph properties. Specifically, for every fixed graph H , we consider the set of all graphs that are obtained by a (possibly unbalanced) blow-up of H . We show a non-adaptive tester of query complexity $\tilde{O}(1/\epsilon)$ that distinguishes graphs that are a blow-up of H from graphs that are ϵ -far from any such blow-up.

Keywords: Property Testing, Adaptivity vs Non-adaptivity, One-sided vs Two-sided Error, Graph Properties, Graph Blow-up.

1 Introduction

The general context of this work is that of testing graph properties in the adjacency matrix representation (as initiated in [GGR]). In this model graphs are viewed as (symmetric) Boolean functions over a domain consisting of all possible vertex-pairs (i.e., an N -vertex graph $G = ([N], E)$ is represented by the function $g : [N] \times [N] \rightarrow \{0, 1\}$ such that $\{u, v\} \in E$ if and only if $g(u, v) = 1$). Consequently, an N -vertex graph represented by the function $g : [N] \times [N] \rightarrow \{0, 1\}$ is said to be ϵ -far from some predetermined graph property if more than $\epsilon \cdot N^2$ entries of g must be modified in order to yield a representation of a graph that has this property. We refer to ϵ as the **proximity parameter**, and the complexity of testing is stated in terms of ϵ and the number of vertices in the graph (i.e., N).

Interestingly, many natural graph properties can be tested within query complexity that depends only on the proximity parameter; see [GGR], which presents testers with query complexity $\text{poly}(1/\epsilon)$, and [AFNS], which characterizes the class of properties that are testable within query complexity that depends only on the proximity parameter (where this dependence may be an arbitrary function of ϵ). A well-known open problem in this area is to characterize the class of graph properties that can be tested within query complexity $\text{poly}(1/\epsilon)$. We mention that such a characterization has been obtained in the special case of induced subgraph freeness properties [AS], but the general case seems quite difficult.

In light of this state of affairs, it was suggested in [GR08] to try to characterize lower query complexity classes, and in particular the class of graph properties that can be tested non-adaptively within query complexity $\tilde{O}(1/\epsilon)$. As a first step towards this goal, it was shown in [GR08, Sec. 6] that, for every constant c , the set of graphs that each consists of at most c isolated cliques is such a property.

In this work we significantly extend the latter result by showing that the class of graph properties that can be tested non-adaptively within query complexity $\tilde{O}(1/\epsilon)$ contains all graph blow-up properties. For any fixed graph $H = ([h], F)$, we say that a graph $G = ([N], E)$ is a **blow-up** of H if the vertices of G can be clustered in up to h clusters such that the edges between these clusters reflect the edge relation of H . That is, vertices in the i^{th} and j^{th} cluster are connected in G if and only if $(i, j) \in F$. Note that, unlike in the case of balanced blow-up (cf. [GKNR]), the clusters are not required to have equal size¹. Also note that the “collection of c cliques” property studied in [GR08, Sec. 6] can be cast as the property of being a blow-up of a c -vertex clique (by considering the complement graph).

Theorem 1.1 (main result): *For every fixed H , the property of being a blow-up of H is testable by $\tilde{O}(1/\epsilon)$ non-adaptive queries. Furthermore, the tester has one-sided error (i.e., it always accepts graphs that are blow-ups of H) and runs in $\text{poly}(1/\epsilon)$ -time.*

We mention that the aforementioned property cannot be tested by $o(1/\epsilon)$ queries, even when adaptivity and two-sided error are allowed (see [GR08, Prop. 6.1]). We also mention that, by [GR08, Prop. 6.2], a tester of $\tilde{O}(1/\epsilon)$ query complexity cannot be canonical (i.e., it cannot rule by inspecting an induced subgraph).

Additional Results. We also consider the complexity of testing “balanced blow-up” properties, showing that the two-sided error query complexity is quadratic in $1/\epsilon$ for both adaptive and non-adaptive testers; see Proposition 2.4. Finally, we present proximity oblivious testers (cf. [GR09]) for any (general) blow-up property; see Theorem 5.2.

Techniques. Theorem 1.1 is proved by presenting a suitable tester and analyzing it. Recall that this tester cannot be canonical; indeed, this tester selects at random a sample of $\tilde{O}(1/\epsilon)$ vertices, but it inspects (or queries) only $\tilde{O}(1/\epsilon)$ of the vertex pairs in this sample. Consequently, the tester (and the analysis) has to deal with partial knowledge of the subgraph induced by the sample. A pivotal notion regarding such partial views is of “inconsistency” between vertices (w.r.t a given partial view), which means that these vertices have different neighbor sets and thus cannot be placed in the same cluster (of a blow-up of H (or any other graph)). Specifically, the tester considers all sets of up to $h + 1$ pairwise inconsistent vertices, and accepts if and only if each such set (along with the

¹ We note that testing balanced blow-up properties requires $\Omega(1/\epsilon^2)$ queries. For details, see Section 2.2.

known incidence relations) can be embedded in H . As usual, the technically challenging part is analyzing the behavior of the tester on arbitrary graphs that are far from being blow-ups of H . Our analysis proceeds in iterations, where in each iteration some progress is made, but this progress is not necessarily reflected by a growing number of incidence constraints but rather in the decreasing density of the violations reflected in the incidence constraints. This progress is captured in Lemma 4.4 (which refers to notions introduced in Section 4.1). Here we merely stress that the number of iterations is polylogarithmic in ϵ^{-1} rather than being $O(h^2)$. (The degree of the polylogarithmic function depends on h .)

Organization. The core of this paper is presented in Sections 3 and 4, which contain a description of the tester and its analysis, respectively. (Indeed, this part establishes Theorem 1.1.) Section 2 provides preliminaries, which may be skipped by the experts, as well as a side discussion (and result) regarding “balanced blow-up” properties. Section 5 provides another secondary discussion; this one regarding proximity oblivious testers. Due to space limitations, three proofs were omitted from the current version; they can be found in the full version of this work [AG].

2 Preliminaries

In this section we review the definition of property testing, when specialized to graph properties in the adjacency matrix model. We also define the blow-up properties (and discuss the case of balanced blow-up).

2.1 Basic Notions

For an integer n , we let $[n] \stackrel{\text{def}}{=} \{1, \dots, n\}$. A generic N -vertex graph is denoted by $G = ([N], E)$, where $E \subseteq \{\{u, v\} : u, v \in [N]\}$ is a set of (unordered) pairs of vertices.² Any set of (such) graphs that is closed under isomorphism is called a **graph property**. By oracle access to such a graph $G = ([N], E)$ we mean oracle access to the Boolean function that answers the query $\{u, v\}$ (or rather $(u, v) \in [N] \times [N]$) with the bit 1 if and only if $\{u, v\} \in E$. At times, we look at E as a subset of $[N] \times [N]$; that is, we often identify E with $\{(u, v) : \{u, v\} \in E\}$.

Definition 2.1 (property testing for graphs in the adjacency matrix model): *A tester for a graph property Π is a probabilistic oracle machine that, on input parameters N and ϵ and access to an N -vertex graph $G = ([N], E)$, outputs a binary verdict that satisfies the following two conditions.*

1. If $G \in \Pi$, then the tester accepts with probability at least $2/3$.
2. If G is ϵ -far from Π , then the tester accepts with probability at most $1/3$, where G is ϵ -far from Π if for every N -vertex graph $G' = ([N], E') \in \Pi$ it holds that the symmetric difference between E and E' has cardinality that is greater than ϵN^2 .

² Thus, we consider *simple* graphs, with no self-loops nor parallel edges.

If the tester accepts every graph in Π with probability 1, then we say that it has one-sided error. A tester is called non-adaptive if it determines all its queries based solely on its internal coin tosses (and the parameters N and ϵ); otherwise it is called adaptive.

The query complexity of a tester is the number of queries it makes to any N -vertex graph oracle, as a function of the parameters N and ϵ . We say that a tester is efficient if it runs in time that is polynomial in its query complexity, where basic operations on elements of $[N]$ are counted at unit cost. We note that all testers presented in this paper are efficient, whereas the lower-bounds hold also for non-efficient testers.

We shall focus on properties that can be tested within query complexity that only depends on the proximity parameter, ϵ . Thus, the query-complexity upper-bounds that we state hold for any values of ϵ and N , but will be meaningful only for $\epsilon > 1/N^2$ or so. In contrast, the lower-bounds (e.g., of $\Omega(1/\epsilon)$) cannot possibly hold for $\epsilon < 1/N^2$, but they will indeed hold for any $\epsilon > N^{-\Omega(1)}$. Alternatively, one may consider the query-complexity as a function of ϵ , where for each fixed value of $\epsilon > 0$ the value of N tends to infinity.

2.2 The Blow-Up Properties

Following the discussion in the introduction, we first define the blow-up properties that are the subject of our study.

Definition 2.2 (graph blow-up): *We say that the graph $G = ([N], E)$ is a blow-up of the graph $H = ([h], F)$ if there is an h -way partition (V_1, \dots, V_h) of the vertices of G such that for every $i, j \in [h]$ and $(u, v) \in V_i \times V_j$ it holds that $(u, v) \in E$ if and only if $(i, j) \in F$. We stress that the V_i 's are not required to be of equal size and that some of them may be empty. We denote by $\mathcal{BU}(H)$ (resp., $\mathcal{BU}_N(H)$) the set of all graphs (resp., N -vertex graphs) that are blow-ups of H .*

In contrast to Definition 2.2, let us briefly consider the more rigid (and popular) definition of a balanced blow-up.

Definition 2.3 (balanced blow-up): *We say that the graph $G = ([N], E)$ is a balanced blow-up of the graph $H = ([h], F)$ if there is an h -way partition (V_1, \dots, V_h) of the vertices of G such that the following two conditions hold:*

1. *For every $i, j \in [h]$ and $(u, v) \in V_i \times V_j$ it holds that $(u, v) \in E$ if and only if $(i, j) \in F$.*
2. *For every $i \in [h]$ it holds that $|V_i| \in \{\lfloor N/h \rfloor, \lceil N/h \rceil\}$.*

We denote by $\mathcal{BBU}(H)$ (resp., $\mathcal{BBU}_N(H)$) the set of all graphs (resp., N -vertex graphs) that are balanced blow-ups of H .

It is easy to see that, except for trivial cases (i.e., when H consists of isolated vertices), balanced blow-up cannot be tested with one-sided error and complexity that does not depend on the size of the graph. The two-sided error testing complexity of this property is $\Theta(1/\epsilon^2)$, as shown next.

Proposition 2.4 (on the complexity of testing balanced blow-up): *For every $H = ([h], F)$ such that $F \neq \emptyset$, testing the property $\mathcal{BU}(H)$ requires $\Omega(1/\epsilon^2)$ queries even if adaptive testers of two sided error are allowed. On the other hand, for any $H = ([h], F)$, there exists a non-adaptive tester of query complexity $O(1/\epsilon^2)$ (and two-sided error) for the property $\mathcal{BU}(H)$.*

The proof can be found in the full version of this work [AG].

3 The $\mathcal{BU}(H)$ -Tester and Its Basic Features

Recall that a tester of the type we seek (i.e., a non-adaptive tester of $\tilde{O}(1/\epsilon)$ query complexity) cannot operate by inspecting an induced subgraph, because by [GR08, Prop. 6.2] such a subgraph will have to be induced by $\Omega(1/\epsilon)$ vertices, which would yield query complexity $\Omega(1/\epsilon^2)$. Thus, like in [GR08, Sec. 6.2], our non-adaptive tester operates by using a less straightforward querying procedure. Specifically, it does select a sample of $\tilde{O}(1/\epsilon)$ vertices, but does *not* query all vertex pairs.

Algorithm 3.1 (testing $\mathcal{BU}(H)$, for a fixed graph $H = ([h], F)$): *On input parameters, N and ϵ , and access to an oracle $g : [N] \times [N] \rightarrow \{0, 1\}$, representing a graph $G = ([N], E)$, the algorithm sets $\ell = \log_2(1/\epsilon) + O(\log \log(1/\epsilon))$ and proceeds as follows.*

1. For every $i \in [\ell]$, it selects uniformly a sample of $\text{poly}(\ell) \cdot 2^i$ vertices, denoted T_i .
Denote $T = \bigcup_{i \in [\ell]} T_i$.
2. For every $i, j \in [\ell]$ such that $i + j \leq \ell$, the algorithm queries all pairs in $T_i \times T_j$.
3. The algorithm accepts if and only if the answers obtained in Step 2 are consistent with some blow-up of H . That is, let $K : T \times T \rightarrow \{0, 1, *\}$ be a partial description of the subgraph of G induced by T such that $K(u, v) = g(u, v)$ if query (u, v) was made in Step 2, and otherwise $K(u, v) = *$. Then, the acceptance condition seeks a mapping $\phi : T \rightarrow [h]$ such that if $K(u, v) = 1$ then $(\phi(u), \phi(v)) \in F$ and if $K(u, v) = 0$ then $(\phi(u), \phi(v)) \notin F$.

Indeed, at this point we ignore the computational complexity of implementing Step 3. We shall return to this issue at the end of the current section. But, first, let us note that the query complexity of Algorithm 3.1 is

$$\sum_{i, j: i+j \leq \ell} \text{poly}(\ell) \cdot 2^{i+j} = \text{poly}(\ell) \cdot 2^\ell = \tilde{O}(1/\epsilon). \tag{1}$$

It is also clear that Algorithm 3.1 is non-adaptive and that it accepts every $G \in \mathcal{BU}(H)$ with probability 1 (i.e., it has one-sided error). The bulk of this work (see Section 4) is devoted to showing that if G is ϵ -far from $\mathcal{BU}(H)$, then Algorithm 3.1 rejects it with probability at least $2/3$.

Relaxing the Acceptance Condition of Algorithm 3.7. A straightforward implementation of Step 3 amounts to considering all $h^{|T|}$ mappings of T to $[h]$, and checking for each such mapping ϕ whether the clustering induced by ϕ fits the graph H . Relaxing the acceptance condition (used in Step 3 of Algorithm 3.1) yields a more time-efficient algorithm. Actually, the relaxed acceptance condition (defined next) seems easier to analyze than the original one. The notion of *pairwise inconsistent rows* (of K) is pivotal to this relaxed acceptance condition. (Indeed, it will be instructive to think of K as a matrix, and to view rectangular restrictions of K as sub-matrices.)

Definition 3.2 (pairwise inconsistent rows): *Let $K' : R \times C \rightarrow \{0, 1, *\}$ be a sub-matrix of $K : T \times T \rightarrow \{0, 1, *\}$; that is, $R, C \subseteq T$ and $K'(r, c) = K(r, c)$ for every $(r, c) \in R \times C$. Then, the rows $r_1, r_2 \in R$ are said to be inconsistent (wrt K') if there exists a column $c \in C$ such that $K'(r_1, c)$ and $K'(r_2, c)$ are different Boolean values (i.e., $K'(r_1, c), K'(r_2, c) \in \{0, 1\}$ and $K'(r_1, c) \neq K'(r_2, c)$). A set of rows of K' is called pairwise inconsistent (wrt K') if each pairs of rows is inconsistent (wrt K').*

Another pivotal notion, which was alluded to before, is the notion of being consistent with some blow-up of H , which we now term *H-mappability*.

Definition 3.3 (*H-mappable sub-matrices*): *Let $K' : R \times C \rightarrow \{0, 1, *\}$ be a sub-matrix of $K : T \times T \rightarrow \{0, 1, *\}$. We say that K' is *H-mappable* if there exists a mapping $\phi : R \rightarrow [h]$ such that if $K'(u, v) = 1$ then $(\phi(u), \phi(v)) \in F$ and if $K'(u, v) = 0$ then $(\phi(u), \phi(v)) \notin F$. We call such a ϕ an *H-mapping* of K' (or R) to $[h]$.*

Note that if K is *H-mappable*, then every two *inconsistent* rows of K must be mapped (by ϕ as in Definition 3.3) to different vertices of H . In particular, if a sub-matrix $K' : R \times C \rightarrow \{0, 1, *\}$ of K has pairwise inconsistent rows, then any *H-mapping* of K to $[h]$ must be injective. Hence, if K contains more than h pairwise inconsistent rows, then K is not *H-mappable*.

Definition 3.4 (the relaxed acceptance condition (of Algorithm 3.1)): *The relaxed algorithm accept if and only if each set of pairwise inconsistent rows in K is *H-mappable*. That is, for every set R of pairwise inconsistent rows in K , we check whether the sub-matrix $K' : R \times T \rightarrow \{0, 1, *\}$ is *H-mappable*, where the pairwise inconsistency condition mandates that this mapping of R to $[h]$ is 1-1. In particular, if K has more than h pairwise inconsistent rows, then the relaxed acceptance condition fails.*

Note that the relaxed acceptance condition can be checked by considering all s -subsets of T , for all $s \leq h + 1$. For each such subset that consists of pairwise inconsistent rows, we consider all possible 1-1 mappings of this subset to $[h]$, and check consistency with respect to H . This can be performed in time $\binom{|T|}{h+1} \cdot (h!) < |T|^{h+1} = \text{poly}(1/\epsilon)$, where the polynomial depends on h .

Clearly, if $G \in \mathcal{BU}(H)$, then for every $T \subseteq [N]$ it holds that the corresponding matrix K satisfies Definition 3.4. Thus, the relaxed algorithm always accepts graphs in $\mathcal{BU}(H)$. Section 4 is devoted to showing that if G is ϵ -far from $\mathcal{BU}(H)$, then the relaxed algorithm rejects with high probability.

4 The Acceptance Condition and Graphs That Are Far from $\mathcal{BU}(H)$

In light of the above, Theorem 1.1 follows from the fact that the relaxed version of Algorithm 3.1 (which uses the condition in Definition 3.4) rejects with very high probability any graph G that is ϵ -far from $\mathcal{BU}(H)$. This fact is established next.

Lemma 4.1 (main lemma): *Suppose that $G = ([N], E)$ is ϵ -far from $\mathcal{BU}_N(H)$, and let $T = \bigcup_{i \in [\ell]} T_i$ be selected at random as in Step 1 of Algorithm 3.1. Then, with probability at least $2/3$, there exists a set $R \subset T$ of pairwise inconsistent rows in the corresponding matrix $K : T \times T \rightarrow \{0, 1, *\}$ that is not H -mappable.*

Before embarking on the actual proof of Lemma 4.1, we provide a very rough outline.

Outline of the Proof of Lemma 4.1. Our very rough plan of action is to partition the selection of T (and each of its parts, i.e., T_0, T_1, \dots, T_ℓ) into $p(\ell) \stackrel{\text{def}}{=} 2^{\ell h}$ many phases such that in the j^{th} phase we select at random samples $T_0^j, T_1^j, \dots, T_\ell^j$ such that $|T_i^j| = \text{poly}(\ell) \cdot 2^i$. Thus, we let each T_i equal $\bigcup_{j=1}^{p(\ell)} T_i^j$, but we shall consider the queries as if they are made in phases such that in the j^{th} phase we only consider queries between $T^j \stackrel{\text{def}}{=} \bigcup_{i \in [\ell]} T_i^j$ and $T^{[j]} \stackrel{\text{def}}{=} \bigcup_{k \leq j} T^k$. Letting $K^j : T^{[j]} \times T^{[j]} \rightarrow \{0, 1, *\}$ denote the partial information obtained on G in the first j phases, we consider a certain set R^j of pairwise inconsistent rows of K^j . If this set R^j is not H -mappable, then we are done. Otherwise, we show that, with high probability over the choice of the sample T^{j+1} , we obtain a new set R^{j+1} of pairwise inconsistent rows such that R^{j+1} has a higher *index* than R^j , where the indices refer to an order over sequences of length at most h over $[\ell]$. Since the number of such sequences is $\sum_{k \in [h]} \ell^k < p(\ell)$, with high probability, this process must reach a set R^j that is not H -mappable, and so we are done.

Needless to say, the crucial issue is the progress achieved in each phase; that is, the fact that at each phase j the index of the new set R^{j+1} is higher than the index of the old set R^j . Intuitively, this progress is achieved because the current (H -mappable) set R^j induces a clustering of all vertices of G that extends this H -mapping, whereas this clustering must contain many vertex pairs that violate the edge relation of H . The sample taken in the current phase (i.e., T^{j+1}) is likely to hit these violations, and this gives rise to a set R^{j+1} with higher index.

4.1 Basic Notions and Notations

In addition to the foregoing notations, T_i^j, T^j and $T^{[j]}$, we shall use the following notations.

- A pair (R, C) is called a j -basic pair if $C \subseteq T^{[j]}$ and $R \subseteq C$. Indeed, j -basic pairs correspond to restrictions of the sample available at phase j (i.e., $T^{[j]}$).
- The j -index of a vertex $v \in T^{[j]}$, denoted $\text{idx}^j(v)$, is the smallest index i such that $v \in T_i^{[j]}$, where $T_i^{[j]} \stackrel{\text{def}}{=} \bigcup_{k \leq j} T_i^k$. (Note that $\text{idx}(\cdot)$ depends on T , but this dependence is not shown in the notation.)

A key observation is that for every $u, v \in T$, it holds that $K(u, v) = g(u, v)$ if and only if $\text{idx}^{p(\ell)}(u) + \text{idx}^{p(\ell)}(v) \leq \ell$. Otherwise, $K(u, v) = *$ (indicating that (u, v) was not queried by Algorithm 3.1).

We comment that, with extremely high probability, for each j and $v \in T^{[j]}$, there exists a unique $i \in [\ell]$ and $k \in [j]$ such that $v \in T_i^k$. Thus, for any $v \in T^{[j]}$, we may assume that $\text{idx}^{j+1}(v) = \text{idx}^j(v)$.

- The indices of individual vertices in $T^{[j]}$ are the basis for defining the index of sets in $T^{[j]}$. Specifically, the j -index of a set $S \subseteq T^{[j]}$, denoted $\text{idx}^j(S)$, is the multi-set consisting of all values $\text{idx}^j(v)$ for $v \in S$. It will be instructive to consider an ordered version of this multi-set; that is, we redefine $\text{idx}^j(S)$ as $(i_1, \dots, i_{|S|})$ such that (1) for every $k < |S|$ it holds that $i_k \geq i_{k+1}$, and (2) for every $i \in [\ell]$ it holds that $|\{k \in [|S|] : i_k = i\}| = |\{v \in S : \text{idx}^j(v) = i\}|$.
- We consider a natural lexicographic order over sequences, denoted \succ , such that for two (monotonically non-increasing) sequences of integers, $a = (a_1, \dots, a_m)$ and $b = (b_1, \dots, b_n)$, it holds that $a \succ b$ if
 - either there exists $i \leq \min(n, m)$ such that $(a_1, \dots, a_{i-1}) = (b_1, \dots, b_{i-1})$ and $a_i > b_i$.
 - or $m > n$ and $(a_1, \dots, a_n) = (b_1, \dots, b_n)$.

Note that \succ is a total order on the set of monotonically non-increasing (finite) sequences of integers.

As hinted in the overview, a key notion in our analysis is the notion of a clustering of the vertices of G that is induced by an H -mapping of some small subset of vertices. Actually, the clustering is induced by a partial knowledge sub-matrix $K' : R \times C \rightarrow \{0, 1, *\}$ as follows.

Definition 4.2 (the clustering induced by K'): *Let $K' : R \times C \rightarrow \{0, 1, *\}$ be a sub-matrix of $K : T \times T \rightarrow \{0, 1, *\}$ such that K' has pairwise inconsistent rows. Then, for every $r \in R$, we denote by $V_r(K')$ the set of vertices $v \in [N]$ that are consistent with row r in K' . That is,*

$$V_r(K') \stackrel{\text{def}}{=} \{v \in [N] : (\forall c \in C) g(v, c) \cong K'(r, c)\} \tag{2}$$

where, for $\sigma, \tau \in \{0, 1, *\}$, we write $\sigma \cong \tau$ if either $\sigma = \tau$ or $\sigma = *$ or $\tau = *$. The vertices that are inconsistent with all rows, are placed in the leftover set $L(K') \stackrel{\text{def}}{=} [N] \setminus \bigcup_{r \in R} V_r(K')$.

Indeed, rows $r_1, r_2 \in R$ are inconsistent wrt K' (as per Definition 3.2) if there exists a column $c \in C$ such that $K'(r_1, c) \not\equiv K'(r_2, c)$ (which means that $K'(r_1, c)$ and $K'(r_2, c)$ are both in $\{0, 1\}$ but are different). Thus, the hypothesis that the rows of K' are pairwise inconsistent implies that the sets in Eq. (2) are disjoint. Hence, the clustering in Definition 4.2 is indeed a partition of the vertex set of G (since $v \in L(K')$ if for every $r \in R$ there exists $c \in C$ such that $g(v, c) \not\equiv K'(r, c)$). This motivates our focus on sub-matrices having pairwise inconsistent rows. The following definition adds a requirement (regarding such sub-matrices) that refers to the relation between the index of row r and the density of the corresponding set $V_r(K')$.

Definition 4.3 (nice pairs): *Let (R, C) be a j -basic pair and $K' : R \times C \rightarrow \{0, 1, *\}$ be the corresponding sub-matrix of K . We say that (R, C) is a j -nice pair if the following two conditions hold.*

1. R is pairwise inconsistent with respect to K' .
2. For every $r \in R$ it holds that $\text{ind}^j(r) \leq \rho(V_r(K')) + 1$, where $\rho(S) \stackrel{\text{def}}{=} \lceil \log(N/|S|) \rceil$.

As a sanity check, suppose that $r \in R$ was selected in phase j (i.e., $r \in T^j$). Then, it is very likely that r (or some other member of $V_r(K')$) is selected in $T^j_{\rho(V_r(K'))-1}$, because $T^j_{\rho(V_r(K'))-1}$ is a random set of cardinality $\text{poly}(\ell) \cdot 2^{\rho(V_r(K'))-1} = \text{poly}(\ell) \cdot N/|V_r(K')|$.

For each phase j , we shall show the existence of a j -nice pair. Furthermore, we shall show that the corresponding set of rows has a higher index than all sets of rows associated with previous phases. The furthermore claim is the crux of the analysis, and is captured by the Progress Lemma presented in Section 4.2. But let us first establish the mere existence of j -nice pairs. Indeed, for every $j \geq 1$, we may pick an arbitrary $r \in T^j_1$, and consider the j -nice pair $(\{r\}, \{r\})$, while noting that $\text{idx}^1(r) = 1$ and $\rho(V_r(K')) \geq 0$ (where $K' : \{r\} \times \{r\} \rightarrow \{0, 1, *\}$).

4.2 The Progress Lemma

Recall that $G = ([N], E)$ is ϵ -far from $\mathcal{BU}(H)$, where $H = ([h], F)$. Furthermore, we consider the partial view $K : T \times T \rightarrow \{0, 1, *\}$ obtained by Algorithm 3.1, where $T = \bigcup_{i \in [\ell], j \in [p(\ell)]} T^j_i$ is the random sample selected.

Lemma 4.4 (Progress Lemma): *Let (R, C) be a j -nice pair and $K' : R \times C \rightarrow \{0, 1, *\}$ be the corresponding sub-matrix of K . If K' is H -mappable then, with overwhelmingly high probability³ over the choice of T^{j+1} , there exists a $(j + 1)$ -nice pair (R', C') such that $\text{ind}^{j+1}(R') \succ \text{ind}^j(R)$.*

Recalling that a (trivial) 1-nice pair always exists and that the number of possible indices is smaller than $p(\ell)$, we conclude that, with overwhelmingly high probability (over the choice of T), there exists a $j < p(\ell)$ and a j -nice pair that is not H -mappable. Lemma 4.1 follows. Thus, all that remains is proving Lemma 4.4, which is undertaken in the full version of this work [AG].

³ I.e., with probability that exceeds $1 - q(\epsilon)$, for any polynomial q .

5 Proximity Oblivious Testing of Blow-Up

In this section we derive, for every fixed graph H , a constant-query proximity oblivious tester of $\mathcal{BU}(H)$. That is, we refer to the following definition of [GR09], when specialized to the dense graph model.

Definition 5.1 (proximity oblivious testing for graphs in the adjacency matrix model): *A proximity oblivious tester for a graph property Π is a probabilistic oracle machine that, on input parameter N and access to an N -vertex graph $G = ([N], E)$, outputs a binary verdict that satisfies the following two conditions.*

1. *If $G \in \Pi$, then the tester accepts with probability 1.*
2. *There exists a monotone function $\rho : (0, 1] \rightarrow (0, 1]$ such that, for every graph $G = ([N], E) \notin \Pi$, it holds that the tester rejects G with probability at least $\rho(\delta_\Pi(G))$, where $\delta_\Pi(G)$ denotes the (relative) distance of G from the set of N -vertex graphs that are in Π .*

The function ρ is called the **detection probability** of the tester.

Combining Lemma 4.1 and the ideas underlying [GR09, Thm. 6.3], we obtain.

Theorem 5.2 *For every fixed graph $H = ([h], F)$, there exists a $O(h^2)$ -query proximity oblivious tester of $\mathcal{BU}(H)$. Furthermore, the tester has detection probability $\rho(\epsilon) = \epsilon^{O(h)}$.*

This extends the result of [GR09, Prob. 4.11], which corresponds to the special case in which H is a h -vertex clique. We also mention that, for constant-query proximity oblivious testers of $\mathcal{BU}(H)$, detection probability of the form $\rho(\epsilon) = \epsilon^{\Omega(h)}$ is essential (cf. [GR09, Prob. 4.3]). The proof of Theorem 5.2 can be found in the full version of this work [AG].

6 Conclusions

We have shown a non-adaptive tester of query complexity $\tilde{O}(1/\epsilon)$ for $\mathcal{BU}(H)$. The degree of the polynomial in the polylogarithmic factor that is hidden in the $\tilde{O}()$ notation is $h + O(1)$, where h is the number of vertices in H . We wonder whether the query complexity can be reduced to $p(h \log(1/\epsilon)) \cdot \epsilon^{-1}$, where p is a fixed polynomial. We mention that such a dependence on h was obtained in [GR08, Sec. 6.2] for the special case in which H is an h -clique. Furthermore, we wonder whether non-adaptive testing of $\mathcal{BU}(H)$ is possible in query complexity $\text{poly}(h) \cdot \epsilon^{-1}$. We mention that such a result is only known for $h = 2$ (cf. [GR08, Sec. 6.1]), whereas an *adaptive* tester of query complexity $O(h^2/\epsilon)$ is known (cf. [A, Sec. 4]).

Acknowledgments. This work is based on the M.Sc. thesis of the first author [A], which was completed under the supervision of the second author. The work was partially supported by the Israel Science Foundation (grant No. 1041/08). We are grateful to Dana Ron and to the anonymous reviewers of *RANDOM'11* for comments regarding previous versions of this work.

References

- [AFKS] Alon, N., Fischer, E., Krivelevich, M., Szegedy, M.: Efficient Testing of Large Graphs. *Combinatorica* 20, 451–476 (1999)
- [AFNS] Alon, N., Fischer, E., Newman, I., Shapira, A.: A Combinatorial Characterization of the Testable Graph Properties: It’s All About Regularity. In: 38th STOC, pp. 251–260 (2006)
- [AS] Alon, N., Shapira, A.: A Characterization of Easily Testable Induced Subgraphs. *Combinatorics Probability and Computing* 15, 791–805 (2006)
- [A] Avigad, L.: On the Lowest Level of Query Complexity in Testing Graph Properties. Master thesis, Weizmann Institute of Science (December 2009)
- [AG] Avigad, L., Goldreich, O.: Testing graph blow-up. In: Goldreich, O. (ed.) *Studies in Complexity and Cryptography*. LNCS, vol. 6650, pp. 156–172. Springer, Heidelberg (2011)
- [CEG] Canetti, R., Even, G., Goldreich, O.: Lower Bounds for Sampling Algorithms for Estimating the Average. In: *IPL*, vol. 53, pp. 17–25 (1995)
- [GGR] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *Journal of the ACM*, 653–750 (July 1998)
- [GKNR] Goldreich, O., Krivelevich, M., Newman, I., Rozenberg, E.: Hierarchy Theorems for Property Testing. *ECCC*, TR08-097 (2008); Extended Abstract in the *Proceedings of RANDOM* (2009)
- [GR08] Goldreich, O., Ron, D.: Algorithmic Aspects of Property Testing in the Dense Graphs Model. *ECCC*, TR08-039 (2008)
- [GR09] Goldreich, O., Ron, D.: On Proximity Oblivious Testing. *ECCC*, TR08-041 (2008); Extended Abstract in the *Proceedings of the 41st STOC* (2009)
- [GT] Goldreich, O., Trevisan, L.: Three theorems regarding testing graph properties. *Random Structures and Algorithms* 23(1), 23–57 (2003)
- [RS] Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing* 25(2), 252–271 (1996)

On Sums of Locally Testable Affine Invariant Properties

Eli Ben-Sasson^{*}, Elena Grigorescu^{**}, Ghid Maatouk^{***}, Amir Shpilka[†],
and Madhu Sudan[‡]

Abstract. Affine-invariant properties are an abstract class of properties that generalize some central algebraic ones, such as linearity and low-degree-ness, that have been studied extensively in the context of property testing. Affine invariant properties consider functions mapping a big field \mathbb{F}_{q^n} to the subfield \mathbb{F}_q and include all properties that form an \mathbb{F}_q -vector space and are invariant under affine transformations of the domain. Almost all the known locally testable affine-invariant properties have so-called “single-orbit characterizations” — namely they are specified by a single local constraint on the property, and the “orbit” of this constraint, i.e., translations of this constraint induced by affine-invariance. Single-orbit characterizations by a local constraint are also known to imply local testability. In this work we show that properties with single-orbit characterizations are closed under “summation”. To complement this result, we also show that the property of being an n -variate low-degree polynomial over \mathbb{F}_q has a single-orbit characterization (even when the domain is viewed as \mathbb{F}_{q^n} and so has very few affine transformations). As a consequence we find that the sum of any sparse affine-invariant property (properties satisfied by $q^{O(n)}$ -functions) with the set of degree d multivariate polynomials over \mathbb{F}_q has a single-orbit characterization (and is hence locally testable) when q is prime. We conclude with some intriguing questions/conjectures attempting to classify all locally testable affine-invariant properties.

Keywords: Property testing, Symmetries, Direct sums, Error-correcting codes.

^{*} Department of Computer Science, Technion — Israel Institute of Technology, Haifa, Israel, eli@cs.technion.ac.il. Research funded partially by the European Community’s Seventh Framework Programme (FP7/2007-2013) Grant 240258 and the US-Israel Binational Science Foundation Grant 2006104.

^{**} Georgia Tech, Atlanta, GA, elena@cc.gatech.edu. Supported in part by NSF award CCR-0829672 and NSF award 1019343 to the Computing Research Association for the Computing Innovation Fellowship Program.

^{***} School of Computer and Communications Sciences, EPFL, Switzerland, ghid.maatouk@epfl.ch. Part of this work was conducted while at Microsoft Research. Supported in part by Grant 228021-ECCSciEng of the European Research Council.

[†] Faculty of Computer Science, Technion — Israel Institute of Technology, Haifa, Israel and Microsoft Research, Cambridge, MA, shpilka@cs.technion.ac.il. Research partially supported by the Israel Science Foundation Grant 339/10.

[‡] Microsoft Research New England, Cambridge, Massachusetts, USA madhu@mit.edu.

1 Introduction

Given finite sets D and R , let $\{D \rightarrow R\}$ denote the set of functions mapping D to R . A *property* \mathcal{F} of functions mapping D to R is simply given by a set $\mathcal{F} \subseteq \{D \rightarrow R\}$. The goal of property testing [20,12] is to design “query efficient” tests for various properties. Specifically, a (k, ϵ, δ) -tester for \mathcal{F} is a probabilistic oracle algorithm that, given oracle access to a function $f : D \rightarrow R$, makes k -queries to f and accepts $f \in \mathcal{F}$ with probability one, while rejecting f that is δ -far from \mathcal{F} with probability at least ϵ . Here, distance is measured by normalized Hamming distance: $\delta(f, g) = |\{x \in D \mid f(x) \neq g(x)\}|/|D|$ denotes the distance between f and g , and $\delta(f, \mathcal{F}) = \min_{g \in \mathcal{F}} \{\delta(f, g)\}$. f is said to be δ -far from \mathcal{F} if $\delta(f, \mathcal{F}) > \delta$ and δ -close otherwise. To minimize notation we say \mathcal{F} is k -locally testable if for every $\delta > 0$ there exists $\epsilon = \epsilon(k, \delta) > 0$ such that \mathcal{F} is (k, ϵ, δ) -locally testable. Our interest is in families of properties that are k -locally testable for some constant k .

In this work we consider testing of “affine-invariant (linear) properties”. The domain and range of such properties are fields. Let \mathbb{F}_q denote the field of size q and let \mathbb{F}_q^* denote the non-zero elements of \mathbb{F}_q . We consider properties $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ (so q is a prime power and n is a positive integer). \mathcal{F} is *linear* if for every $f, g \in \mathcal{F}$ and $\alpha \in \mathbb{F}_q$, the function $\alpha \cdot f + g$ belongs to \mathcal{F} , where $(\alpha \cdot f + g)(x) = \alpha \cdot f(x) + g(x)$. A function $A : \mathbb{F}_{q^n} \rightarrow \mathbb{F}_{q^n}$ is *affine* if there exist $\alpha, \beta \in \mathbb{F}_{q^n}$ such that $A(x) = \alpha x + \beta$. We say A is an *affine permutation* if A is affine and bijective. Note this is equivalent to saying $A(x) = \alpha x + \beta$ for some $\alpha \in \mathbb{F}_{q^n}^*$ and $\beta \in \mathbb{F}_{q^n}$. A property \mathcal{F} is said to be *affine-invariant* if for $f \in \mathcal{F}$ and every affine permutation $A : \mathbb{F}_{q^n} \rightarrow \mathbb{F}_{q^n}$, the function $f \circ A$ is also in \mathcal{F} , where $(f \circ A)(x) = f(A(x))$.¹

The main contribution of this work is to describe a new class of affine-invariant properties that are locally testable. We show that a broad class of locally testable affine-invariant properties (one that includes most known ones) is closed under “sums”. But before presenting our results, we motivate the study of affine-invariant properties briefly.

Motivation: The study of affine-invariance was originally motivated in [19] by its connections to locally testable codes and to property testing (cf. the recent survey [21]). Indeed, many “base-constructions” of locally testable codes — crucially used in constructing probabilistically checkable proofs [4,3] — are algebraic in nature and come from families of low-degree polynomials. This motivates the search for the minimal algebraic requirements sufficient to obtain families of locally testable codes, and affine-invariance offers a rich and interesting framework in which to study abstract properties shared by low-degree functions and other algebraic locally testable properties. In this respect, the study of affine-invariant

¹ In all previous works starting with [19], affine-invariance was defined as invariance with respect to all affine functions, and not only with respect to affine permutations. In this paper, we define affine-invariance as invariance with respect to the group of affine-permutations. Fortunately, the class of properties does not change despite the mild change in the definition. We prove this equivalence in the full version [7].

property testing is similar to the study of graph property testing initiated by [12]. Graph-property testing abstracts and unifies properties such as k -colorability and triangle-free-ness, by focussing only on the invariance induced by being a “graph property” (i.e., the property should remain invariant under renaming of the vertices). Affine-invariant testing similarly attempts to abstract and unify algebraic properties such as being linear or of low-degree or a BCH codeword by focussing only on the invariance of the property (and the linearity of the code/property). The study of graph property testing however is much further advanced and has culminated in a complete combinatorial characterization of locally-testable properties in the “dense-graph model” [110]. Testing of affine-invariant properties lacks such a characterization and indeed it is not yet clear what shape such a characterization might take.

An additional reason to study affine-invariant properties is because they correspond to *locally correctable codes*. An error correcting code of blocklength n is said to be locally correctable if it has an associated “local corrector”. Given an adversarially corrupted codeword $w \in \mathbb{F}_q^n$ and index $i \in \{1, \dots, n\}$ the (randomized) local corrector makes a *constant* number (hence it is called “local”) of queries to entries of w and outputs, with high probability, the i th entry of the “correct” codeword w' — closest in Hamming distance to w . Linear codes that are locally correctable are easily seen to be locally decodable codes as defined by [16] and can be used to construct databases that support private information retrieval [11] (in general though, local correctability is a stronger property than local decodability, see e.g. [65]). It can be verified that affine-invariant locally testable codes are in fact locally correctable [19] hence our results imply new families of locally correctable (and decodable) codes.

Known Testable Properties: Previous works have shown local testability results for two broad categories of affine-invariant properties: (1) Reed-Muller properties, and (2) Sparse properties.

In our language, Reed-Muller properties are obtained by equating the sets \mathbb{F}_q^n and \mathbb{F}_q^n with an \mathbb{F}_q -linear bijection. This allows us to view \mathbb{F}_q -linear subspaces of $\{\mathbb{F}_q^n \rightarrow \mathbb{F}_q\}$ as linear subspaces of $\{\mathbb{F}_q^n \rightarrow \mathbb{F}_q\}$ where the latter is the set of n -variate functions over \mathbb{F}_q . The q -ary Reed-Muller property of weight degree w is given by the set of functions that are n -variate polynomials of degree at most w in this view. The testing result here shows that the Reed-Muller property with parameter w over \mathbb{F}_q is testable with $q^{O(w/q)}$ queries [17] (see also [2,15]), independent of n .

Sparse properties are less structured ones. Roughly, a property is t -sparse if it is of size at most $q^{O(tn)}$. The main theorem here, due to [18] shows that for every prime q and integer t there exists k , such that for every n every t -sparse $\mathcal{F} \subseteq \{\mathbb{F}_q^n \rightarrow \mathbb{F}_q\}$ is k -locally testable.

Aside from the classes above, the known testable properties are less “explicit” and are derived from the concept of single-orbit characterizations, described next.

Single-orbit characterizations: Local tests of linear properties work by picking k query locations $\alpha_1, \dots, \alpha_k \in \mathbb{F}_q^n$ (non-adaptively) and then verifying that

$f(\alpha_1), \dots, f(\alpha_k)$ satisfy some given constraint (which will restrict this k -tuple to satisfy some linear constraints over \mathbb{F}_q). If a property is affine-invariant, it should be equally effective to query $A(\alpha_1), \dots, A(\alpha_k)$ for some affine permutation A , and then test to see if the function values at these points also satisfy the given constraint. The collection of tests so obtained (by trying out all A s) is referred to as the *orbit* of the constraint at $\alpha_1, \dots, \alpha_k$. If the only functions that satisfy all these constraints are the functions in \mathcal{F} , then we say that \mathcal{F} has a *single orbit characterization*.

Single-orbit characterizations seem to be playing a central role in testing of affine-invariant properties. On the one hand, it is known that every k -single-orbit characterized property is k -locally testable [19] and some non-single-orbit characterized properties are known to be not locally-testable even though they can be characterized by a collection of k -local constraints [8]. On the other hand, most known locally testable properties also seem to have some “single-orbit” property. Sparse codes over prime fields were shown to be single-orbit characterized in [18] (see also [14]). The Reed-Muller property has the single orbit property over the (large) group of affine transformations over the vector space \mathbb{F}_q^n by natural considerations. (This will be insufficient for our purposes and so we will strengthen it to get a single-orbit characterization over the field \mathbb{F}_{q^n} in this work.)

Remaining cases of known locally testable codes are obtained in one of two ways: (1) By lifting: This is an operation introduced in [8]. Here we start with a single-orbit property over some field \mathbb{F}_{q^n} and then “lift” this property to one over an extension field $\mathbb{F}_{q^{nm}}$ (in a manner we will describe later). (2) By taking intersections: The intersection of testable properties is always testable. The lifts turn out to be single-orbit characterized by definition, and the intersection of a constant number of single-orbit characterized properties also turns out to be single-orbit characterized essentially by definition.

1.1 Main Result

In this work we extend the class of properties over \mathbb{F}_{q^n} that have single orbit characterizations.

Our first result considers the sum of affine invariant properties. For properties $\mathcal{F}_1, \mathcal{F}_2 \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ their *sum* is $\mathcal{F}_1 + \mathcal{F}_2 = \{f_1 + f_2 \mid f_1 \in \mathcal{F}_1, f_2 \in \mathcal{F}_2\}$. For general linear properties $\mathcal{F}_1 + \mathcal{F}_2$ is also linear, but the testability of $\mathcal{F}_1, \mathcal{F}_2$ does not imply their sum is locally testable. Indeed it may be the case that $\mathcal{F}_1 + \mathcal{F}_2$ satisfies no local constraints. Sums of affine-invariant properties behave more nicely. It is straightforward to see the the sum of affine-invariant properties is affine-invariant. More interestingly, it is also possible to show (relatively easily) that if for every $i \in \{1, 2\}$, \mathcal{F}_i satisfies a k_i -local constraint, then $\mathcal{F}_1 + \mathcal{F}_2$ satisfies a $k_1 \cdot k_2$ -local constraint. However this does not seem to imply local-testability. Here we focus on single-orbit characterized properties and prove their sum is single-orbit characterized.

Theorem 1. *For every q, k_1, k_2 , there exists $\kappa = \kappa(k_1, k_2, q)$ such that for every n , if $\mathcal{F}_1, \mathcal{F}_2 \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ are affine-invariant properties with \mathcal{F}_i having a k_i -single orbit characterization, then $\mathcal{F}_1 + \mathcal{F}_2$ has a κ -single orbit characterization. Specifically, if $n \geq n_0 = 10k^2 \log k + 10$, where $k = \max\{k_1, k_2\}$, then we can set $\kappa = k_1 \cdot k_2$, else $\kappa = q^{n_0}$ works.*

To apply the theorem above to get new families of single-orbit characterized properties, we need good base properties. However, the two families mentioned earlier, sparse properties and Reed-Muller properties were not known to have the single-orbit property over the same group. Reed-Muller properties were known to have the single-orbit property over the group of affine permutations over \mathbb{F}_q^n , while sparse properties are invariant only over \mathbb{F}_{q^n} . (And there is no point using the theorem above to prove that the sum of two sparse families is single-orbit — this is already known since the sum of sparse families is also sparse!) To remedy this situation we show that the Reed-Muller property is actually single orbit over the group of affine permutations over \mathbb{F}_{q^n} .

Theorem 2 (Reed-Muller codes have local single-orbit property). *Let $q = p^s$ be a prime power. Let w, n be integers such that $w + 1 < \sqrt{\frac{n}{\log_q(3ns)}}$. Denote $w + 1 = r(p - 1) + \ell$, where $0 \leq \ell < p - 1$. Then, the q -ary Reed-Muller family of weight degree w , $\text{RM}_q(w, n)$, has a k -single orbit characterization for $k = p^r \cdot (\ell + 1)$. In particular, for every w, q there exists a $k = k(w, q)$ such that the q -ary Reed-Muller family of weight degree w has a k -single orbit characterization.*

Indeed an immediate consequence of the two theorems above is that the sum of Reed-Muller and sparse properties over prime fields are locally testable.

Corollary 1. *For integers t, d and prime p , there exists a $k = k(t, d, p)$ such that for every n and every pair of properties $\mathcal{F}_1, \mathcal{F}_2 \in \{\mathbb{F}_{p^n} \rightarrow \mathbb{F}_p\}$, where \mathcal{F}_1 is the p -ary Reed-Muller property of weight degree d , and \mathcal{F}_2 is t -sparse, the property $\mathcal{F}_1 + \mathcal{F}_2$ has a k -single orbit characterization, and is hence k -locally testable.*

The corollary above describes the broadest known class of testable properties when n and q are prime. When n is not prime, the earlier-mentioned notion of lifting leads to other locally testable binary properties, and then intersection also leads to further richness.

Due to space restrictions, we give just a brief hint of the proof of our main theorem. We also describe some of the open questions and conjectures arising from our work. A full version of this work is available as [7].

2 The Structure of Affine-Invariant Properties

In what follows \mathbb{F}_q will denote the field of q elements of characteristic p , where $q = p^s$ for some integer s . Let $d = \sum_i d_i p^i$ be the base p representation of an integer d . The **weight** (or p -weight) of d is defined as $\text{wt}(d) = \sum_i d_i$. I.e. it is

the sum of coefficients in the p -ary representation of d . A non-negative integer $e = \sum_i e_i p^i$ is said to be in the p -shadow of d (or simply in the **shadow** of d), denoted $e \leq_p d$, if $e_i \leq d_i$ for all i . We denote $a \equiv_k b$ whenever a is equal to b modulo k . As we will be studying polynomials modulo identities of the form $x^q - x \equiv_p 0$ it will be convenient to define the following variant of the modular operation. Let a and k be integers. We define $a \bmod^* k$ as

$$a \bmod^* k = \begin{cases} 0 & a = 0 \\ b & \text{where } 1 \leq b \leq k \text{ is such that } b \equiv_k a \end{cases}$$

We also say that $a \equiv b \pmod{k}$ if $a \bmod^* k = b \bmod^* k$. Note that the only difference between \bmod and \bmod^* is that \bmod^* does not send nonzero multiples of k to zero but rather to k . It is now clear that $x^a \equiv_q x^{a \bmod^* q-1}$.

The class of properties that we consider are characterized by their algebraic properties. To describe such properties we need to introduce several notions from the works of [19,13,14,9,8].

We view functions $f : \mathbb{F}_{q^n} \rightarrow \mathbb{F}_q$ as functions from $\mathbb{F}_{q^n} \rightarrow \mathbb{F}_{q^n}$ whose image just happens to be contained in $\mathbb{F}_q \subseteq \mathbb{F}_{q^n}$. This allows us to view f as (the evaluation of) a univariate polynomial of degree $q^n - 1$.

Let $f(x) = \sum_{d=0}^{q^n-1} c_d x^d$. The *support* of f , denoted $\text{supp}(f)$, is the set $\text{supp}(f) = \{d \mid c_d \neq 0\}$.

The following definition captures an important feature of the structure of affine invariant families.

Definition 1 ($\text{Deg}(\mathcal{F})$). *Let $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ be a family of functions. The **degree set** of \mathcal{F} , denoted $\text{Deg}(\mathcal{F})$, is the set of degrees of monomials that appear in some polynomial in \mathcal{F} . Formally,*

$$\text{Deg}(\mathcal{F}) = \{d \mid \exists f \in \mathcal{F} \text{ such that } d \in \text{supp}(f)\}.$$

To better understand affine-invariance we need to describe some basic properties of the degree sets (the ones that are known to lead to local testability). We do so in the next two definitions.

Definition 2 ($\text{Shift}(d), \text{Shift}(D)$, **shift-closed**, **shift-representatives**, $\text{Fam}(D)$). *Let d be an integer in $\{0, \dots, q^n - 1\}$. The **shift** of d is defined as the set of degrees obtained when taking all q powers of x^d . Formally, $\text{Shift}_{q,n}(d) = \{q^i \cdot d \bmod^* q^n - 1 \mid \forall 0 \leq i \leq n\}$. Recall that $q^i \cdot d \bmod^* q^n - 1$ is the integer d' such that if $d \neq 0$ then $d' \equiv q^i d \pmod{(q^n - 1)}$ and $1 \leq d' \leq q^n - 1$, and if $d = 0$ then $d' = 0$. (In what follows, we will always be considering degrees in the support of functions from \mathbb{F}_{q^n} to \mathbb{F}_q , so that we drop the subscripts.)*

*We extend the notion to a set of degrees naturally. For a set $D \subseteq \{0, \dots, q^n - 1\}$, the shift of D is defined as $\text{Shift}(D) = \bigcup_{d \in D} \text{Shift}(d)$. A set D is said to be **shift-closed** if $\text{Shift}(D) = D$. For a shift-closed D , a set $S \subseteq D$ is said to be a set of **shift-representatives** of D if $\text{Shift}(S) = D$ and $\text{Shift}(d) \cap \text{Shift}(d') = \emptyset$ for $d, d' \in S$. (In other words S contains one element from each “shift” class*

in D ; by convention we assume each element of S is the smallest amongst its shifts. \square

Finally, for a shift-closed D , we define $\text{Fam}(D) = \{\text{Trace}(f) \mid f : \mathbb{F}_{q^n} \rightarrow \mathbb{F}_{q^n}, \text{supp}(f) \subseteq D\}$.

Another important ingredient that we will use is the *shadow* of a degree.

Definition 3 (Shadow, Shadow-closed set). For a non-negative integer d , the *shadow* of d is the set $\text{Shadow}(d) = \{e \mid e \leq_p d\}$. The shadow of a set S of non-negative integers is simply the union of the shadows of its elements, i.e., $\text{Shadow}(S) = \bigcup_{d \in S} \text{Shadow}(d)$. A set S of non-negative integers is **shadow-closed** if $\text{Shadow}(S) = S$.

For a general (linear) family \mathcal{F} , the degree set of \mathcal{F} does not give much useful information about \mathcal{F} . However, for affine invariant families, this set completely describes the family. Furthermore, sets of degrees that are closed under shifts and under shadows completely characterize affine-invariant properties.

Our next lemma repeats in different forms in the literature [19,13,14,9]. Specifically, it is Lemma 3.5 in [8].

Lemma 1 (Closed degree sets specify affine-invariant properties). Let \mathcal{F} be a linear and affine-invariant family. Then $\text{Deg}(\mathcal{F})$ is shadow-closed and shift-closed, and $\mathcal{F} = \text{Fam}(\text{Deg}(\mathcal{F}))$. Conversely, if D is shadow-closed and shift-closed then D is the degree set of some affine invariant family. More specifically, $\text{Fam}(D)$ is affine-invariant and $D = \text{Deg}(\text{Fam}(D))$.

3 Sums of Affine-Invariant Properties

In this section we prove Theorem 1. The main idea behind the proof is that instead of looking at the sets of degrees of a locally characterizable family \mathcal{F} , we look at the *border* set of degrees. These are the integers that do not themselves belong to $\text{Deg}(\mathcal{F})$ but every integer in their shadow is in $\text{Deg}(\mathcal{F})$.

Definition 4 (Border of a family). Let $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ be a family of functions. The **border** of \mathcal{F} is the set of degrees given by

$$\text{Border}(\mathcal{F}) = \{d \notin \text{Deg}(\mathcal{F}) \mid \forall e <_p d, e \in \text{Deg}(\mathcal{F})\}.$$

We start by noticing that a *k-single orbit characterization* can be specified by a pair $(\bar{\alpha}; \{\bar{\lambda}_i\}_{i=1}^t)$, where $\bar{\alpha} = (\alpha_1, \dots, \alpha_k) \in \mathbb{F}_{q^n}^k$ and $\bar{\lambda}_i = (\lambda_{i,1}, \dots, \lambda_{i,k}) \in \mathbb{F}_q^k$, and $f \in \mathcal{F}$ if and only if it satisfies $\sum_{j=1}^k \lambda_{i,j} f(\pi(\alpha_j)) = 0$ for every $i \in \{1, \dots, t\}$ and every affine map $\pi : \mathbb{F}_{q^n} \rightarrow \mathbb{F}_{q^n}$. Note further that we can assume $t \leq k$ in the specification above. The following lemma gives several equivalent definitions to being a *k-single orbit characterizable family*. The lemma can be seen as an extension of Lemma 3.6 in [8].

² As $d' \in \text{Shift}(d)$ if and only if $d \in \text{Shift}(d')$, such S always exists.

Lemma 2. [Equivalent definitions of k -single orbit characterizable family]

Let $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ be a linear affine-invariant family. The following are equivalent:

1. $(\bar{\alpha}; \{\bar{\lambda}_i\}_{i=1}^t)$ is a k -single orbit characterization of \mathcal{F} , where $\bar{\alpha} = (\alpha_1, \dots, \alpha_k) \in \mathbb{F}_{q^n}^k$ and $\bar{\lambda}_i = (\lambda_{i,1}, \dots, \lambda_{i,k}) \in \mathbb{F}_q^k$.
2. For all $d, d \in \text{Deg}(\mathcal{F}) \Leftrightarrow \forall i \sum_{j=1}^k \lambda_{i,j}(\alpha_j x + y)^d \equiv 0$ (as a formal polynomial in x and y).
3. For all $d, d \in \text{Deg}(\mathcal{F}) \Leftrightarrow \forall e \leq_p d, \forall i \sum_{j=1}^k \lambda_{i,j} \alpha_j^e = 0$.
4. For all $d \in \text{Deg}(\mathcal{F}) \cup \text{Border}(\mathcal{F}), d \in \text{Deg}(\mathcal{F}) \Leftrightarrow \forall i \sum_{j=1}^k \lambda_{i,j} \alpha_j^d = 0$.

3.1 Proof of Main Theorem

Due to space limitations we give only an idea of the proof of Theorem [11](#)

Let \mathcal{F}_1 and \mathcal{F}_2 be k -single orbit characterized properties (we assume for simplicity that $k_1 = k_2 = k$, and make several other simplifying assumptions here). Suppose the k -single orbit characterization of \mathcal{F}_1 is simply $(\bar{\alpha}, \bar{1})$ (so \mathcal{F}_1 satisfies $\sum_{i=1}^k f(\alpha_i) = 0$). Similarly, let the k -single orbit characterization of \mathcal{F}_2 be $(\bar{\beta}, \bar{1})$.

A candidate k^2 -single orbit characterization of $\mathcal{F}_1 + \mathcal{F}_2$ would be the the “outer product” of the two given constraints, namely the k^2 local constraint given by $((\alpha_i \beta_j)_{i,j}; \bar{1})$.

To analyze this potential constraint, we look at the degree set based descriptions of single-orbit characterizations. First we use the (easily verifiable fact) that $\text{Deg}(\mathcal{F}_1 + \mathcal{F}_2) = \text{Deg}(\mathcal{F}_1) \cup \text{Deg}(\mathcal{F}_2)$. Next we see that for every $d \in \text{Deg}(\mathcal{F}_1) \cup \text{Deg}(\mathcal{F}_2), \sum_{i=1}^k \sum_{j=1}^k \alpha_i^d \beta_j^d = (\sum_{i=1}^k \alpha_i^d) \cdot (\sum_{j=1}^k \beta_j^d) = 0$, so $((\alpha_i \beta_j)_{i,j}; \bar{1})$ is a valid constraint on $\mathcal{F}_1 + \mathcal{F}_2$.

Unfortunately, it is not clear that for every $d \in \text{Border}(\mathcal{F}_1 + \mathcal{F}_2)$ the sum $\sum_{i=1}^k \sum_{j=1}^k \alpha_i^d \beta_j^d$ does not equal 0, which is necessary (by Part [4](#) of Lemma [2](#)).

To remedy this, we take a random constraint in the orbit of $(\bar{\alpha}; \bar{1})$ and a random constraint in the orbit of $(\bar{\beta}; \bar{1})$ and take their “outer product”. Specifically we pick random non-zero $a_1, a_2 \in \mathbb{F}_{q^n}$ and random $b_1, b_2 \in \mathbb{F}_{q^n}$ and consider the potential constraint $(\bar{\gamma}; \bar{1})$ where $\bar{\gamma} = (\gamma_{i,j})_{i,j}$ is given by $\gamma_{i,j} = (a_1 \alpha_i + b_1)(a_2 \beta_j + b_2)$. It is again easy to verify that $\sum_{i,j} \gamma_{i,j}^d = 0$ for every $d \in \text{Deg}(\mathcal{F}_1) \cup \text{Deg}(\mathcal{F}_2)$.

We then note that for any fixed $d \notin \text{Deg}(\mathcal{F}_1) \cup \text{Deg}(\mathcal{F}_2)$ the formal sum $\sum_{i,j} ((x_1 \alpha_i + y_1)(x_2 \beta_j + y_2))^d \neq 0$ (as a polynomial in x_1, x_2, y_1, y_2 — this uses Part [2](#) of Lemma [2](#)). Thus when we pick random assignments $x_1 = a_1, x_2 = a_2$ etc., we find that $\sum_{i,j} \gamma_{i,j}^d \neq 0$ with probability at least $1 - O(d/q^n)$, and so this random choice does eliminate any particular “bad” d .

To conclude the argument we need to make sure that every $d \in \text{Border}(\mathcal{F}_1 + \mathcal{F}_2)$ is eliminated (i.e., $\sum_{i,j} \gamma_{i,j}^d \neq 0$). To do so, we use the union bound, with two crucial ingredients: First we use the main theorem from [9](#) to conclude that all $d \in \text{Border}(\mathcal{F}_1 + \mathcal{F}_2)$ have p -weight at most k and there are only $(q + n)^{O(k)}$ such d 's. Next we use the fact that we need to consider only one d from every “shift” class, to take the smallest one. This allows us to work with $d \leq q^{n(1-1/k)}$.

Combining these ingredients, we can take the union bound over all possible bad events and conclude that a random choice a_1, a_2, b_1, b_2 eliminates every $d \in \text{Border}(\mathcal{F}_1 + \mathcal{F}_2)$ with positive probability.

4 Consequences, Questions and Conjectures

Our work further highlights the role played by single-orbit characterizations in the testing of affine-invariant properties. This feature is common (e.g. Reed-Muller property is single-orbit over the smaller group) and also useful (sums of single-orbit characterized properties also have this feature). In this section we describe some of the questions surrounding this concept that emerge from this (and related) research.

At the moment almost all known locally-testable affine-invariant properties are known to be single-orbit characterized. The only exception is the case of sparse properties where the range is not a prime field. This leads to the following question, which we hope can be resolved affirmatively (soon).

Question 1. For every q and t , does there exists a constant $k = k(q, t)$ such that every t -sparse property $\mathcal{F} \subseteq \{\mathbb{F}_q^n \rightarrow \mathbb{F}_q\}$ is k -single orbit characterized?

Assuming an affirmative answer to the questions above, we get a “concise” description of all known testable properties.

4.1 Known Locally Testable Properties

As mentioned earlier, the known “basic” single-orbit characterized affine-invariant families are the Reed-Muller families and sparse families. Three “operations” are also now known that preserve “single-orbit characterizations” and hence local testability of these basic families: (1) Sums of two families, (2) Intersections of two families, and (3) Lift of a single family. Below we define this lifting operator.

Definition 5 (Lifted code [8]). Let $\mathbb{K} \supseteq \mathbb{L} \supseteq \mathbb{F}_q$ be finite fields with $q = p^s$. For $D \subseteq \{0, \dots, |\mathbb{L}| - 1\}$ we define the lift of D from \mathbb{L} to \mathbb{K} to be the set of integers $\text{lift}_{\mathbb{L}/\mathbb{K}}(D) = \{d' \in \{0, \dots, |\mathbb{K}| - 1\} \mid (\text{shadow}_p(d') \pmod{|\mathbb{L}| - 1}) \in D\}$.

For an affine-invariant family $\mathcal{F} \subseteq \{\mathbb{L} \rightarrow \mathbb{F}_q\}$ with degree set $D = \text{Deg}(\mathcal{F})$, let $\text{lift}_{\mathbb{L}/\mathbb{K}}(\mathcal{F})$ be the affine-invariant family with degree set $\text{lift}_{\mathbb{L}/\mathbb{K}}(D)$, i.e., $\text{lift}_{\mathbb{L}/\mathbb{K}}(\mathcal{F}) = \{f : \mathbb{K} \rightarrow \mathbb{F}_q \mid \text{supp}(f) \subseteq \text{lift}_{\mathbb{L}/\mathbb{K}}(D)\} = \text{Fam}(\text{lift}_{\mathbb{L}/\mathbb{K}}(D))$.

The following proposition follows easily from the definitions

Proposition 1 ([8]). Lifts of single orbit characterized families are also single-orbit characterized. Specifically, if $\mathbb{F}_q \subseteq \mathbb{L} \subseteq \mathbb{K}$ and $(\bar{\alpha}, \{\bar{\lambda}_i\}_{i=1}^t)$ is a k -single orbit characterization of $\mathcal{F} \subseteq \{\mathbb{L} \rightarrow \mathbb{F}_q\}$ then $(\bar{\alpha}, \{\bar{\lambda}_i\}_{i=1}^t)$ is also k -single orbit characterization of $\text{lift}_{\mathbb{L}/\mathbb{K}}(\mathcal{F})$.

Given the operations above, it is easy to see that one can compose a finite number of basic single-orbit characterized families using a “formula” whose operations are sum, intersection and lifts. We define this concept below.

Definition 6 (Formula, size). *A formula Φ of size s , degree d , sparsity t producing a family $\mathcal{F} \subseteq \{\mathbb{K} \rightarrow \mathbb{F}_q\}$, denoted $(s, d, t, \mathbb{K}, \mathbb{F})$ -formula, is given by the following inductive definition:*

1. *A formula Φ of size 1, is given by $\mathcal{F} \subseteq \{\mathbb{K} \rightarrow \mathbb{F}_q\}$ where \mathcal{F} is either a Reed-Muller family of order d , or a t -sparse family.*
2. *A formula of size s is obtained by one of the following operations:*
 - (a) *Picking \mathbb{L} such that $\mathbb{F}_q \subseteq \mathbb{L} \subseteq \mathbb{K}$ and letting $\Phi = \text{lift}_{\mathbb{L}/\mathbb{K}}(\Phi_1)$ where Φ_1 is a $(s - 1, t, d, \mathbb{L}, \mathbb{F})$ formula.*
 - (b) *Picking s_1, s_2 such that $s_1 + s_2 + 1 = s$ and letting $\Phi = \Phi_1 \cap \Phi_2$ where Φ_i is an $(s_i, t, d, \mathbb{K}, \mathbb{F})$ formula.*
 - (c) *Picking s_1, s_2 such that $s_1 + s_2 + 1 = s$ and letting $\Phi = \Phi_1 + \Phi_2$ where Φ_i is an $(s_i, t, d, \mathbb{K}, \mathbb{F})$ formula.*

The following theorem summarizes the state of knowledge of single-orbit characterized families.

Theorem 3. *For every s, t, d, q there exists a $k = k(s, t, d, q)$ such that for every n , every $(s, t, d, \mathbb{F}_{q^n}, \mathbb{F}_q)$ -formula produces a k -single orbit characterized family, for prime q .*

Note that the caveat that q is prime can be dropped if we have an affirmative answer to Question [11](#).

4.2 Conjectures and Questions

We start with the most obvious question.

Question 2. Is the following statement true? For every k, q there exist s, t, d such that for every n , if $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ is a k -locally testable affine-invariant family then \mathcal{F} is given by an $(s, t, d, \mathbb{F}_{q^n}, \mathbb{F}_q)$ -formula.

At the moment our understanding of affine-invariance with respect to its local testability is so far that it is too optimistic to conjecture an affirmative answer to this question. All we can say is that an affirmative answer is not yet ruled out.

The nature of the question seems to become much simpler if we disallow lifts, by insisting that n is prime (then we get no fields \mathbb{L} strictly between \mathbb{F}_q and \mathbb{F}_{q^n}). In this setting, intersections become uninteresting and lead to a much tamer question.

Question 3. Is the following statement true? For every k, q there exist t, d such that for every prime n , if $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ is a k -locally testable affine-invariant family then $\mathcal{F} = \mathcal{F}_1 + \mathcal{F}_2$ where $\mathcal{F}_1 = \text{RM}_q(d', n)$ and \mathcal{F}_2 is t' -sparse, for some $d' \leq d$ and $t' \leq t$.

This question remains quite challenging even when we restrict to the case where $q = 2$ (where our state of understanding does seem somewhat better), and even when we restrict our families to be contained in $\text{RM}_2(2, n)$.

Conjecture 1. For every k there exists a t such that the following holds for every prime n : If $\mathcal{F} \subsetneq \text{RM}_2(2, n)$ is k -locally testable then \mathcal{F} is t -sparse.

Attempting to prove the conjecture above leads to some interesting questions about the rank of certain Vandermonde like matrices that seem interesting in their own right. We state the conjecture below. We don't prove the connection to the conjecture above, but claim that an affirmative answer to the following implies an affirmative answer to the above.

Conjecture 2. For every k , there exists a t such that for every prime n and every sequence $\alpha_1, \dots, \alpha_k \in \mathbb{F}_{2^n}$ of elements that are \mathbb{F}_2 -linearly independent, and every sequence of t distinct elements $e_1, \dots, e_t \in \{0, \dots, n-1\}$, the $k \times t$ matrix $M = [M_{ij}]_{ij}$ with $M_{ij} = \alpha_i^{2^{e_j}}$ has rank exactly k .

Finally a couple of questions which relate to the structure of locally-testable codes (an affirmative answer to both is implied by an affirmative answer to Question 2).

Question 4. For every k, q does there exist a \tilde{k} such that for every n , if $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ is k -locally testable, then \mathcal{F} has a \tilde{k} -single orbit characterization?

Question 5. For every k, q does there exist a \tilde{k} such that for every n , if $\mathcal{F}_1, \mathcal{F}_2 \subseteq \{\mathbb{F}_{q^n} \rightarrow \mathbb{F}_q\}$ are k -locally testable, then $\mathcal{F}_1 + \mathcal{F}_2$ is \tilde{k} -locally testable?

Acknowledgments. We would like to thank Shripad Garge, Neeraj Kayal and Dipendra Prasad for valuable pointers.

References

1. Alon, N., Fischer, E., Newman, I., Shapira, A.: A combinatorial characterization of the testable graph properties: it's all about regularity. In: Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, pp. 251–260. ACM, New York (2006)
2. Alon, N., Kaufman, T., Krivelevich, M., Litsyn, S., Ron, D.: Testing Reed-Muller codes. *IEEE Transactions on Information Theory* 51(11), 4032–4039 (2005)
3. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and the hardness of approximation problems. *Journal of the ACM* 45(3), 501–555 (1998)
4. Arora, S., Safra, S.: Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM* 45(1), 70–122 (1998)
5. Barak, B., Dvir, Z., Wigderson, A., Yehudayoff, A.: Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In: Electronic Colloquium on Computational Complexity (ECCC), vol. 17, p. 149 (2010); To appear in STOC 2011

6. Barkol, O., Ishai, Y., Weinreb, E.: On locally decodable codes, self-correctable codes, and t -private PIR. *Algorithmica* 58, 831–859 (2010)
7. Ben-Sasson, E., Grigorescu, E., Maatouk, G., Shpilka, A., Sudan, M.: On sums of locally testable affine invariant properties. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 18, p. 79 (2011)
8. Ben-Sasson, E., Maatouk, G., Shpilka, A., Sudan, M.: Symmetric LDPC codes are not necessarily locally testable. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 17, p. 199 (2010); To appear in *CCC 2011*
9. Ben-Sasson, E., Sudan, M.: Limits on the rate of locally testable affine-invariant codes. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 17, p. 108 (2010)
10. Borgs, C., Chayes, J.T., Lovász, L., Sós, V.T., Szegedy, B., Vesztegombi, K.: Graph limits and parameter testing. In: *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, Seattle, WA, USA, May 21–23, pp. 261–270. ACM, New York (2006)
11. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private information retrieval. *Journal of the ACM* 45(6), 965–981 (1998)
12. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *Journal of the ACM* 45(4), 653–750 (1998)
13. Grigorescu, E., Kaufman, T., Sudan, M.: 2-transitivity is insufficient for local testability. In: *Proceedings of the 23rd Annual IEEE Conference on Computational Complexity, CCC 2008*, College Park, Maryland, USA, June 23–26, pp. 259–267. IEEE Computer Society, Los Alamitos (2008)
14. Grigorescu, E., Kaufman, T., Sudan, M.: Succinct Representation of Codes with Applications to Testing. In: Dinur, I., Jansen, K., Naor, J., Rolim, J.D.P. (eds.) *APPROX-RANDOM 2009*. LNCS, vol. 5687, pp. 534–547. Springer, Heidelberg (2009)
15. Jutla, C.S., Patthak, A.C., Rudra, A., Zuckerman, D.: Testing low-degree polynomials over prime fields. *Random Struct. Algorithms* 35(2), 163–193 (2009)
16. Katz, J., Trevisan, L.: On the efficiency of local decoding procedures for error-correcting codes. In: *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, STOC 2000*, pp. 80–86. ACM, New York (2000)
17. Kaufman, T., Ron, D.: Testing polynomials over general fields. *SIAM J. on Computing* 36(3), 779–802 (2006)
18. Kaufman, T., Lovett, S.: Testing of exponentially large codes, by a new extension to Weil bound for character sums. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 17, p. 65 (2010)
19. Kaufman, T., Sudan, M.: Algebraic property testing: the role of invariance. In: Dwork, C. (ed.) *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, Victoria, British Columbia, Canada, May 17–20, pp. 403–412. ACM, New York (2008)
20. Rubinfeld, R., Sudan, M.: Robust characterizations of polynomials with applications to program testing. *SIAM J. on Computing* 25(2), 252–271 (1996)
21. Sudan, M.: Invariance in property testing. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. 17, p. 51 (2010)

Limits on the Rate of Locally Testable Affine-Invariant Codes

Eli Ben-Sasson* and Madhu Sudan**

Abstract. Despite its many applications, to program checking, probabilistically checkable proofs, locally testable and locally decodable codes, and cryptography, “algebraic property testing” is not well-understood. A significant obstacle to a better understanding, was a lack of a concrete definition that abstracted known testable algebraic properties and reflected their testability. This obstacle was removed by [Kaufman and Sudan, STOC 2008] who considered (linear) “affine-invariant properties”, i.e., properties that are closed under summation, and under affine transformations of the domain. Kaufman and Sudan showed that these two features (linearity of the property and its affine-invariance) play a central role in the testability of many known algebraic properties. However their work does not give a complete characterization of the testability of affine-invariant properties, and several technical obstacles need to be overcome to obtain such a characterization. Indeed, their work left open the tantalizing possibility that locally testable codes of rate dramatically better than that of the family of Reed-Muller codes (the most popular form of locally testable codes, which also happen to be affine-invariant) could be found by systematically exploring the space of affine-invariant properties.

In this work we rule out this possibility and show that general (linear) affine-invariant properties are contained in Reed-Muller codes that are testable with a slightly larger query complexity. A central impediment to proving such results was the limited understanding of the structural restrictions on affine-invariant properties imposed by the existence of local tests. We manage to overcome this limitation and present a clean restriction satisfied by affine-invariant properties that exhibit local tests. We do so by relating the problem to that of studying the set of solutions of a certain nice class of (uniform, homogenous, diagonal) systems of multivariate polynomial equations. Our main technical result completely characterizes (combinatorially) the set of zeroes, or algebraic set, of such systems.

Keywords: Property testing, Symmetries, Direct sums, Error-correcting codes.

* Department of Computer Science, Technion — Israel Institute of Technology, Haifa, Israel, eli@cs.technion.ac.il. Research funded partially by the European Community’s Seventh Framework Programme (FP7/2007-2013) Grant 240258 and the US-Israel Binational Science Foundation Grant 2006104.

** Microsoft Research New England, Cambridge, Massachusetts, USA, madhu@mit.edu.

1 Introduction

In this work we consider an interesting subclass of “locally correctable” and “locally testable codes”, namely “affine-invariant” codes, and prove upper bounds (limitations) on their rate. In the process we also introduce techniques of relevance to “algebraic property testing” and present a characterization of the set of solutions of a certain natural system of multivariate polynomials that lies at the core of our study.

1.1 Locally Correctable and Locally Testable Codes, Affine-Invariance, and Main Result

Locally correctable codes (LCCs) are error-correcting codes with the property that every entry of a corrupted codeword can be corrected, with high probability, by examining a small (random) subset of other entries of the corrupted codeword. Locally correctable codes are a special class of locally decodable codes (LDCs) studied in the work of [5,24,27] and formally defined by [21]. These codes are tightly connected to the construction of private information retrieval schemes [13] and we refer the reader to [30] for more information. One of the major open problems regarding LDCs is that of determining the minimal length n of a binary LDC by which messages of k bits can be encoded, and the current lower and upper bounds on n display an exponential gap. Namely, [28] showed n must be at least (roughly) $k^{1+\frac{2}{7}}$ whereas the best upper bounds of [29] show n is at most (roughly) $\exp(k^{1/\log \log k})$ (cf. [15]).

Locally testable codes are error-correcting codes for whom membership can be tested extremely efficiently, probabilistically. Specifically, a linear code $\mathcal{C} \subseteq \Sigma^N$ is k -locally testable if there exists an algorithm T that accesses a word $w \in \Sigma^N$ as an oracle, queries the value of w on k coordinates, and accepts with probability one if $w \in \mathcal{C}$ and rejects with constant probability if w is “far” from all codewords of \mathcal{C} . (“Far” here refers to the relativized Hamming distance between words.)

Locally testable codes have implicitly been a subject of active study ever since the work of [11] that showed that (effectively) the Hadamard code is 3-locally testable. They play a major role in the construction of PCPs [43] from the early days of this theorem and continuing through the recent work of [14]. Their systematic investigation was started in [17] and yet most basic questions about their limits remain unanswered (e.g., is there an asymptotically good family of locally testable codes?).

A particularly interesting class of locally testable and locally correctable codes are the affine-invariant ones. Here the code is a linear code over some finite field \mathbb{F} and the coordinates of the code are themselves vector spaces over some finite extension field \mathbb{K} of \mathbb{F} . Thus words in such codes can be viewed as functions from \mathbb{K}^m to \mathbb{F} and the code is a subfamily of such functions. The code is said to be affine invariant if it is invariant under affine-transformations of the domain. Specifically if $A : \mathbb{K}^m \rightarrow \mathbb{K}^m$ is an affine transformation and $f : \mathbb{K}^m \rightarrow \mathbb{F}$ is a function in \mathcal{C} , then so is $f \circ A$ where $f \circ A(x) = f(A(x))$.

Affine-invariant codes form a natural class of algebraic codes and have been studied by the error-correcting-codes community since the late 1960's (cf. [20] and references therein). In the context of locally testable and locally correctable codes, affine-invariance facilitates natural local correction/testing procedures under minimal conditions. Specifically, it is well known that for a linear code to be testable it must have a low weight codeword in its dual, or equivalently a local "constraint" (see, for instance, [7]). In the notation used above for affine invariant codes, a k -local "constraint" is a collection of points $\alpha_1, \dots, \alpha_k \in \mathbb{K}^m$ and values $\lambda_1, \dots, \lambda_k \in \mathbb{F}$ such that for every function $f \in \mathcal{C}$, it is the case that $\sum_{i=1}^k \lambda_i f(\alpha_i) = 0$. For affine-invariant codes the presence of one local constraint immediately implies many local constraints by affine "rotations": For every affine map A , the set of points $A(\alpha_1), \dots, A(\alpha_k)$ also define a constraint on \mathcal{C} . This abundance of constraints leads easily to a local-correction procedure and also raises the hope that affine-invariant codes may be locally testable, and indeed [23] show that if the code is characterized by the set of constraints derived from the affine rotations of a single constraint, then it is also locally testable. (The more optimistic hope, that all affine-invariant locally-characterized codes are also locally testable, has been recently refuted in [8] as a result of this work.)

We point out that it is the *abundance* of local constraints, not their mere existence, that seems to be essential for obtaining locally testable codes. In extreme cases where there is no abundance of local constraints, such as for low-density-parity-check (LDPC) codes based on random expanders, or for codes that have the very minimal number of local constraints needed to characterize them, there cannot be any hope for local testability [7,6]. But, all things considered, abundance of local constraints should reduce the rate of the code, unless the constraints are carefully chosen in an algebraically consistent way. The class of affine invariant codes offered a promising approach to balance the need for abundance of local constraints with maintaining high rate.

This leads to the question: Which affine invariant codes have local constraints (and characterizations), and in particular how local can the constraints be, given other parameters of the code, most notably, its rate? One, somewhat optimistic hope, was that affine-invariance might lead to simpler constructions of locally testable codes matching the best known parameters (the current constructions are immensely complicated [9,14]), or even improve on them, since the scope is wider than just the class of Reed-Muller codes. This question however, resisted attacks till now, since the question of determining when a low-weight constraint can exist in an affine-invariant code leads to questions about the zeroes of certain systems of multivariate polynomial equations and these are challenging to analyze.

Here we take some first steps in this direction, though unfortunately to give a negative answer to the optimistic hope above. Specifically, we give a full analysis of a certain class of polynomial equations that arise in this setting to get a rate upper bound on affine invariant codes. For simplicity of exposition we describe our result for the case of prime fields $\mathbb{F} = \mathbb{F}_p$. The statement for the case fields of size $p^r, r > 1$ is somewhat more technical but the rate bounds we get for

this case are similar to that of prime fields (cf. Theorem 2 and Corollary 1). Our main theorem (Theorem 2) shows that if \mathbb{K} is an extension field of \mathbb{F} and \mathcal{C} is a k -locally testable/correctable code, then \mathcal{C} is contained in a p^{k-1} -locally testable Reed-Muller code. If k and p are constants (which is the desired setting of parameters) then it says that going to general affine-invariance only buys (at best) a constant difference in the locality, when compared to the Reed-Muller codes. Since Reed-Muller codes with constant locality over constant field sizes are known to have exponentially low-rate, this rules out the hope described in the previous paragraph, by a long margin.

Notice there is an exponential gap between the query complexity of affine-invariant codes with a k -local constraint and the query complexity of the Reed-Muller code which we show contains them, which is p^{k-1} . Getting a full characterization of affine-invariant codes with a k -local constraint, even over specific fields (like \mathbb{F}_{2^n} for prime n , a field which contains no subfields other than \mathbb{F}_2) seems to us like an interesting question for future research.

1.2 Algebraic Property Testing

Property testing considers the task of testing if a function f from a large domain D to a small range R satisfies some given property, where the property itself is given by the set of functions $\mathcal{F} \subseteq \{g : D \rightarrow R\}$ that satisfy the property. Again the interest here is in “quick and dirty” tests, i.e., probabilistic tests that query the given function f on few inputs, and accept if $f \in \mathcal{F}$ and reject with constant probability if f is far from \mathcal{F} . (Note that a locally testable code is just property testing where we view the set of functions \mathcal{F} as an error-correcting code.)

Property testing also emerged in the work of [11], was formally defined by [25], and was systematically explored (in particular in non-algebraic contexts) by [16]. Subsequently the study of combinatorial property testing, and in particular, graph property testing has developed into a rich study and by now we have almost complete understanding (at least in the dense-graph model) of which graph properties are locally testable [11,12].

In contrast algebraic properties have not been understood as well, despite the overwhelming applications in complexity, and indeed till recently even an understanding of what makes a property algebraic was missing. The concept of affine-invariance was introduced by [23] to propose such a notion, and when the domain is a vector space over a small field \mathbb{K} (of constant size) they manage to characterize locally testable properties completely. Such codes are constant-locally testable if and only if they admit a constant local constraint, and the size of the constraint can be related loosely to the highest degree of polynomials in the family.

This naturally leads to the question: What about affine invariant codes over large fields. (This family of properties includes, for instance, all sets of low-degree polynomials, i.e., the families of Hadamard, Reed-Solomon and Reed-Muller codes.) In particular for the extreme, and essentially most general case when $m = 1$ and the functions of interest map \mathbb{K} to a prime subfield \mathbb{F}_p , there was no interesting relationships known between the degrees of the functions in

the family and the locality of the test. And such understanding is essential to get a characterization of affine-invariant locally testable codes that would be analogous to the characterizations of graph properties of [11,12].

Our work takes a step in this direction by giving non-trivial lower bounds on the locality of tests for affine-invariant properties in the general case. Below we describe the main technical question resolved in this paper (which has a self-contained description).

2 Definitions and Main Results

2.1 Preliminaries — Locally Testable, and Reed-Muller Codes

Notation We use $[n]$ to denote the set $\{1, \dots, n\}$. Throughout we let $\mathbb{F}, \mathbb{K}, \mathbb{L}$ denote fields. The q element finite field is denoted by \mathbb{F}_q . An $[N, K, D]_{\mathbb{F}}$ -(linear) code is a K -dimensional subspace $\mathcal{C} \subseteq \mathbb{F}^N$ of Hamming distance D . Elements of \mathcal{C} are referred to as codewords (of \mathcal{C}). Two vectors $u, w \in \mathbb{F}^N$ are said to be δ -close if they are within Hamming distance $\leq \delta N$ of each other, otherwise they are said to be δ -far. A vector u is said to be δ -close to \mathcal{C} if it is δ -close to some codeword $w \in \mathcal{C}$, otherwise we say w is δ -far from \mathcal{C} . We define $\langle u, w \rangle \triangleq \sum_{i=1}^N u_i w_i$. Let $\mathcal{C}^\perp = \{u \in \mathbb{F}^N \mid \langle u, w \rangle = 0 \text{ for all } w \in \mathcal{C}\}$ denote the space that is dual to \mathcal{C} (it is also known as the *dual code* of \mathcal{C}).

We recall the standard definitions of a tester for a linear code and a linear locally testable code. All codes considered in this paper are linear so from here on we drop further reference to this linearity (of testers, codes, etc.).

Definition 1 (Tester). *Suppose \mathcal{C} is a $[N, K, D]_{\mathbb{F}}$ -code. A k -query tester for \mathcal{C} is a probabilistic oracle algorithm T that makes at most k oracle queries to a word $w \in \mathbb{F}^N$ and outputs an accept/reject verdict. The tester is said to have completeness c and ϵ -soundness s if it accepts every codeword of \mathcal{C} with probability at least c and accepts words that are ϵ -far from \mathcal{C} with probability at most s .*

Definition 2 (Locally Testable Code (LTC)). *An $[N, K, D]_{\mathbb{F}}$ -code \mathcal{C} is said to be a (k, ϵ, ρ) -Locally Testable Code (LTC) if there exists a k -query tester that has completeness c and ϵ -soundness $c - \rho$.*

We are typically interested in infinite family of codes. If an infinite family of codes is a (k, ϵ, ρ) -LTC for absolute constants k and $\epsilon, \rho > 0$, then we simply refer to this (family of) code(s) as an LTC. For linear LTCs the nature of tests can be simplified significantly, due to a result of [7], to get them to a canonical form, which has perfect completeness ($c = 1$), and is *non-adaptive* (while the soundness parameter ρ changes by a constant factor). This leads to the following definition.

Definition 3 (Canonical tester). *A canonical k -query test for \mathcal{C} is given by an element $u \in \mathcal{C}^\perp$ that has support size at most k , i.e., $|\{i \mid u_i \neq 0\}| \leq k$, where the test accepts $w \in \mathbb{F}^n$ if and only if $\langle u, w \rangle = 0$. A k -query canonical tester T for \mathcal{C} is defined by a distribution μ over canonical k -query tests. Invoking the tester T on a word $w \in \mathbb{F}^n$ is done by sampling a test u according to the distribution μ and outputting **accept** if the canonical test given by u accepts.*

The following proposition of [7] — stated as Theorem 3.3 there — shows that tests may always be assumed to be canonical (up to a constant factor change in soundness).

Proposition 1. *For every $\epsilon, \rho > 0$ and positive integer k , there exist $\rho' > 0$ such that every (k, ϵ, ρ) -LTC has a canonical k -query tester with perfect completeness and ϵ -soundness $1 - \rho'$.*

Our main theorem compares the performance of affine-invariant locally testable codes to that of Reed-Muller codes, which we define next.

Definition 4 (Reed-Muller codes). *For \mathbb{F} a finite field of size q and m, k integers, the m -variate Reed-Muller code of degree k over \mathbb{F} , denoted $\text{RM}[q, m, k]$ is the $[N = q^m, K = \binom{m+k}{k}, D = q^{m-k}]_{\mathbb{F}}$ -code whose codewords are all evaluations of m -variate polynomials over \mathbb{F} of degree at most k .*

These codes have also been studied for the testability properties (see, e.g., [25], [2], [26], [22], [19], and [10]) and the case most relevant to us is that of constant q and k and arbitrarily large m . For this choice of parameters the codes are known to be $(q^{O(k)}, \epsilon, \rho)$ -locally testable for some constants $\epsilon, \rho > 0$ that may depend on q and k [2,22].

2.2 Affine Invariant Codes

The main concept of interest to us is that of affine-invariance. We borrow some of the main definitions related to this concept from [23].

From here on we associate a code with a family of functions. Let p be a prime, $\mathbb{F} = \mathbb{F}_q$ for $q = p^r$ be a finite field and let $\mathbb{K} = \mathbb{F}_Q$ for $Q = q^n$ be an extension of \mathbb{F} . For integer m we can consider $[N = Q^m, k, d]_{\mathbb{F}}$ -codes whose entries are indexed by elements of \mathbb{K}^m . In other words, from here on a code will be identified with an \mathbb{F} -linear subspace of $\{\mathbb{K}^m \rightarrow \mathbb{F}\}$, the space of all functions from \mathbb{K}^m to \mathbb{F} .

Definition 5 (Affine invariant codes). *Let \mathbb{K} be a finite degree extension of \mathbb{F} . A code $\mathcal{C} \subseteq \{\mathbb{K}^m \rightarrow \mathbb{F}\}$ is said to be affine invariant if it is invariant under the action of the affine monoid¹ over \mathbb{K}^m . In other words, for every $f \in \mathcal{C}$ and every affine transformation $A : \mathbb{K}^m \rightarrow \mathbb{K}^m$, the function $f \circ A$ defined by $(f \circ A)(x) = f(A(x))$ belongs to \mathcal{C} as well.*

The work of [7] shows that in order for a linear property to be testable, it must have some “local constraints” (low-weight words in its dual). For affine invariant codes, [23] show that when \mathbb{K} is small, then the existence of such constraints is also a sufficient condition. (Our main result will show that the existence of such constraints imposes a bound on the rate of a code, over any field \mathbb{K} , not just over fields of constant size.) We recall the following definition from [23].

¹ The set of all affine maps from \mathbb{K}^m to itself forms a monoid under composition. If one restricted this set to full rank maps, then one gets a group.

Definition 6 (*k*-local constraint). A *k*-local constraint is given by *k* distinct points in \mathbb{K}^m $\alpha = (\alpha_1, \dots, \alpha_k) \in (\mathbb{K}^m)^k$. We say that a code $\mathcal{C} \subseteq \{\mathbb{K}^m \rightarrow \mathbb{F}\}$ satisfies (or, has) a *k*-local constraint α if there exists nonzero $\Lambda = (\lambda_1, \dots, \lambda_k) \in \mathbb{F}^k$ such that $\sum_{i=1}^k \lambda_i f(\alpha_i) = 0$ for every $f \in \mathcal{C}$.

The following statement is the main result of [23] regarding the local testability of affine invariant codes, and is stated as Theorem 2.10 there.

Theorem 1 (Affine invariant codes satisfying a *k*-local constraint over a small field are locally testable). For fields $\mathbb{F} \subseteq \mathbb{K}$ with $|\mathbb{F}| = q$ and $|\mathbb{K}| = Q$, let $\mathcal{F} \subseteq \{\mathbb{K}^m \rightarrow \mathbb{F}\}$ be an affine-invariant code satisfying a *k*-local constraint. Then for any $\delta > 0$, the code \mathcal{F} is

$$\left(k' = (Q^2 k)^{Q^2}, \delta, \frac{\delta}{2(2k' + 1)(k' + 1)} \right)\text{-locally testable.}$$

Notice the above theorem implies local testability only when the field \mathbb{K} is relatively small, and is of interest only when $m \rightarrow \infty$. When m is small (and \mathbb{K} large) no general bounds were known on the locality of the tests. [18] show that it is possible to have affine invariant families with one 8-local constraint that cannot be characterized by $O(1)$ -local constraints. And all this previous work leaves open the possibility that there may exist other affine-invariant families that are $O(1)$ -locally characterized, perhaps even $O(1)$ -testable (say, over fields of growing size and $m = 1$), and do have large rate. Our work rules this out.

We can now state our main theorem which bounds the rate of affine invariant codes containing a *k*-local constraint.

Theorem 2 (Affine invariant families with a local constraint are contained in low-degree Reed-Muller codes). Let p be a prime and r, n, m be positive integers and let $q = p^r$ and $Q = q^n$. For $\mathbb{F} = \mathbb{F}_q$ and $\mathbb{K} = \mathbb{F}_Q$ a degree- n extension of \mathbb{F} , let $\mathcal{C} \subseteq \{\mathbb{K}^m \rightarrow \mathbb{F}\}$ be an affine-invariant family that satisfies a *k*-local constraint for $k \geq 2$. Then

1. The dimension of \mathcal{C} as a vector space over \mathbb{F}_q is at most $(mrn)^{k-2}$. Since the blocklength of \mathcal{C} is $Q^m = p^{rmm}$ we get

$$\dim(\mathcal{C}) \leq (m \log_p Q)^{k-2}.$$

2. \mathcal{C} is isomorphic to a subcode² of $\text{RM}[q, nm, (k - 2)q/p]$. In particular, for $q = p$ we get that \mathcal{C} is isomorphic to a subcode of $\text{RM}[p, nm, k - 2]$.

Note that when $q = p$, Part (1) of the theorem above follows from Part (2), since the dimension of $\text{RM}[p, nm, s]$ is at most $(mn)^s$. When $q = p^r$ for $r > 1$, this is not true, and the dimension of the code $\text{RM}[q, nm, sq/p]$ is much larger. In this case Part (2) is a weak description of our understanding of \mathcal{C} . A somewhat better

² In other words, there exists an isomorphism $\phi : \mathbb{F}^{nm} \rightarrow \mathbb{K}^m$ such that for every $f \in \mathcal{C}$, the function $(f \circ \phi) \in \{\mathbb{F}^{nm} \rightarrow \mathbb{F}\}$ defined by $(f \circ \phi)(x) = f(\phi(x))$ belongs to $\text{RM}[q, nm, (k - 2)q/p]$.

understanding of affine-invariant codes over \mathbb{F}_{p^r} , for $r > 1$ can be obtained if we use a broader class of codes. In particular, by viewing a code over \mathbb{F}_{p^r} as a code over the vector space \mathbb{F}_p^r , or as an r -tuple of codes over \mathbb{F}_p , one gets a more strict inclusion for such codes. Specifically, let $\text{RM}[p, n, k - 2]^r$ denote codes obtained by evaluations of $f = \langle f_1, \dots, f_r \rangle : \mathbb{F}_p^n \rightarrow \mathbb{F}_p^r$, where each $f_i : \mathbb{F}_p^n \rightarrow \mathbb{F}_p$ is an n -variate polynomial over \mathbb{F}_p of degree at most $k - 2$. We then have the following Corollary of Theorem 2.

Corollary 1 (Affine invariant families with a local constraint over fields of prime powers). *Let p be a prime and r, n, m be positive integers and let $q = p^r$ and $Q = q^n$. For $\mathbb{F} = \mathbb{F}_q$ and $\mathbb{K} = \mathbb{F}_Q$ the degree- n extension of \mathbb{F} , let $\mathcal{C} \subseteq \{\mathbb{K}^m \rightarrow \mathbb{F}\}$ be an affine-invariant family that satisfies a k -local constraint. Then for every \mathbb{F}_p -linear bijection $\psi : \mathbb{F}_q \rightarrow \mathbb{F}_p^r$, the code $\mathcal{C}' = \{\psi \circ f \mid f \in \mathcal{C}\} \subseteq \{\mathbb{K}^m \rightarrow \mathbb{F}_p^r\}$ is isomorphic to a subcode of $\text{RM}[p, nmr, k - 2]^r$.*

Proof. For $i \in [r]$, let \mathcal{C}_i be the projection of \mathcal{C}' to the i th coordinate. Then \mathcal{C}_i is a \mathbb{F}_p -linear, affine-invariant code (over the domain \mathbb{F}_Q^m). By Theorem 2 we get that it is isomorphic to a subcode of $\text{RM}[p, nmr, k - 2]$. It follows that \mathcal{C}' is a subcode of $\text{RM}[p, nmr, k - 2]^r$.

3 Proof of Main Theorems

In this section we prove Theorem 2 modulo some technical lemmas. It is not hard to show that if Theorem 2 holds for the case $m = 1$ then it holds for all positive integers m . (Proof omitted due to space limitations.)

From now on we consider only univariate functions, i.e., $\mathcal{C} \subseteq \{\mathbb{K} \rightarrow \mathbb{F}\}$. Recall that every function from $\mathbb{K} \rightarrow \mathbb{K}$ and hence from $\mathbb{K} \rightarrow \mathbb{F}$ is the evaluation of a polynomial in $\mathbb{K}[x]$ of degree at most $q^n - 1$. For a polynomial $g \in \mathbb{K}[x]$ given by $g(x) = \sum_d c_d x^d$, let $\text{supp}(g)$ denote its support, i.e., $\text{supp}(g) = \{d \mid c_d \neq 0\}$. The set of degrees in the support of the functions in \mathcal{C} turns out to be a central ingredient in understanding the structure of \mathcal{C} , motivating the following definition.

Definition 7 (Degree set of \mathcal{C}). *For a class of functions $\mathcal{C} \subseteq \{\mathbb{K} \rightarrow \mathbb{F}\}$, its degree set is the set $D(\mathcal{C}) = \cup_{g \in \mathcal{C}} \text{supp}(g)$.*

It turns out that the representations of elements of $D(\mathcal{C})$ in base p play a central role in the structure of affine-invariant families over fields of characteristic p . To this end we introduce some terminology.

For integer d , let $[d]_p = \langle d_0, d_1, \dots \rangle$ denotes its representation in base p (i.e., $0 \leq d_i < p$ and $d = \sum_{i=0}^\infty d_i p^i$). The p -weight of d , denoted $\text{wt}_p(d)$, is the quantity $\sum_{i=0}^\infty d_i$. We say e is in the p -shadow of d , denoted $e \leq_p d$, if $[e]_p = \langle e_0, e_1, \dots \rangle$ and $e_i \leq d_i$ for all i . The set $\{e \mid e \leq_p d\}$ is called the p -shadow of d . The following Lemma appears as Theorem 1 in [20].

Lemma 1. *For every affine invariant family $\mathcal{C} \subseteq \{\mathbb{K} \rightarrow \mathbb{F}\}$ where \mathbb{F}, \mathbb{K} are fields of characteristic p , $D(\mathcal{C})$ is closed under p -shadow, i.e., if $d \in D(\mathcal{C})$ and $e \leq_p d$ then $e \in D(\mathcal{C})$.*

3.1 Uniform Homogenous Diagonal Systems of Polynomial Equations

The task of finding the set of zeroes of a system of multivariate polynomial equations is a central theme in mathematics. (Linear algebra considers the special case where all equations are linear/affine and understanding the “variety” of a given system of (higher-degree) equations is a central theme in algebraic geometry.) In general of course, the set of zeroes may be too complex, even for degree two polynomials. Nevertheless, our quest to understand the locality of constraints in an affine-invariant property leads to such a question, where the set of polynomials has a reasonably clean description. Somewhat surprisingly, we are even able to describe the set of zeroes in a fairly precise way. We describe the class of polynomial systems that we consider next.

Definition 8 (Uniform Homogenous Diagonal (UHD) System). *Fix a system of polynomials $P_1, \dots, P_m \in \mathbb{F}[X_1, \dots, X_k]$.*

- *We say the system is homogenous if every polynomial in the system is homogenous.*
- *We say that the system is diagonal if every monomial in the support of every polynomial is a power of a single variable. I.e., a homogenous system is diagonal if for every $j \in [m]$, it is the case that $P_j(X_1, \dots, X_k) = \sum_{i=1}^k \lambda_{ji} \cdot X_i^{d_j}$.*
- *We say a homogenous diagonal system is uniform if the coefficients are the same for every polynomial, i.e., λ_{ji} is independent of j .*

We conclude that a uniform homogenous diagonal system is given by a sequence of coefficients $\Lambda = \langle \lambda_1, \dots, \lambda_k \rangle \in \mathbb{F}^k$ and degrees $D = \{d_1, \dots, d_m\}$ such that $P_j(X_1, \dots, X_k) = \sum_{i=1}^k \lambda_i X_i^{d_j}$. We refer to such a system as the (D, Λ) -UHD system. We say that the (D, Λ) -system has a pairwise-distinct solution over some field \mathbb{K} if there exist distinct values $\alpha_1, \dots, \alpha_k \in \mathbb{K}$ such that $P_j(\alpha_1, \dots, \alpha_k) = 0$ for every $j \in [m]$.

The following lemma motivates the study of UHD systems in our setting.

Lemma 2. *If an affine-invariant property $\mathcal{C} \subseteq \{\mathbb{K} \rightarrow \mathbb{F}\}$ has a k -local constraint, then there exists a non-zero vector $\Lambda \in \mathbb{F}^k$ such that the $(D(\mathcal{C}), \Lambda)$ -UHD system has a pairwise-distinct solution over \mathbb{K} .*

We omit the proof due to space considerations. The following theorem is the main technical claim of this paper.

Theorem 3 (Shadow-closed UHD systems containing nontrivial solutions have bounded weight). *Let \mathbb{F} be any field of characteristic p and let D be a p -shadow-closed set of integers containing an element d with $\text{wt}_p(d) \geq k$. Then for every $\Lambda = \langle \lambda_1, \dots, \lambda_k \rangle \in \mathbb{F}^k$ where not all λ_i ’s are zero, the (D, Λ) -UHD system has no pairwise-distinct solutions over \mathbb{K} for any field \mathbb{K} extending \mathbb{F} .*

3.2 Proof of Main Theorem

We are now ready to prove the main theorem assuming the lemmas claimed in the previous subsections.

Proof (Theorem 2). We know that it suffices to prove the theorem for the univariate case (i.e., $m = 1$). Let $D(\mathcal{C})$ be the degree set of \mathcal{C} . By Lemma 1, we know that $D(\mathcal{C})$ is p -shadow closed. Furthermore if \mathcal{C} has a k -local constraint then, by Lemma 2 there exists a non-zero vector $A \in \mathbb{F}^{k'}$ such that the $(D(\mathcal{C}), A)$ -UHD system has a pairwise-distinct solution. But then, by Theorem 3, we have that the weight of every element $d \in D(\mathcal{C})$ must be at most $k - 2$.

The dimension of \mathcal{C} , which is at most $|D(\mathcal{C})|$, can now be bounded from above by the number of integers $d \in \{0, \dots, q^n - 1\}$ of p -weight less than $k - 1$ which is (crudely) at most $(rn)^{k-2}$ (where $q = p^r$). It follows that \mathcal{C} is isomorphic to a subcode of $\text{RM}[q, nm, (k - 2)q/p]$, thus concluding the proof of the theorem.

3.3 Proof of Theorem 3

We now prove our main technical theorem, Theorem 3. The proof below is a simplification of our original proof and is due to Shachar Lovett. We start by introducing notation that will help us in the proof. The p -weight of a set of integers D , denoted $\text{wt}_p(D)$, is the maximal p -weight of an element in D . For the purposes of this proof, the length of the UHD system defined by $(D, A = (\lambda_1, \dots, \lambda_k))$ is k . We assume throughout the proof that $\lambda_1, \dots, \lambda_k$ are nonzero (otherwise, one can construct a UHD system over D with smaller length). We say that $\alpha = (\alpha_1, \dots, \alpha_k)$ is a solution of the UHD system if $\alpha_i \neq \alpha_j$ for all $i \neq j$ and α is a root of the UHD system. Using this terminology, Theorem 3 says

If the $(D, (\lambda_1, \dots, \lambda_k))$ -UHD system has a solution then $k > \text{wt}_p(D) + 1$.

Notice that it suffices to prove the theorem for the case where $D = \text{shadow}_p(d)$ for some integer d . (Else we can simply take the element d of largest weight in D and work with the set $D' = \text{shadow}_p(d)$.)

We prove this by induction on $\text{wt}_p(d)$. The base case of $\text{wt}_p(d) = 0$ says that if the $(\text{shadow}_p(d), A)$ -UHD system is of p -weight 0 and length 1 then it has no solution. The proof in this case is immediate because $\text{shadow}_p(d) = \{0\}$, so if $A = (\lambda_1)$ is nonzero then there is no solution to the system $\lambda_1 X^0 = 0$ (recall $0^0 = 1$).

For the inductive case we have $\text{wt}_p(d) > 0$. Let $[d]_p = \langle d_0, d_1, \dots \rangle$ be the base- p representation of d and suppose $d_j > 0$. Assume by way of contradiction that $\alpha = (\alpha_1, \dots, \alpha_k)$ is a solution to the $(\text{shadow}_p(d), A)$ -UHD system of length k . The base-case of the induction shows $k > 1$ because α is also a solution to the $(\{0\}, A)$ -UHD system (as $0 \in \text{shadow}_p(d)$), so we may assume without loss of generality that $\alpha_k \neq 0$ because all α_i are distinct. Our proof goes by showing that $\alpha' = (\alpha_1, \dots, \alpha_{k-1})$ is a solution of a UHD-system of p -weight $w = \text{wt}_p(d) - 1$. By the inductive hypothesis we have $k - 1 > w$ which implies $k > \text{wt}_p(d) + 1$ and completes the proof.

To construct the said UHD system set $e = d - p^j$, noticing $\text{wt}_p(e) = \text{wt}_p(d) - 1$ and let $E = \text{shadow}_p(e)$. Construct $A' = (\lambda'_1, \dots, \lambda'_{k-1})$ by setting

$$\lambda'_i = \lambda_i(\alpha_i^{p^j} - \alpha_k^{p^j}).$$

Notice $\lambda'_i \neq 0$ because $\lambda_i \neq 0$ and $\alpha_i \neq \alpha_k$ and the transformation $\alpha \mapsto \alpha^{p^j}$ is a bijection on \mathbb{K} . We shall now show that $(\alpha_1, \dots, \alpha_{k-1})$ is a solution of length $k - 1$ to the (E, A') -UHD system of p -weight $\text{wt}_p(d) - 1$ thereby completing the proof of the theorem. To show that $(\alpha_1, \dots, \alpha_{k-1})$ is a solution we argue that for all $r \in \text{shadow}_p(e)$ we have $\sum_{i=1}^{k-1} \lambda'_i \alpha_i^r = 0$. Indeed

$$\begin{aligned} \sum_{i=1}^{k-1} \lambda'_i \alpha_i^r &= \sum_{i=1}^{k-1} \lambda_i (\alpha_i^{p^j} - \alpha_k^{p^j}) \alpha_i^r \\ &= \sum_{i=1}^k \lambda_i (\alpha_i^{p^j} - \alpha_k^{p^j}) \alpha_i^r \\ &= \sum_{i=1}^k \lambda_i \alpha_i^{r+p^j} - \alpha_k^{p^j} \sum_{i=1}^k \lambda_i \alpha_i^r = 0 \end{aligned}$$

The last equality follows because $r + p^j \in \text{shadow}_p(d)$ for all $r \in \text{shadow}_p(e)$. This completes the proof of the theorem.

Acknowledgements. We thank the anonymous referees for helpful comments. We thank Shachar Lovett for allowing us to include the simplified proof of Theorem 3 in this paper.

References

1. Alon, N., Fischer, E., Newman, I., Shapira, A.: A combinatorial characterization of the testable graph properties: it's all about regularity. In: STOC 2006, pp. 251–260 (2006)
2. Alon, N., Kaufman, T., Krivelevich, M., Litsyn, S., Ron, D.: Testing Reed-Muller codes. *IEEE Transactions on Information Theory* 51(11), 4032–4039 (2005)
3. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and the hardness of approximation problems. *Journal of the ACM* 45(3), 501–555 (1998)
4. Arora, S., Safra, S.: Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM* 45(1), 70–122 (1998)
5. Babai, L., Fortnow, L., Levin, L.A., Szegedy, M.: Checking computations in poly-logarithmic time. In: STOC 1991, pp. 21–32 (1991)
6. Ben-Sasson, E., Guruswami, V., Kaufman, T., Sudan, M., Viderman, M.: Locally testable codes require redundant testers. In: CCC 2009, pp. 52–61 (2009)
7. Ben-Sasson, E., Harsha, P., Raskhodnikova, S.: Some 3CNF properties are hard to test. *SIAM J. Comput.* 35(1), 1–21 (2005)
8. Ben-Sasson, E., Maatouk, G., Shpilka, A., Sudan, M.: Symmetric LDPC codes are not necessarily locally testable. In: CCC (2011)

9. Ben-Sasson, E., Sudan, M.: Simple PCPs with poly-log rate and query complexity. In: STOC 2005, pp. 266–275 (2005)
10. Bhattacharyya, A., Kopparty, S., Schoenebeck, G., Sudan, M., Zuckerman, D.: Optimal testing of Reed-Muller codes. In: FOCS 2010, pp. 488–497 (2010)
11. Blum, M., Luby, M., Rubinfeld, R.: Self-testing/correcting with applications to numerical problems. In: STOC 1982, pp. 73–83 (1982)
12. Borgs, C., Chayes, J.T., Lovász, L., Sós, V.T., Szegedy, B., Vesztergombi, K.: Graph limits and parameter testing. In: STOC 2006, pp. 261–270 (2006)
13. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private information retrieval. *Journal of the ACM* 45, 965–981 (1998)
14. Dinur, I.: The PCP theorem by gap amplification. *Journal of the ACM* 54(3), 12:1–12:44 (2007)
15. Efremenko, K.: 3-query locally decodable codes of subexponential length. In: STOC 2009, pp. 39–44 (2009)
16. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *J. ACM* 45(4), 653–750 (1998)
17. Goldreich, O., Sudan, M.: Locally testable codes and PCPs of almost-linear length. *J. ACM* 53(4), 558–655 (2006)
18. Grigorescu, E., Kaufman, T., Sudan, M.: 2-transitivity is insufficient for local testability. In: CCC 2008, pp. 259–267 (2008)
19. Jutla, C.S., Patthak, A.C., Rudra, A., Zuckerman, D.: Testing low-degree polynomials over prime fields. In: FOCS 2004, pp. 423–432 (2004)
20. Kasami, T., Lin, S., Peterson, W.: Some results on cyclic codes which are invariant under the affine group and their applications. *Information and Control* 11(5-6), 475–496 (1967)
21. Katz, J., Trevisan, L.: On the efficiency of local decoding procedures for error-correcting codes. In: STOC 2000, pp. 80–86 (2000)
22. Kaufman, T., Ron, D.: Testing polynomials over general fields. In: FOCS 2004, pp. 413–422 (2004)
23. Kaufman, T., Sudan, M.: Algebraic property testing: the role of invariance. In: STOC 2008, pp. 403–412 (2008)
24. Polishchuk, A., Spielman, D.A.: Nearly-linear size holographic proofs. In: STOC 1994, pp. 194–203 (1994)
25. Rubinfeld, R., Sudan, M.: Robust characterizations of polynomials with applications to program testing. *SIAM J. Comput.* 25(2), 252–271 (1996)
26. Samorodnitsky, A.: Low-degree tests at large distances. In: STOC 2007, pp. 506–515 (2007)
27. Sudan, M.: Efficient checking of polynomials and proofs and the hardness of approximation problems. PhD thesis, UC Berkeley (1992)
28. Woodruff, D.: New lower bounds for general locally decodable codes. In: *Electronic Colloquium on Computational Complexity (ECCC)*, vol. (006) (2007)
29. Yekhanin, S.: Towards 3-query locally decodable codes of subexponential length. *Journal of the ACM* 55, 1–16 (2008)
30. Yekhanin, S.: Locally decodable codes. In: *Foundations and Trends in Theoretical Computer Science* (2011)

The Computational Complexity of Estimating MCMC Convergence Time

Nayantara Bhatnagar^{1,*}, Andrej Bogdanov^{2,**}, and Elchanan Mossel^{1,3,4,***}

¹ Department of Statistics, University of California, Berkeley

² Department of Computer Science and Engineering, The Chinese University of Hong Kong

³ Electrical Engineering and Computer Sciences, University of California, Berkeley

⁴ Faculty of Mathematics and Computer Science, Weizmann Institute of Science

Abstract. An important problem in the implementation of Markov Chain Monte Carlo algorithms is to determine the convergence time, or the number of iterations before the chain is close to stationarity. For many Markov chains used in practice this time is not known. There does not seem to be a general technique for upper bounding the convergence time that gives sufficiently sharp (useful in practice) bounds in all cases of interest. Thus, practitioners like to carry out some form of statistical analysis in order to assess convergence. This has led to the development of a number of methods known as convergence diagnostics which attempt to diagnose whether the Markov chain is far from stationarity. We study the problem of testing convergence in the following settings and prove that the problem is hard computationally:

- Given a Markov chain that mixes rapidly, it is hard for Statistical Zero Knowledge (SZK-hard) to distinguish whether starting from a given state, the chain is close to stationarity by time t or far from stationarity at time ct for a constant c . We show the problem is in $AM \cap coAM$.
- Given a Markov chain that mixes rapidly it is $coNP$ -hard to distinguish from an arbitrary starting state whether it is close to stationarity by time t or far from stationarity at time ct for a constant c . The problem is in $coAM$.
- It is PSPACE-complete to distinguish whether the Markov chain is close to stationarity by time t or still far from stationarity at time ct for $c \geq 1$.

1 Introduction

Markov Chain Monte Carlo (MCMC) simulations are an important tool for sampling from high dimensional distributions in Bayesian inference, computational physics and biology and in applications such as image processing. An important problem that arises in the implementation is that if bounds on the convergence time are not known or impractical for simulation then one would like a method for determining if the chain is still far from its stationary distribution.

* Supported by DOD ONR grant N0014-07-1-05-06 and DMS 0528488. Part of this work was done while visiting the Faculty of Mathematics and Computer Science, Weizmann Institute.

** Supported by the National Basic Research Program of China Grant (973 Project No. 2007 CB 807900) and a CUHK Direct Faculty Grant.

*** Supported by DMS 0548249 (CAREER) award, DOD ONR grant N0014-07-1-05-06, ISF grant 1300/08 and EU grant PIRG04-GA-2008-239317.

A number of techniques are known to theoretically bound the rate of convergence time as measured by the *mixing time* of a Markov chain, see e.g. [11,13,16]. These have been applied with success to problems such as volume estimation [17], Monte Carlo integration of log-concave functions [18], approximate counting of matchings [15] and estimation of partition functions from physics [14]. However, in most practical applications of MCMC there are no effective bounds on the convergence time. For a chain on 2^{100} states whose states are encoded by 100 bits, it may not be known if the mixing time is $100^2 = 10,000$ or $2^{100/2} = 2^{50}$. Even when rapid mixing is known, the bounds are often impractical. In the above example, a polynomial mixing bound of 100^{10} may be impractical while the actual mixing time may be 100^2 .

As a result, practitioners have focused on the development of a large variety of statistical methods, called convergence diagnostics, which try to determine whether the Markov chain is far from stationarity (see e.g. surveys by [12,15,7,6,8,19]). A majority of practitioners of the MCMC method run multiple diagnostics to test if the chains have converged. The two most popularly used public domain diagnostic software packages are CODA and BOA [20,4]. The idea behind many of the methods is to use the samples from the empirical distribution obtained when running one or multiple copies of the chain, possibly from multiple starting states to compute various functionals and identify non-convergence. While diagnostics are commonly used for MCMC, it has been repeatedly argued that they cannot guarantee convergence, see e.g. [7,15,2].

We formalize convergence to stationarity detection as an algorithmic problem and study its complexity in terms of the size n of the implicit description of the Markov chain. We consider Markov chains whose state spaces have size exponential in n . Our main contribution is showing that even in cases where the mixing time of the chain is known to be bounded by n^C for some large C , the problem of distinguishing whether a Markov chain is close to or far from stationarity at time n^c for c much smaller than C is “computationally hard”. In other words, under standard assumptions in computational complexity, distinguishing whether the chain is close to or far from stationarity cannot be solved in time n^D for any constant D .

The strength of our results is in their generality as they apply to *all possible* diagnostics and in the weakness of the assumption - in particular in assuming that the mixing time of the chain is not too long and that the diagnostic is also given the initial state of the chain. Our results highlight the role of the complexity classes Statistical Zero Knowledge, AM, coAM and coNP in the computational study of MCMC.

2 Preliminaries and Results

We begin by defining the mixing time which measures the rate of convergence to the stationary distribution. Recall that the *variation distance* (or statistical distance) between two probability distributions μ and ν on a state space Ω is given by $d_{tv}(\mu, \nu) = \frac{1}{2} \sum_{\omega \in \Omega} |\mu(\omega) - \nu(\omega)|$.

Definition 1 (Mixing time). *Let M be a Markov chain with state space Ω , transition matrix P and a unique stationary distribution π . Let*

$$d(t) := \max_{x, y \in \Omega} d_{tv}(P^t(x, \cdot), P^t(y, \cdot)).$$

The ε -mixing time is defined to be $\tau(\varepsilon) := \min\{t : d(t) \leq \varepsilon\}$. We refer to $\tau(1/4)$ as the mixing time. The ε -mixing time starting from x is:

$$\tau_x(\varepsilon) := \min\{t : d_{tv}(P^t(x, \cdot), \pi) \leq \varepsilon\}.$$

We note that $\tau_x(\varepsilon) \leq \tau(\varepsilon)$ for all x (see e.g. [17] Chap. 2, Lem. 20).

To formulate the problem, we think of the Markov chain as a “rule” for determining the next state of the chain given the current state and some randomness.

Definition 2. We say that a circuit $C : \{0, 1\}^n \times \{0, 1\}^m \rightarrow \{0, 1\}^n$ specifies P if for every pair of states $x, y \in \Omega$, $\Pr_{r \sim \{0, 1\}^m} [C(x, r) = y] = P(x, y)$.

Above, x is the “current state”, r is the “randomness”, y is the “next state” and C is the “rule”. To formalize the notion of “Testing convergence” we imagine the practitioner has a time t in mind that she would like to run the Markov chain algorithm for. She would like to use the diagnostic to determine whether at time t :

- The chain is within (say) $1/4$ variation distance of stationarity
- or at least at distance $1/4$ away from it.

Requiring the diagnostic to determine the total variation at time t exactly, or even to distinguish mixing at exactly the time t is not needed in many situations. Thus weaker requirement for the diagnostic is to:

- Declare the chain has mixed if it is within $1/8$ variation distance of stationarity at time t .
- Declare it did not mix if it is at least at distance $1/2$ away from it at time ct , where $c \geq 1$. (The diagnostic may behave in an arbitrary manner if the distance is between $1/8$ and $1/2$).

In this formulation, the practitioner is satisfied with an approximate output of the diagnostics both in terms the time and in terms of the total variation distance. This is the problem we will study. In fact, we will make the requirement from the diagnostic even easier by providing it with a (correct) bound on the actual mixing time of the chain. This bound will be denoted by t_{\max} .

In realistic settings it is natural to measure the running time of the diagnostics in relation to the running time of the chain itself as well as to the size of the chain. In particular it is natural to consider diagnostics that would run for time that is polynomial in t and t_{\max} . The standard way to formalize such a requirement is to insist that the inputs t, t_{\max} to the diagnostic algorithm to be given in unary form (if t, t_{\max} were specified as binary numbers, an efficient algorithm would be required to run in time *poly-logarithmic* in these parameters, a much stronger requirement).

2.1 Given Starting Point

The discussion above motivates the definition of the first problem below. Assume that we had a diagnostic algorithm. As input, it would take the tuple $(C, x, 1^t, 1^{t_{\max}})$, i.e., a description of the circuit which describes the moves of the Markov chain, an initial state for the chain, and the times t and t_{\max} , which are specified as unary numbers.

Problem: GAPPOLYTESTCONVERGENCEWITHSTART $_{c,\delta}$ (GPTCS $_{c,\delta}$).

Input: $(C, x, 1^t, 1^{t_{\max}})$, where C is a circuit specifying a Markov chain P on state space $\Omega \subseteq \{0, 1\}^n$, $x \in \Omega$ and $t, t_{\max} \in \mathbb{N}$.

Promise: The Markov chain P is ergodic and $\tau(1/4) \leq t_{\max}$.

YES instances: $\tau_x(1/4 - \delta) < t$.

NO instances: $\tau_x(1/4 + \delta) > ct$.

Informally the input to this problem is the MC rule C , a starting state x , and times t and t_{\max} . It is promised that the chain mixes by time t_{\max} . The expectation from the diagnostic is to:

- Declare the chain has mixed if it is within $1/4 - \delta$ variation distance of stationarity at time t .
- Declare it did not mix if it is at least at distance $1/4 + \delta$ away from it at time ct , where $c > 1$.

The choice of the constant $1/4$ above is arbitrary and does not affect the nature of the results except in the exact constants. Note again that the diagnostic is given room for error both in terms of the total variation distance and in terms of the time. The following theorem refers to the complexity class SZK, which is the class of all promise problems that have statistical zero-knowledge proofs with completeness $2/3$ and soundness $1/3$. It is believed that these problems cannot be solved in polynomial time.

Theorem 1. *Let $c \geq 1$.*

- For $0 < \delta \leq 1/4$, GPTCS $_{c,\delta}$ is in $\text{AM} \cap \text{coAM}$.
- For $\frac{\sqrt{3}-3/2}{2} = .116025.. < \delta \leq 1/4$ and $c \leq n^{O(1)}$, GPTCS $_{c,\delta}$ is in SZK.
- Let $0 \leq \delta < 1/4$. For $c < (t_{\max}/4t) \ln(2/(1+4\delta))$, GPTCS $_{c,\delta}$ is SZK-hard.

The most interesting part of the theorem is the last part which says that the problem GPTCS $_{c,\delta}$ is SZK-hard. In other words, solving it in polynomial time will result in solving all the problems in SZK in polynomial time. The second part of the theorem states that for some values of δ this is the “exact” level of hardness. The first part of the theorem states that without restrictions on δ the problem belongs to the class $\text{AM} \cap \text{coAM}$ (which contains the class SZK). The classes AM and coAM respectively contain the classes NP and coNP and it is believed that they are equal to them, but this is as yet unproven.

The restriction on the constant δ in the second part of the result comes from the fact that the proof is by reduction to the SZK-complete problem STATISTICAL DISTANCE (SD, see Section 3 for precise definitions). Holenstein and Renner give evidence in [11] that SD is in SZK only when there is a lower bound on the gap between the completeness and soundness. We show that without the restriction in Theorem 1, SD is in SZK for a smaller value of the completeness-soundness gap.

On the other hand, we can show a slightly weaker result and put GPTCS $_{c,\delta}$ into $\text{AM} \cap \text{coAM}$ without any restrictions on δ . To show this, we first prove that SD is in $\text{AM} \cap \text{coAM}$ when no restriction is put on the gap between the completeness and soundness. This result may be interesting in its own right as it involves showing protocols for STATISTICAL DISTANCE that are new, to our knowledge.

2.2 Arbitrary Starting Point

So far we have discussed mixing from a given starting point. A desired property of a Markov chain is fast mixing from an arbitrary starting point. Intuitively, this problem is harder than the previous one since it involves all starting points. This is consistent with our result below where we obtain a stronger hardness.

Problem: $\text{GAPPOLYTESTCONVERGENCE}_{c,\delta}$ ($\text{GPTC}_{c,\delta}$).

Input: $(C, x, 1^t, 1^{t_{\max}})$, where C is a circuit specifying a Markov chain P on state space $\Omega \subseteq \{0, 1\}^n$, $x \in \Omega$ and $t, t_{\max} \in \mathbb{N}$.

Promise: The Markov chain P is ergodic and $\tau(1/4) \leq t_{\max}$.

YES instances: $\tau(1/4 - \delta) < t$.

NO instances: $\tau(1/4 + \delta) > ct$.

The only difference between this and the previous problem is that the total variation distance is measured from the worst starting point instead of from a given starting point.

Theorem 2. *Let $c \geq 1$.*

- For $0 < \delta \leq 1/4$, $\text{GPTC}_{c,\delta} \in \text{coAM}$.
- Let $0 \leq \delta < 1/4$. For $c < \frac{3/4-\delta}{2} \sqrt{\frac{t_{\max}}{t^2}} n^3$ it is coNP-hard to decide $\text{GPTC}_{c,\delta}$.

The second part of the theorem shows that the diagnostic problem is coNP hard so it is very unlikely to be solved in polynomial time. This hardness is stronger than SZK-hardness because SZK is unlikely to contain coNP-hard problems. If it did, this would imply that $\text{NP} = \text{coNP}$ since $\text{SZK} \subseteq \text{AM}$ and it is believed that $\text{AM} = \text{NP}$. The first part of the theorem shows that the problem is always in coAM.

2.3 Arbitrary Mixing Times

We also consider the case without the restriction that the algorithm should be polynomial in the times t, t_{\max} . This corresponds to situations where the mixing time of the chain may be exponentially large in the size of the rule defining the chain. While this rules out many situations of practical interest, it is relevant in scenarios where analysis of the mixing time is of *theoretical* interest. For example there is extensive research in theoretical physics on the rate of convergence of Gibbs samplers on spin glasses in cases where the convergence rate is very slow (see [9] and follow-up work). It is natural to define the problem as follows:

Problem $\text{GAPTTESTCONVERGENCE}_{c,\delta}$ ($\text{GTC}_{c,\delta}$).

Input: (C, t) , where C is a circuit specifying a Markov chain P on state space $\Omega \subseteq \{0, 1\}^n$, and $t \in \mathbb{N}$.

Promise: The Markov chain P is ergodic.

YES instances: $\tau(1/4 - \delta) < t$.

NO instances: $\tau(1/4 + \delta) > ct$.

The main difference is that in this problem the time t is given in binary representation. Thus, informally in this case the efficiency is measured with respect to the logarithm of t . The mixing time of the chain itself does not put any restrictions on the diagnostic. We prove the following result:

Theorem 3. *Let $1 \leq c \leq \exp(n^{O(1)})$.*

- *For $\exp(-n^{O(1)}) < \delta \leq 1/4$ it is in PSPACE to decide $GTC_{c,\delta}$.*
- *Let $0 \leq \delta < 1/4$, then, it is PSPACE-hard to decide $GTC_{c,\delta}$.*

It is known that PSPACE-hard problems are at least as hard as all the problems in polynomial time, coNP, NP and all other problems in the polynomial hierarchy. The proof of this result can be found in [3, App. C]. Since the case of arbitrary mixing times is of lesser practical interest, we focus in this extended abstract on the case when the mixing time is known to be polynomial. We begin with some results on the problem of estimating the statistical distance between two distributions.

3 Protocols for Statistical Distance

Given a circuit $C: \{0, 1\}^n \rightarrow \{0, 1\}^n$, the probability distribution p associated with C assigns probability $p(\omega) = |C^{-1}(\omega)|/2^n$ to every $\omega \in \{0, 1\}^n$. We will be interested in estimating the statistical distance between the distributions associated with a pair of circuits $C, C': \{0, 1\}^n \rightarrow \{0, 1\}^n$. Denote those distributions by p and p' , respectively.

For a pair of constants $0 \leq s < c \leq 1$, $SD_{c,s}$ is defined to be the following promise problem. The inputs are pairs of circuits $C, C': \{0, 1\}^n \rightarrow \{0, 1\}^n$, the YES instances satisfy $d_{tv}(p, p') \geq c$, and the NO instances satisfy $d_{tv}(p, p') < s$.

Sahai and Vadhan [22] show that for every pair of constants c, s the problem $SD_{c,s}$ is SZK-hard. They also show that when $c^2 > s$, $SD_{c,s}$ is in SZK. Our theorem yields a weaker conclusion, but covers a wider spectrum of parameters.

Theorem 4. *For any pair of constants $0 \leq s < c \leq 1$, $SD_{c,s}$ is in $AM \cap coAM$.*

3.1 An AM protocol

The following interactive protocol P for $SD_{c,s}$ essentially appears in [22] but we rewrite it here for the precise parameters we need.

- V:** Flip a fair coin. If heads, generate a random sample from C . If tails, generate a random sample from C' . Send the sample x to the prover.
- P:** Say if x came from C or from C' .
- V:** If prover is correct accept, otherwise reject.

Claim. Protocol P is an interactive proof for $SD_{c,s}$ with completeness $1/2 + c$ and soundness $1/2 + s$.

See [3, App. A.1] for the proof.

3.2 A coAM protocol

A coAM protocol for $SD_{c,s}$ should accept when the statistical distance between p and p' is small, and reject when the statistical distance is large. To develop some intuition, let us first attempt to distinguish the cases when p and p' are the same distribution (i.e. $s = 0$) and the case when they are at some distance from one another (say $c = 1/2$). Suppose the verifier could get hold of the values

$$N(t) = |\{\omega : |C^{-1}(\omega)| \geq t \text{ and } |C'^{-1}(\omega)| \geq t\}|$$

for every t (which could potentially range between 0 and 2^n). Then it can compute the desired statistical distance via the following identity (proof in [3] App. A.2).

$$\sum_{t=1}^{2^n} t \cdot (N(t) - N(t + 1)) = (1 - d_{tv}(p, p')) \cdot 2^n. \tag{1}$$

For the verifier to run in polynomial time, there are two issues with this strategy: First, the verifier does not have time to compute the values $N(t)$. Secondly, the verifier cannot evaluate the exponentially long summation in (1). If we only want to compute the statistical distance approximately, the second issue can be resolved by quantization: instead of computing the sum on the left for all the values of t , the verifier chooses a small number of representative values and estimates the sum approximately. For the first issue, the verifier will rely on the prover to provide (approximate) values for $N(t)$. While the verifier cannot make sure that the values provided by a (cheating) prover will be exact, she will be able to ensure that the prover never grossly over-estimates the sum on the left by running a variant of the Goldwasser-Sipser protocol for lower bounding the size of a set [10]. Since the sum on the left is proportional to one minus the statistical distance, it will follow that no matter what the prover’s strategy is, he cannot force the verifier to significantly underestimate the statistical distance without being detected and we have the desired result. The protocol for lower bounding $N(t)$ and the details of the coAM protocol can be found in [3] App. A.2].

4 Diagnosing Convergence for Polynomially Mixing Chains

The results of this section imply that even if the mixing time is restricted to being polynomial the diagnostic problem remains hard. The two cases we consider are the worst case start mixing time and the mixing time from a given starting state. Both hardness results are by reduction from a complete problem in the respective classes. We first prove Theorem 1.

Lemma 1. *The problem $GPTCS_{c,\delta}$ is in SZK for $1 \leq c \leq n^{O(1)}$ and $\frac{\sqrt{3}-3/2}{2} = .116025\dots < \delta \leq 1/4$.*

Proof. The proof is by reduction to $SD_{c,s}$ where c and s are chosen as follows. Choose k , a parameter as follows. First, let k be large enough such that

$$\left(\frac{1}{4} + \delta - \frac{1}{k}\right)^2 > \frac{1}{4} - \delta + \frac{1}{k}. \tag{2}$$

It can be verified that $k > 2/(\delta - (\sqrt{3} - 3/2)/2)$ suffices. Secondly, let k be bounded by $\exp(n^{O(1)})$, but large enough so that $ct \leq t_{\max} \ln k / \ln 4$. This is possible since $c \leq n^{O(1)}$. Let

$$\mathbf{s} = \frac{1}{4} - \delta + \frac{1}{k} \text{ and } \mathbf{c} = \frac{1}{4} + \delta - \frac{1}{k}.$$

Suppose we are given an instance of $\text{GPTCS}_{c,\delta}$ with input $(C, x, 1^t, 1^{t_{\max}})$. Let $\hat{\tau} = \tau_x(1/k)$ be the time to come within $1/k$ in variation distance of the stationary distribution. Let C' output the distribution $P^t(x, \cdot)$ over Ω . Let C'' output the distribution $P^{\hat{\tau}}(x, \cdot)$ over Ω . The circuits C' and C'' for k chosen as above are polynomial-size. The former because t is a polynomial and the latter by the following argument (using e.g. [1] Ch 2, Lemma 20): The distance $d(\cdot)$ is submultiplicative. Hence $d(\ln(k)t_{\max}/\ln(4)) \leq 1/k$. Also, for any time t , $d_{tv}(P^t(x, \cdot), \pi) \leq d(t)$. Thus for $\hat{\tau} = \ln(k)t_{\max}/\ln(4)$, we have $d_{tv}(P^{\hat{\tau}}(x, \cdot), \pi) \leq 1/k$. Since $\ln k \leq n^{O(1)}$, $\hat{\tau}$ is polynomial, and the size of C'' is bounded by a polynomial. In the YES case,

$$d_{tv}(P^t(x, \cdot), P^{\hat{\tau}}(x, \cdot)) \leq \frac{1}{4} - \delta + \frac{1}{k}$$

while in the NO case,

$$d_{tv}(P^{ct}(x, \cdot), P^{\hat{\tau}}(x, \cdot)) > \frac{1}{4} + \delta - \frac{1}{k}.$$

Since $t \leq ct \leq \hat{\tau}$ and the distance to stationarity decreases (see e.g. [1] Chap. 2, Lem. 20),

$$d_{tv}(P^t(x, \cdot), P^{\hat{\tau}}(x, \cdot)) > \frac{1}{4} + \delta - \frac{1}{k}.$$

By [2], the constructed instance of $\text{SD}_{c,s}$ is in SZK and the lemma follows. □

Lemma 2. *The problem $\text{GPTCS}_{c,\delta}$ is in $\text{AM} \cap \text{coAM}$ for all $c \geq 1$ and $0 < \delta \leq 1/4$.*

This part of the result follows directly from Theorem 4 by reducing $\text{GPTCS}_{c,\delta}$ to $\text{SD}_{c,s}$ as above, without the restriction on the gap between \mathbf{c} and \mathbf{s} .

We provide evidence that the gap for δ in Lemma 1 is required for membership in SZK (see [3] App. B, Prop. 1). Sahai and Vadhan [22] show that when $c^2 > s$, $\text{SD}_{c,s}$ is in SZK. Holenstein and Renner [11] give evidence that this condition on the gap between \mathbf{c} and \mathbf{s} is in fact essential for membership in SZK, assuming $\text{SZK} \neq \text{NP}$. We now complete the proof of Theorem 1.

Lemma 3. *Let $0 \leq \delta < 1/4$. For $1 \leq c < \frac{t_{\max}}{4t} \ln \left(\frac{2}{1+4\delta} \right)$, $\text{GPTCS}_{c,\delta}$ is SZK-hard.*

Proof. The proof is by reduction from $\text{SD}_{c,s}$ for fixed \mathbf{s}, \mathbf{c} (specified below) to $\text{GPTCS}_{c,\delta}$. Let (C, C') be an instance of $\text{SD}_{c,s}$ where C and C' are circuits which output distributions μ_1 and μ_2 over $\{0, 1\}^n$. Construct the Markov chain P , whose state space is $[m] \times \{0, 1\}^n$ where $m = p(n)$ is a polynomial in n . The transitions of the chain are defined as follows. Let the current state be (X_t, Y_t) where $X_t \in [m]$ and $Y_t \in \{0, 1\}^n$.

- If $X_t = 1$, choose Y_{t+1} according to μ_1 .
- If $X_t = 2$, choose Y_{t+1} according to μ_2 .
- Otherwise, set $Y_{t+1} = Y_t$.
- Choose X_{t+1} uniformly at random from $[m]$.

The stationary distribution of the chain is given by $\pi(z, y) = \frac{1}{m}(\frac{1}{2}\mu_1(y) + \frac{1}{2}\mu_2(y))$. Take the starting state to be $x = (1, 0^n)$. In one step, the total variation distance from stationary can be bounded as

$$d_{tv}(P(x, \cdot), \pi) = \frac{1}{2}d_{tv}(\mu_1, \mu_2)$$

and after t steps, the distance is given by

$$d_{tv}(P^t(x, \cdot), \pi) = \frac{1}{2} \left(\frac{m-2}{m} \right)^{t-1} d_{tv}(\mu_1, \mu_2) \tag{3}$$

Choose $m \geq 3$. Set $\mathbf{s} = 1/4 - \delta$ and $\mathbf{c} = 1$. In the YES case, $d_{tv}(\mu_1, \mu_2) < \mathbf{s}$ and hence for any $t \geq 1$,

$$d_{tv}(P^t(x, \cdot), \pi) < \frac{1}{2}\mathbf{s} < \frac{1}{4} - \delta \tag{4}$$

In the NO case, $d_{tv}(\mu_1, \mu_2) > \mathbf{c}$ and hence

$$d_{tv}(P^{ct}(x, \cdot), \pi) \geq \frac{1}{2} \left(\frac{m-2}{m} \right)^{ct-1} \mathbf{c} \geq \frac{1}{2} \left(\frac{m-2}{m} \right)^{ct-1}. \tag{5}$$

Since $m \geq 3$, if $ct < \frac{m}{4} \ln \left(\frac{2}{1+4\delta} \right)$, then $d_{tv}(P^{ct}(x, \cdot), \pi) > \frac{1}{4} + \delta$. Further, in both the YES and NO case, $\tau(1/4) \leq m$. We conclude the reduction by setting $t_{\max} = m$. □

We now prove Theorem 2 which classifies the complexity of diagnosing mixing from an arbitrary starting state of a polynomially mixing chain. The following result relates mixing time to the *conductance*.

Definition 3 (Conductance, see e.g. [21]). Let M be a Markov chain corresponding to the random walk on an edge weighted graph with edge weights $\{w_e\}$. Let $d_x = \sum_{y \sim x} w_{xy}$ denote the weighted degree of a vertex x . Define the conductance of M to be $\Phi(M) := \min_{\emptyset \neq A \subsetneq \Omega} \Phi_A(M)$ where

$$\Phi_A(M) := \frac{\sum_{x \in A, y \in A^c} w_{xy}}{\sum_{x \in A} d_x} \tag{6}$$

Theorem 5 (see [21]). Let M be a Markov chain corresponding to the random walk on an edge weighted graph with edge weights $\{w_e\}$ as above. Let π be the stationary distribution of the Markov chain.

$$\tau(\varepsilon) \leq \frac{2}{\Phi^2(M)} \log \left(\frac{2}{\pi_{\min} \varepsilon} \right)$$

where π_{\min} is the minimum stationary probability of any vertex.

Lemma 4. For every $c \geq 1, 0 < \delta \leq 1/4$, $\text{GPTC}_{c,\delta}$ is in coAM.

Proof. In the first step of the coAM protocol for $\text{GPTC}_{c,\delta}$ the prover sends a pair $x, y \in \Omega$ that maximizes $d_{tv}(P^t(x, \cdot), P^t(y, \cdot))$. Let C_x be the circuit which outputs the distribution $P^t(x, \cdot)$ and let C_y output the distribution $P^t(y, \cdot)$.

In the YES case $\tau(1/4 - \delta) < t$ and for every $x, y, d_{tv}(P^t(x, \cdot), P^t(y, \cdot)) < 1/4 - \delta$. In the NO case, $\tau(1/4 + \delta) > ct$ and $c \geq 1$, therefore there must exist x, y such that $d_{tv}(P^t(x, \cdot), P^t(y, \cdot)) > 1/4 + \delta$.

By Claim 3.1 there is an AM protocol P for $\text{SD}_{1/4+\delta, 1/4-\delta}$ with completeness $3/4 + \delta$ and soundness $3/4 - \delta$. The prover and the verifier now engage in the AM protocol to distinguish whether the distance between the two distributions is large or small. The completeness and soundness follow from those of the protocol P . \square

Lemma 5. Let $0 \leq \delta < 1/4$. For $1 \leq c < 1/2\sqrt{t_{\max}/t^2 n^3}(3/4 - \delta)$, it is coNP-hard to decide $\text{GPTC}_{c,\delta}$.

Proof. The proof is by reduction from UNSAT, which is coNP hard. Let ψ be an instance of UNSAT, that is, a CNF formula on n variables. The vertices of the Markov chain are the vertices of the hypercube $H, V(H) = \{0, 1\}^n$ and edges $E(H) = \{(y_1, y_2) : |y_1 - y_2| = 1\}$. We set edge weights for the Markov chain as follows. Let d be a parameter to be chosen later which is at most a constant.

- For each edge in $E(H)$ set the weight to be 1.
- If $\psi(y) = 0$ add a self loop of weight n at y .
- If $\psi(y) = 1$ add a self loop of weight n^d at y .

In the YES case, if ψ is unsatisfiable, the Markov chain is just the random walk on the hypercube with probability $1/2$ of self loop at each vertex and it is well known that

$$\tau(1/4 - \delta) \leq C_\delta n \log n$$

where C_δ is a constant depending on $(1/4 - \delta)^{-1}$ polynomially.

In the NO case, where ψ is satisfiable, we will lower bound the time to couple from a satisfying state y and the state \bar{y} , obtained by flipping all the bits of y . Consider the distributions $X(t), Y(t)$ of the chain which are started at y and at \bar{y} . We can bound the variation distance after t steps as follows:

$$d(t) \geq 1 - P[\exists s \leq t \text{ s.t. } X(s) \neq y] - P[\exists s \leq t \text{ s.t. } Y(s) = y]$$

In each step, the chain started at y has chance at most $1/(n^{d-1} + 1)$ of leaving. On the other hand, the probability that the walk started from \bar{y} hits y in time t is exponentially small. Therefore

$$d(t) \geq 1 - 2t/(n^{d-1} + 1)$$

which implies that

$$\tau(1/4 + \delta) > \frac{1}{2}n^{d-1}(3/4 - \delta).$$

Let d be large enough (possibly depending polynomially on δ^{-1}) so that

$$\frac{1}{2}n^{d-1}(3/4 - \delta) > cC_\delta n \log n. \tag{7}$$

On the other hand we can show a polynomial upper bound on the mixing time by bounding the conductance as follows. Let M' be the Markov chain which is the random walk on the hypercube with self loop probabilities of $1/2$ (where the edge weights are as in the case where ψ is unsatisfiable). We bound the conductance of M by showing it is not too much smaller than the conductance of M' . We use the fact that for any vertex x , the weighted degree $d_x \leq (n^{d-1} + 1)d'_x$. Let $A \subseteq V(H)$.

$$\Phi_A(M) = \frac{\sum_{x \in A, y \in A^c} w_{xy}}{\sum_{A \in \Omega} d_x} = \frac{\sum_{x \in A, y \in A^c} w'_{xy}}{\sum_{A \in \Omega} d_x} \geq \frac{\sum_{x \in A, y \in A^c} w'_{xy}}{(n^{d-1} + 1) \sum_{A \in \Omega} d'_x} \geq \frac{\Phi_A(M')}{n^{d-1} + 1} \geq \frac{1}{n^d + n}$$

where we are assuming the lower bound on the conductance of the hypercube is $\frac{1}{n}$. We can lower bound π_{min} by $1/(n2^{n-1} + n^{d2^n})$ and hence we have for large enough n ,

$$\log(\pi_{min})^{-1} \leq 2n$$

and hence by Theorem 5 $\tau(1/4) \leq 32n^{2d+1}$. The reduction can be completed by setting $x = 0^n$, the vector of all 0's, $t_{max} = 32n^{2d+1}$ and $t = C_\delta n \log n$. By 7, we see that $ct < 1/2\sqrt{t_{max}/n^3}(3/4 - \delta)$ as required. \square

5 Discussion and Future Directions

We have shown that diagnosing convergence for a Markov chain is a hard computational problem, even when the chain is known to mix reasonably fast. However, we make no other assumptions about the Markov chain while in practice, one may know more.

After reading a draft of our paper the following question was raised by Dawn Woodard: Do our hardness results hold if the stationary distribution is known upto a normalizing constant as is often the case in practice? In the setting of Theorem 1 the hardness is not clear though we expect it may hold for a weaker complexity class. In the settings of Theorems 2 and 3 the hardness results do hold. In fact it follows immediately from the proofs that the hardness is true even in the case that the stationary distribution is known exactly.

It would be interesting to extend the classification for convergence diagnostics to specialized sampling algorithms. For example, Markov random fields are used to model high dimensional distributions in applications such as image processing. Gibbs sampling is a popular Markov Chain Monte Carlo algorithm for sampling from such distributions and standard diagnostics are used for testing convergence. What is the computational complexity of detecting convergence for Gibbs samplers?

Acknowledgments. The authors would like to thank Salil Vadhan for helpful discussions, and Dawn Woodard for kindly allowing us to include her question in Section 5. We would further like to acknowledge helpful comments and suggestions by anonymous referees.

References

1. Aldous, D., Fill, J.: Reversible Markov chains and random walks on graphs. Draft, <http://www.stat.Berkeley.edu/users/aldous>
2. Asmussen, S., Glynn, P.W., Thorisson, H.: Stationarity detection in the initial transient problem. *ACM Transactions on Modeling and Computer Simulation* 2(2), 130–157 (1992)
3. Bhatnagar, N., Bogdanov, A., Mossel, E.: The complexity of estimating MCMC convergence time (2010), <http://arxiv.org/abs/1007.0089>
4. BOA, Bayesian Output Analysis, <http://www.public-health.uiowa.edu/BOA>
5. Brooks, S., Roberts, G.: Assessing convergence of Markov Chain Monte Carlo algorithms. *Statistics and Computing* 8, 319–335 (1998)
6. Carlin, B., Louis, T.: *Bayes and Empirical Bayes methods for data analysis*. Chapman and Hall, Boca Raton (2000)
7. Cowles, M., Carlin, B.: Markov Chain Monte Carlo Convergence Diagnostics: A Comparative Review. *J. Am. Stat. Assoc.* 91(434), 883–904 (1996)
8. Gilks, W., Richardson, S., Spiegelhalter, D. (eds.): *Monte Carlo Statistical Methods*. Chapman and Hall, Boca Raton (1995)
9. Derrida, B., Weisbuch, G.: Dynamical phase transitions in 3-dimensional spin glasses. *Europhys. Lett.* 4(6), 657–662 (1987)
10. Goldwasser, S., Sipser, M.: Private Coins versus Public Coins in Interactive Proof Systems. In: Micali, S. (ed.) *Advances in Computing Research: a Research Annual, Randomness and Computation*, vol. 5, pp. 73–90 (1989)
11. Holenstein, T., Renner, R.S.: One-way secret-key agreement and applications to circuit polarization and immunization of public-key encryption. In: Shoup, V. (ed.) *CRYPTO 2005*. LNCS, vol. 3621, pp. 478–493. Springer, Heidelberg (2005)
12. Ntzoufras, I.: *Bayesian Modeling Using WinBUGS*. Wiley, Chichester (2009)
13. Jerrum, M.: *Counting, Sampling and Integrating: Algorithms and Complexity*, Birkhäuser, Basel (2003)
14. Jerrum, M., Sinclair, A.: Polynomial-time Approximation Algorithms for the Ising Model. *SIAM Journal on Computing* 22, 1087–1116 (1993)
15. Jerrum, M., Sinclair, A., Vigoda, E.: A Polynomial-time Approximation Algorithm for the Permanent of a Matrix with Non-negative Entries. *Journal of the ACM* 51(4), 671–697 (2004)
16. Levin, D., Peres, Y., Wilmer, E.: *Markov Chains and Mixing Times* (2008)
17. Lovász, L., Vempala, S.: Simulated Annealing in Convex Bodies and an $O^*(n^4)$ Volume Algorithm. In: *Proc. of the 44th IEEE Symposium on Foundations of Computer Science* (2003)
18. Lovász, L., Vempala, S.: Fast Algorithms for Logconcave Functions: Sampling, Rounding, Integration and Optimization. In: *Proc. of the 47th IEEE Symposium on Foundations of Computer Science* (2006)
19. Roberts, C., Casella, G.: *Monte Carlo Statistical Methods*. Springer, Heidelberg (2004)
20. Plummer, M., Best, N., Cowles, K., Vines, K.: CODA: Convergence Diagnosis and Output Analysis for MCMC. *R News* 6(1), 7–11 (2006), <http://CRAN.R-project.org/doc/Rnews/>
21. Sinclair, A.: *Algorithms for Random Generation and Counting*. Birkhauser, Basel (1993)
22. Sahai, A., Vadhan, S.: A complete promise problem for statistical zero-knowledge. In: *Proceedings of the 38th Annual Symposium on the Foundations of Computer Science*, pp. 448–457 (1997)
23. Saks, M.: Randomization and Derandomization in Space-bounded Computation. In: *Proceedings of the 11th Annual IEEE Conference on Computational Complexity*, pp. 128–149 (1996)
24. Savitch, W.J.: Relationships Between Nondeterministic and Deterministic Space Complexities. *J. Comp. and Syst. Sci.* 4(2), 177–192 (1970)

Streaming Algorithms with One-Sided Estimation

Joshua Brody^{1,*} and David P. Woodruff²

¹ IIS, Tsinghua University
joshua.e.brody@gmail.com

² IBM Research-Almaden
dpwoodru@us.ibm.com

Abstract. We study the space complexity of randomized streaming algorithms that provide one-sided approximation guarantees; e.g., the algorithm always returns an overestimate of the function being computed, and with high probability, the estimate is not too far from the true answer. We also study algorithms which always provide underestimates.

We also give lower bounds for several one-sided estimators that match the deterministic space complexity, thus showing that to get a space-efficient solution, two-sided approximations are sometimes necessary. For some of these problems, including estimating the longest increasing sequence in a stream, and estimating the Earth Mover Distance, these are the first lower bounds for randomized algorithms of any kind.

We show that for several problems, including estimating the radius of the Minimum Enclosing Ball (MEB), one-sided estimation is possible. We provide a natural function for which the space for one-sided estimation is asymptotically less than the space required for deterministic algorithms, but more than what is required for general randomized algorithms.

What if an algorithm has a one-sided approximation from both sides? In this case, we show the problem has what we call a Las Vegas streaming algorithm. We show that even for two-pass algorithms, a quadratic improvement in space is possible and give a natural problem, counting non-isolated vertices in a graph, which achieves this separation.

1 Introduction

Computing on data streams is of growing interest in many areas of computer science, such as databases, networks, and algorithm design. Here it is assumed that the algorithm sees updates to elements of an underlying object one by one in an arbitrary order, and needs to output certain statistics of the input. Therefore

* Supported in part by the National Basic Research Program of China Grant 2007CB807900, 2007CB807901, and the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174. Also supported in part from the Danish National Research Foundation and the National Science Foundation of China (under the grant 61061130540) for the Sino-Danish Center for the Theory of Interactive Computation, within which part this work was performed.

it must maintain a short summary or sketch of what it has seen. We refer the reader to the survey by Muthukrishnan [17] for a list of applications.

In this paper, we consider the space complexity of streaming algorithms which return estimates with *one-sided approximation*—either the streaming algorithm always returns an overestimate, or it always returns an underestimate. As with the case of standard randomized streaming algorithms, we want the algorithm to return an accurate estimate with high probability. While one-sided approximation has been extensively studied in the property testing literature, it has not been considered as an object of study for streaming algorithms.

Definition 1.1. *An ε -overestimator for f is a randomized algorithm that, given a stream σ returns $\hat{f}(\sigma)$ such that*

- $\hat{f}(\sigma) \geq f(\sigma)$.
- *With probability at least $2/3$, $\hat{f}(\sigma) \leq f(\sigma)(1 + \varepsilon)$.*

An ε -underestimator for f is a randomized algorithm that returns an underestimate $\hat{f}(\sigma)$ such that with probability at least $2/3$, we have $\hat{f}(\sigma) \geq f(\sigma)(1 - \varepsilon)$.

An important class of one-sided approximations are problems where the *information lost* by using a small amount of space is one-sided. Perhaps the best known example in this class is the COUNT-MIN sketch [5], which is used to maintain approximate frequency counts and can produce accurate estimations of ϕ -quantiles or ϕ -heavy hitters. The COUNT-MIN sketch essentially works by maintaining a random hash table h of counters and updating the counter in bucket $h(i)$ each time item i is seen on the stream. The counter in bucket $h(i)$ then provides an *overestimate* of the true frequency of item i , since collisions can only increase the count. By maintaining several hash tables h_1, h_2, \dots, h_t and returning the minimum $h_j(i)$ over all j , the COUNT-MIN sketch gets an overestimate of the frequency of item i that with high probability remains close to the true frequency. Since its inception, the COUNT-MIN sketch has also been used as a subroutine in several other applications.

Surprisingly, the COUNT-MIN sketch is also used to generate ε -underestimators. In the k -median problem, the input is a set of points P on a discrete grid $[\Delta]^d$, and the goal is to output a set of k points Q that minimizes $C(Q, P) := \sum_{p \in P} \min_{q \in Q} \|p - q\|$. Such a set is called a k -median. Indyk [12] uses a COUNT-MIN sketch to *underestimate* $C(P, Q)$.

We are interested in the space complexity of one-sided approximations and how this space complexity relates to the complexity of randomized and deterministic algorithms that give two-sided approximations. We also study what happens when both underestimates and overestimates are possible. By properly scaling the one-sided estimates, we can get an algorithm that provides a $(1 \pm \varepsilon)$ -approximation with high probability, and **knows** when its estimate is a poor approximation. We call such algorithms Las Vegas algorithms.

Definition 1.2. *A Las Vegas algorithm for f is a randomized streaming algorithm that, given a stream σ either returns $\hat{f}(\sigma)$ such that $|\hat{f}(\sigma) - f(\sigma)| \leq \varepsilon f(\sigma)$ or outputs FAIL. The algorithm FAILS with probability at most $1/3$.*

Remark 1.1. Las Vegas algorithms can alternatively be thought of as multipass algorithms that never fail; instead, they repeat until accepting an estimate. That notion corresponds more with the concept of Las Vegas algorithms used in communication complexity. Our definition has meaning even for one-pass algorithms.

1.1 Our Problems

We consider the space complexity of streaming algorithms under several models: two-sided $(1 \pm \varepsilon)$ -approximations, ε -overestimates, ε -underestimates, Las Vegas algorithms, and deterministic algorithms. Let $S_{1\pm\varepsilon}(f)$, $S_{\varepsilon\text{-under}}(f)$, and $S_{\varepsilon\text{-over}}(f)$ denote the space complexity of two-sided estimators, ε -underestimators, and ε -overestimators. $S_{LV}(f)$ and $S_{det}(f)$ denote the space complexity of Las Vegas and deterministic algorithms that compute f exactly; $S_{\varepsilon,LV}(f)$ and $S_{\varepsilon,det}(f)$ are the complexity of Las Vegas and deterministic algorithms that return $(1 \pm \varepsilon)$ -approximations. The relationship between these measures is captured in the following lemma, which we prove in Section 3.

Lemma 1.1. *For any f , the space complexities are characterized (up to small changes in ε) by the following:*

$$\begin{aligned} S_{1\pm\varepsilon}(f) &\leq \min\{S_{\varepsilon\text{-under}}(f), S_{\varepsilon\text{-over}}(f)\} \\ &\leq \max\{S_{\varepsilon\text{-under}}(f), S_{\varepsilon\text{-over}}(f)\} = \Theta(S_{\varepsilon,LV}(f)) \leq S_{\varepsilon,det}(f) . \end{aligned}$$

Our next collection of results provides strict separations for these inequalities.

Cascaded Norms. In Section 3, we consider the problem of estimating the cascaded norm $\ell_0(Q)(A)$ in a stream of updates to an $n \times n$ matrix A . Here, $\ell_0(Q)(A)$ is the number of non-zero rows of A . We show that two-sided approximations are possible in $\text{poly}(\log(n)/\varepsilon)$ space; an ε -overestimate is possible in $\tilde{O}(n)$ space, and $\Omega(n^2)$ space is required for deterministic algorithms.

Theorem 1.1. *For the problem of estimating $\ell_0(Q)(A)$ in the streaming model, the following bounds hold: (i) $S_{1\pm\varepsilon}(\ell_0(Q)) = \tilde{O}(1)$, (ii) $S_{\varepsilon\text{-over}}(\ell_0(Q)) = \tilde{\Theta}(n)$, and $S_{\varepsilon,det}(\ell_0(Q)) = \Omega(n^2)$.*

This problem also corresponds to estimating the number of non-isolated vertices in a graph [8] and can be useful for counting outliers in social networks.

Earth Mover Distance. In this problem, the elements on the stream define two point sets $A, B \subseteq [\Delta^2]$, and the algorithm should estimate the cost of the best matching between A and B . In Section 3, we show

Theorem 1.2. *For all constant ε , $S_{\varepsilon\text{-under}}(EMD) = S_{\varepsilon\text{-over}}(EMD) = \Omega(\Delta^2)$. Moreover, these bounds hold even for underestimators that return a value that is at least $1/c \cdot EMD$ or overestimators that return a value that is at most $c \cdot EMD$ with constant probability, for any constant $c > 1$.*

This is the first lower bound for EMD for any class of randomized algorithms. A result of Andoni et al. [1] gives a c -approximation in $\Delta^{O(1/c)}$ space for any $c > 1$, and so this separates the complexity of one-sided and two-sided estimations.

¹ The $\tilde{O}(\cdot)$ notation hides terms polynomial in $\log n$ and ε .

List Equality. To separate the deterministic and Las Vegas space complexities, we adapt a problem of Mehlhorn and Schmidt [15] to the streaming setting. The problem is called LIST-EQUALITY. The inputs are two lists of n -bit numbers $X, Y \in (\{0, 1\}^n)^n$, and the goal is to compute $\text{ListEQ}(X, Y) := \bigvee_{i=1}^n \text{EQ}(X_i, Y_i)$. Mehlhorn and Schmidt [15] introduced this problem and use it to show a quadratic separation between the deterministic and Las Vegas versions of communication complexity. In the streaming version of this problem, X, Y appear sequentially on a stream of n^2 bits. We give a $\tilde{O}(n)$ space Las Vegas algorithm; an $\Omega(n^2)$ bound follows from [15].

Theorem 1.3. *For the LIST-EQUALITY problem in the streaming model, we have $S_{LV}(\text{ListEQ}) = \tilde{O}(n)$, while $S_{det}(\text{ListEQ}) = \Omega(n^2)$.*

In addition to the space complexity separations, in Section 4 we give new one-sided estimators for two problems motivated by machine learning: the Minimum Enclosing Ball and Classification problems. These problems were studied in the streaming setting by Clarkson et al. [4] who gave efficient two-sided estimates for both problems. We extend their work to give one-sided estimates.

In Section 5, we give lower bounds for one-sided estimates for a large range of problems, including estimating the length of the longest increasing subsequence (LIS), the ℓ_p -norms and ℓ_p -heavy hitters, and the empirical entropy of a stream.

We also discuss open questions in Section 5.

2 Preliminaries

For many of the problems we consider, the stream is a sequence of m tokens $(i_1, v_1), \dots, (i_m, v_m) \in [n] \times \{-M, \dots, M\}$ interpreted as updates to a frequency vector $z \in \mathbb{N}^n$, where a token (i, v) causes $z_i \leftarrow z_i + v$. In these problems we implicitly associate the frequency vector z with the corresponding stream σ . In an *insertion-only* stream, v_i is always positive. In the *strict turnstile* model, the current value of z_i is always positive, though some of the v_i may be negative. The *general turnstile* model allows arbitrary z_i .

Given $z \in \mathbb{R}^m$, the ℓ_p -norm of z is defined as $\|z\|_p := (\sum_{i=1}^m |z_i|^p)^{1/p}$. The p th frequency moment is $F_p(z) := \|z\|_p^p = \sum_{i=1}^m |z_i|^p$. We use $\delta(x, y)$ to denote the Hamming distance between strings x and y , that is, the number of positions that differ in x and y .

In rest of this section, we briefly describe the basic terminology and notation we need for communication complexity, as well as the problems we use to prove our streaming lower bounds. For a more complete treatment, we refer the reader to the standard text by Kushilevitz and Nisan [14].

Given a boolean function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \{0, 1\}$, let $R_\varepsilon(f)$ denote the minimum communication cost of a public-coin randomized protocol P such that on all inputs, $P(x, y) = f(x, y)$ with probability at least $1 - \varepsilon$. We are particularly interested in the communication complexity of protocols with one-sided error. For $b \in \{0, 1\}$, let $R_\varepsilon^b(f)$ be the cost of the best randomized protocol P for f such that (i) when $f(x, y) \neq b$, P correctly computes $f(x, y)$, and (ii) when $f(x, y) = b$,

P computes $f(x, y)$ with probability $\geq 1 - \varepsilon$. We usually take $\varepsilon := 1/3$; in this case, we drop the subscript.

Next we describe two problems we use extensively to show our bounds. In the EQUALITY problem, Alice and Bob receive n -bit strings x, y and wish to compute $\text{EQ}(x, y) = 1$ iff $x = y$. The standard EQ test gives $R_\varepsilon^0(\text{EQ}) = O(\log(1/\varepsilon))$; in contrast, we have $R^1(\text{EQ}) = \Omega(n)$. In essence, protocols which must be correct when $x \neq y$ are as hard as the deterministic case. When making reductions in this case, we'll often describe the problem as NEQ to emphasize that the protocol must be correct on $x \neq y$ instances.

Our second problem is the promise problem $\text{GAP-EQ}_{n,t}$. Here, Alice and Bob receive n -bit strings under the promise that either $x = y$ or $\delta(x, y) = t$ and output 1 iff $x = y$. Using a combinatorial result of Frankl and Rödl [9], Buhrman et al. [3] proved that $R^1(\text{GAP-EQ}_{n,t}) = \Omega(n)$ for all $t = \Theta(n)$ and used it to get separations between classical and quantum communication complexity. We suppress the subscripts when n is clear from context and $t = n/2$.

3 Space Complexity Separations

In this section, we develop separations between the space complexities for different classes of streaming algorithms.

Lemma 3.1 (Restatement of Lemma 1.1). *For any f , the space complexities are characterized (up to small changes in ε) by the following inequality*

$$\begin{aligned} S_{1\pm\varepsilon}(f) &\leq \min\{S_{\varepsilon\text{-under}}(f), S_{\varepsilon\text{-over}}(f)\} \\ &\leq \max\{S_{\varepsilon\text{-under}}(f), S_{\varepsilon\text{-over}}(f)\} = \Theta(S_{\varepsilon, LV}(f)) \leq S_{\varepsilon, \text{det}}(f) . \end{aligned}$$

Proof. The inequalities are trivial inclusions. To prove the equality, fix an ε -underestimator \mathcal{A}_U and an ε -overestimator \mathcal{A}_O , and create a Las Vegas algorithm in the following way: Run \mathcal{A}_U and \mathcal{A}_O in parallel, scale the underestimator by $(1 + \varepsilon)$ and the overestimator by $(1 - \varepsilon)$, and FAIL if the scaled underestimate remains less than the scaled overestimate. If the algorithm accepts, return the geometric mean of the estimates. This algorithm accepts with high probability, since it accepts whenever both estimators return good ranges. Furthermore, it's easy to show that when it accepts, the algorithm returns a $(1 \pm \varepsilon)$ -approximation.

We provide strict separations for each of the inequalities in Lemma 3.1. Our first separation result is for the problem of estimating Cascaded Norms.

Many streaming papers have focused on *single-attribute* aggregation, such as norm estimation. Most applications however deal with multi-dimensional data where the real insights are obtained by slicing the data several times and applying several aggregations in a cascaded fashion. A *cascaded aggregate* $P \circ Q$ of a matrix is defined by evaluating aggregate Q repeatedly over each row of the matrix, and then evaluating aggregate P over results obtained from each row. A well-studied aggregate is the so-called cascaded norm problem on numerical data, for which we first compute the Q norm of each row, then the P norm of the vector of values

obtained, for arbitrary norms P and Q . These were introduced by Cormode and Muthukrishnan [6], and studied in several followup works [16, 2, 13, 11], with particular attention to the case when $P = \ell_p$ and $Q = \ell_q$. In the streaming model, the underlying matrix is initialized to 0, and receives multiple updates in the form of increments and decrements to its entries in an arbitrary order.

One special case of this problem is $\ell_0(Q)$, which corresponds to the number of non-zero rows in an $n \times d$ matrix A . This problem was studied in [13], where the authors obtained a $\text{poly}(\log(nd)/\varepsilon)$ space randomized algorithm for $(1 \pm \varepsilon)$ -approximation. This measure is important since it corresponds to estimating the number of non-isolated vertices in a graph. This follows by taking $d = n$ and viewing the matrix A as the adjacency matrix of a graph. Its complement, $n - \ell_0(Q)$, is the number of isolated vertices and may be useful for counting outliers in social networks. This was studied in a sampling (a special case of streaming) context in, e.g., [8].

The following theorem characterizes the space complexity of the different estimators for $\ell_0(Q)$.

Theorem 3.1 (Restatement of Theorem 1.1). *The problem of estimating the cascaded norm $\ell_0(Q)$ in the general turnstile model has the following space complexities:*

1. *There exists a $(1 \pm \varepsilon)$ -approximation that uses $O(\text{poly}(\log(nd)/\varepsilon))$ space.*
2. *There is an ε -underestimator for $\ell_0(Q)$ that uses $O(n \text{poly}(\log(nd)/\varepsilon))$ space.*
3. *Any ε -underestimator for $\ell_0(Q)$ requires $\Omega(n)$ space.*
4. *Any ε -overestimator for $\ell_0(Q)$ requires $\Omega(nd)$ space.*
5. *Any deterministic approximation for $\ell_0(Q)$ requires $\Omega(nd)$ space.*

Proof. The upper bound for $(1 \pm \varepsilon)$ -approximation comes from Jayram and Woodruff [13]. To get an upper bound for ε -underestimators, let u be the maximal possible element in the matrix. We assume that u is polynomially related to n, d and the length of the stream. Next, choose a prime $q = \Theta(und)$, and let V be the $q \times d$ Vandermonde matrix, where the i th row $V_i = (1, i, i^2, \dots, i^{d-1}) \in GF(q)^d$. It's well known that any d rows of V are linearly independent. It follows that for any nonzero A_i , at most $d - 1$ rows v of V have $\langle A_i, v \rangle = 0 \pmod{q}$.

We estimate $\ell_0(Q)(A)$ by picking a random row of V , computing the inner product $\langle A_i, v \rangle$ in $GF(q)$ for each row i of A , and returning the number of rows that give $\langle A_i, v \rangle = 0$. It's easy to see that each inner product can be maintained in $O(\log(u))$ space. Furthermore, we always underestimate the number of nonzero rows, and for a random v , $\langle A_i, v \rangle = 0$ with probability at most $O(q/d) = O(1/n)$. By the union bound, our choice of v identifies *all* nonzero rows with probability $9/10$. Therefore, we always underestimate the number of rows with nonzero entries, and with high probability we compute $\ell_0(Q)(A)$ exactly. The space required is $O(\log(nd))$ to store a pointer to v , and $n \log u$ to maintain $\langle A_i, v \rangle$ for each row i .

On the other hand, a reduction from GAP-EQ gives an $\Omega(n)$ lower bound for ε -underestimators. Specifically, fix $d := 1$, and given n -bit strings x, y , Alice converts each bit x_i of her input into a token $(i, 1, 1 - x_i)$. Bob converts each

bit of y_i of his input into a token $(i, 1, -y_i)$. They then simulate the algorithm for underestimating $\ell_0(Q)$ on the resulting matrix A and output NO when the estimate is at most $n/2$. Note that $\ell_0(Q)(A) = n$ when $x = y$ and $\ell_0(Q)(A) = n/2$ when $\delta(x, y) = n/2$, hence an ε -underestimator for $\ell_0(Q)$ always produces a correct answer for $\delta(x, y) = n/2$, and with high probability produces a correct answer for the $x = y$ case.

For ε -overestimators, a stronger lower bound is possible, via reduction from NEQ on strings of length nd . Each coordinate in the string maps to an entry in the matrix. Alice maps each $x_{i,j} \rightarrow (i, j, x_{i,j})$, and Bob maps each $y_{i,j} \rightarrow (i, j, -y_{i,j})$. Thus, $\ell_0(Q)(A) > 0$ iff $x \neq y$. Alice and Bob then compute NEQ(x, y) by simulating an ε -overestimator for $\ell_0(Q)(A)$ and outputting $x \neq y$ whenever it returns a positive value. This also implies the $\Omega(nd)$ deterministic bound.

Next, we prove lower bounds for estimating Earth Mover Distance. The Earth Mover Distance between multisets $A, B \subseteq [\Delta]^2$ is the cost of the best matching between A and B . Formally, we define

$$\text{EMD}(A, B) = \min_{\pi: A \rightarrow B} \sum_{a \in A} \|a - \pi(a)\| .$$

Andoni et al. [1] gave a 1-pass, $\Delta^{O(1/c)}$ -space algorithm that returns \widehat{EMD} such that $\text{EMD}(A, B)/c \leq \widehat{EMD}(A, B) \leq c\text{EMD}(A, B)$. In general, this approximation factor c can be much greater than 1; for this reason, we refer to results in this section as c -approximations instead of ε -approximations.

Proof (of Theorem 1.2). Partition $[\Delta]^2$ into $n := \Delta^2/2$ pairs of adjacent points $\{(p_{i,0}, p_{i,1}) : 1 \leq i \leq n\}$. The nature of this construction is immaterial; we only require that the pairs of points are adjacent.

To get the lower bound for c -overestimators, we reduce from NEQ. Given $x, y \in \{0, 1\}^n$, Alice creates a set of points $A := \{a_1, \dots, a_n\}$ by mapping each coordinate $x_i \rightarrow p_{i,x_i} =: a_i$. Bob similarly creates $B := \{b_1, \dots, b_n\}$ by mapping $y_i \rightarrow p_{i,y_i} =: b_i$. Then, Alice and Bob simulate a c -overestimating algorithm for EMD and output $x \neq y$ if $\widehat{EMD}(A, B) > 0$.

Note that if $x \neq y$ then clearly $EMD(A, B) > 0$, and since the streaming algorithm returns an overestimate, Alice and Bob will always correctly compute $x \neq y$. Furthermore, when $x = y$, then $EMD(A, B) = 0$; hence, the overestimator will output $\widehat{EMD}(A, B) \leq c\text{EMD}(A, B) = 0$ with high probability. In this way, a c -overestimator for EMD gives a protocol for NEQ. Since $R^1(\text{NEQ}) = \Omega(n) = \Omega(\Delta^2)$, the lower bound for c -overestimators follows.

To get a lower bound for c -underestimators, set $\gamma := 1 - 1/2c$, and reduce from GAP-EQ $_{n,\gamma n}$. As in the lower bound for overestimators, Alice and Bob map their inputs x, y to pointsets A, B . This time, Alice again sets $a_i := p_{i,x_i}$, but Bob creates $b_i := p_{i,1-y_i}$. Then they simulate a c -underestimator for EMD and output $\delta(x, y) = \gamma n$ if $\widehat{EMD}(A, B) \leq n(1 - \gamma)$.

Essentially, Alice and Bob solve GAP-EQ by using the EMD algorithm to estimate $\delta(x, -y)$. Note that

$$\text{EMD}(A, B) = \begin{cases} n & \text{if } x = y , \\ n(1 - \gamma) & \text{if } \delta(x, y) = \gamma n . \end{cases}$$

Since $\text{EMD}(A, B) = n(1 - \gamma)$ when $\delta(x, y) = \gamma n$, a c -underestimator always returns a correct value for $\delta(x, y) = \gamma n$. When $x = y$, the c -underestimator returns $\widehat{\text{EMD}}(A, B) \geq n/c > n(1 - \gamma)$ with high probability. Hence, the $\Omega(\Delta^2)$ lower bound follows from the $\Omega(n)$ lower bound on GAP-EQ $_{n, \gamma n}$.

We end this section with a two-pass Las Vegas algorithm for LIST-EQUALITY.

Proof (of Theorem 7.3). We convert the two player communication protocol of Mehlhorn and Schmidt [15] to work in a Las Vegas environment. In the first pass, the algorithm uses an r -bit EQUALITY test to compare X_i and Y_i for each i . Let I be the set of indices i that pass this test. If I is empty, then the algorithm outputs $\text{ListEQ}(X, Y) = 0$. Otherwise, if $|I| > m$, let I' be a random m -subset of I . In the second pass, the algorithm saves X_i for each $i \in I'$ and compares X_i, Y_i directly. If it finds any i such that $X_i = Y_i$, then the algorithm outputs $\text{ListEQ}(X, Y) = 1$. Otherwise, the algorithm outputs FAIL.

This algorithm uses nr space to maintain the n equality tests, nm space to store X_i for up to m indices $i \in I'$, and $O(n)$ other space for bookkeeping. Therefore, it uses $O(n(r+m))$ bits total. As for correctness, the algorithm never outputs incorrectly, since the EQUALITY test is one-sided in the first pass, and the test in the second pass has zero error. By a union bound, the chance that the algorithm does *not* terminate after the first pass when $\text{ListEQ}(X, Y) = 0$ is at most $n2^{-r}$. When $\text{ListEQ}(X, Y) = 1$, the algorithm fails to terminate only when $X_i \neq Y_i$ for all $i \in I'$. This only happens when at least m EQUALITY tests fail in the first pass, which happens with probability (much less than) $n^m 2^{-rm} = 2^{m \log n - rm}$. Taking $m = r = 2 \log n$ gives a two-pass, $O(n \log n)$ space Las Vegas algorithm for ListEQ.

4 Upper Bounds

In this section, we present new one-sided estimators for two problems motivated by machine learning from the recent work of Clarkson et al. [4]

Minimum Enclosing Ball. In the Minimum Enclosing Ball (MEB) problem, the input is a matrix $A \in \{-M, -M + 1, \dots, M\}^{nd}$, for some $M = \text{poly}(nd)$, whose rows are treated as points in d -dimensional space. The goal is to estimate the radius of the minimum enclosing ball of these points; i.e., to estimate $\min_{y \in \mathbb{R}^d} \max_{1 \leq i \leq n} \|A_i - y\|$.

In the streaming version of this problem, we assume that we see the rows of A one at a time (and exactly once), but in an arbitrary order. An algorithm from [4] runs in $\tilde{O}(1/\varepsilon^2)$ space and uses $\tilde{O}(1/\varepsilon)$ passes and returns $1/\varepsilon$ indices

$i_1, \dots, i_{1/\varepsilon}$ such that with probability at least $1/2$, the ball centered around these indices that encloses all points has radius close to the smallest possible. In other words, the point $y := \sum_{j=1}^{1/\varepsilon} \varepsilon A_{i_j}$ is the center of a ball whose radius $r := \max_{1 \leq i \leq n} \|A_i - y\|$ is an ε -overestimate of the radius of the MEB. It is easy to see that y can be computed with one more pass and $O(d \log M)$ more bits of space. Given y , the radius of the ball centered at y can be computed in an extra pass using $O(\log M)$ additional space by maintaining the maximum distance of a point from y . This radius is thus an ε -overestimator for MEB. One can reduce the failure probability from $1/2$ by repeating this process independently and in parallel several times and taking the minimum radius found.

Theorem 4.1. *There is an ε -overestimator for Minimum Enclosing Ball that uses $O(d \log(nd) + \text{polylog}(nd/\varepsilon)/\varepsilon^2)$ space and $O(\text{polylog}(nd/\varepsilon)/\varepsilon)$ passes.*

Classification. As with the previous problem, the input is a set of n points $A_1, \dots, A_n \in \{-M, -M+1, \dots, M\}^d$ (Points A_i are assumed to have $\|A_i\| \leq 1$.) Given $x \in \mathbb{R}^d$, define $\sigma_x := \min_i \langle A_i, x \rangle$. In the classification problem, the goal is to output the margin $\sigma := \min_{x: \|x\| \leq 1} \sigma_x$. Another algorithm from [4] runs in $\tilde{O}(1/\varepsilon^2)$ space and $\tilde{O}(1/\varepsilon^2)$ passes and returns a set of $t = O(1/\varepsilon^2)$ indices i_1, \dots, i_t such that with constant probability, a certain linear combination y of $\{A_{i_j}\}_{j=1}^t$ gives an additive ε -approximation to the margin. As in the case of MEB, y can be computed in $O(d \log M)$ additional bits of space, from which σ_y can be computed exactly, which is an ε -underestimator for the margin.

Theorem 4.2. *There is an $O(d \log(nd) + \text{polylog}(nd/\varepsilon)/\varepsilon^2)$ space, $O(\text{polylog}(nd/\varepsilon)/\varepsilon^2)$ -pass algorithm that computes y such that $\sigma \geq \sigma_y \geq \sigma - \varepsilon$.*

5 Lower Bounds

In the LONGEST-INCREASING-SUBSEQUENCE problem, the input is a stream of n tokens $\sigma \in [m]^n$, and the goal is to estimate the length of the longest increasing sequence of σ , which we denote $\text{lis}(\sigma)$. Gopalan et al. [11] gave an $O(\sqrt{n/\varepsilon})$ deterministic algorithm for estimating $\text{lis}(\sigma)$; this space complexity was later proven tight by Gál and Gopalan [10] and Ergun and Jowhari [7].

The proof of Gál and Gopalan uses a reduction from the Hidden-Increasing-Subsequence ($\text{HIS}_{\ell,t,k}$) problem. $\text{HIS}_{\ell,t,k}$ is a t -player communication problem where PLR_i is given the i th row of a matrix $M \in [m]^{t\ell}$, with the promise that either (i) all columns are decreasing, or (ii) there exists a column with an increasing subsequence of length k . The players wish to distinguish these cases.

Gál and Gopalan proved a lower bound on the maximum communication complexity of deterministic, one-way protocols for $\text{HIS}_{\ell,t,k}$. We need similar lower bounds for randomized protocols that make no mistakes when there exists a hidden increasing sequence. Let $R^{\text{MAX},0}(\text{HIS}_{\ell,t,k})$ denote the maximum communication complexity (i.e., the size of the largest message) of the best randomized, one-way protocol for $\text{HIS}_{\ell,t,k}$ that errs only when all columns of M are decreasing. We observe that the deterministic lower bound technique of Gál and Gopalan generalizes to $R^{\text{MAX},0}(\text{HIS}_{\ell,t,k})$.

Theorem 5.1. $R^{\text{MAX},0}(\text{HIS}_{\ell,t,k}) \geq \ell((1 - k/t) \log(m/(k - 1)) - H(k/t)) - \log t$. In particular, taking $n := \ell t, k := t/2 + 1$, and $\varepsilon := (k - 1)/\ell$, we have

$$R^{\text{MAX},0}(\text{HIS}_{\ell,t,k}) = \Omega(\sqrt{n/\varepsilon} \log(m/\varepsilon n)) = \tilde{\Omega}(\sqrt{n/\varepsilon}) .$$

Using the reduction from Gopalan et al [11], we get the following corollary.

Corollary 5.1. An ε -overestimator for LONGEST-INCREASING-SUBSEQUENCE requires $\Omega(\sqrt{n/\varepsilon})$ space.

In the rest of this section, we provide a suite of lower bounds for streaming statistics. Unless otherwise specified, the underlying vector $z \in [m]^n$ is initialized to zero, and tokens (i, v) represent updates $z \leftarrow z_i + v$. Our lower bounds cover the following problems.

- ℓ_p -**norm**: estimate $\|z\|_p := (\sum_{i=1}^n |z_i|^p)^{1/p}$.
- ℓ_p **heavy hitters**: For “heavy hitter thresholds” $\hat{\phi} < \phi$, return all i such that $|z_i|^p \geq \phi F_p(z)$ and no i such that $|z_i|^p \leq \hat{\phi} F_p(z)$.
- **empirical entropy**: estimate $H(z) = \sum_i (|z_i|/F_1(z)) \log(F_1(z)/|z_i|)$. (Recall that $F_1(z) := \sum_i |z_i|$ is the ℓ_1 -norm of the stream.)

All of these lower bounds come from reductions from NEQ or GAP-EQ. Alice and Bob convert strings x, y into streams σ_A, σ_B . The communication protocol works by simulating a streaming algorithm on $\sigma := \sigma_A \circ \sigma_B$ and estimating the resulting statistic. Because these lower bounds are similar and space is limited, we include only a few proofs and defer others to the full version of the paper.

Theorem 5.2. For all $p, S_{\varepsilon\text{-over}}(\ell_p\text{-norm}) = \Omega(n)$ in the general turnstile model.

Proof. This is a simple reduction from NEQ. We omit the details.

Theorem 5.3. For all $p \neq 1$, there exists $\varepsilon > 0$ such that $S_{\varepsilon\text{-under}}(\ell_p\text{-norm}) = S_{\varepsilon\text{-over}}(\ell_p\text{-norm}) = \Omega(n)$ in the insertion-only model.

Proof. We require different reductions for ε -overestimators and ε -underestimators and for when $p < 1$ and $p > 1$; however, in all cases, we reduce from GAP-EQ by embedding either (x, y) or $(x, -y)$ into the streaming problem. In all cases, choosing ε appropriately ensures that the relevant one-sided estimator gives a protocol for GAP-EQ with one-sided error. All four reductions are similar; we include a proof for the case where $p < 1$ and we want a lower bound for ε -overestimators and defer the other proofs to the full version.

Suppose that $p < 1$, and let \mathcal{A}_O be an ε -overestimator for the ℓ_p -norm, where $\varepsilon := \min\{1/3, (1 - 2^{p-1})/(2^{p+1}p)\}$. Given x , Alice creates a stream $\sigma_A = (a_1, \dots, a_n)$, where $a_i := (2i - x_i, 1)$. Bob converts y into a stream $\sigma_B := (b_1, \dots, b_n)$, where $b_i := (2i - y_i, 1)$. Note that

$$\|z\|_p = \begin{cases} 2n^{1/p} & \text{if } x = y , \\ 2n^{1/p} \left(\frac{1}{2} + 2^{-p}\right)^{1/p} & \text{if } \delta(x, y) = n/2 . \end{cases}$$

When $p < 1$, the ℓ_p -norm given by the $x = y$ case is less than the $\delta(x, y) = n/2$ case. Therefore, Alice and Bob can solve GAP-EQ by simulating \mathcal{A}_O and returning “ $\delta(x, y) = n/2$ ” if \mathcal{A}_O returns an estimate at least $2n^{1/p}(1/2+2^{-p})^{1/p}$. Since \mathcal{A}_O always provides an overestimate, the protocol always computes the $\delta(x, y) = n/2$ cases. Further, note that

$$(1 + \varepsilon)^p < (1 + 2\varepsilon p) \leq 1 + 2p \left((1 - 2^{p-1})/p2^{p+1} \right) \leq (1/2 + 2^{-p}) ,$$

where the first inequality uses $(1 + x)^r < 1 + 2xr$, which holds for $r > 0$ and $0 \leq x \leq 1/2$. Therefore, when $x = y$, \mathcal{A}_O (with high probability) returns an estimate at most $2n^{1/p}(1+\varepsilon) < 2n^{1/p}(1/2+2^{-p})^{1/p}$, hence the protocol computes “ $x = y$ ” correctly with high probability.

Note that this reduction fails for the case $p = 1$ because the gap in ℓ_p -norm in the YES and NO instances disappears. The ℓ_p -norm in this case corresponds to counting the net number of items in the stream. This can easily be exactly computed in $O(\log n)$ space.

Finally, we consider one-sided estimates for ℓ_p -heavy hitters. The notion of one-sidedness is slightly different here, since the algorithm is to output a set of items instead of an estimation. Here, we define the over- and under-estimation to refer to the set of items that are reported.

Definition 5.1. *A two-sided estimator for the $(\hat{\phi}, \phi, \ell_p)$ heavy hitters problem is a randomized algorithm that with probability $2/3$*

1. *returns all i such that $|z_i|^p \geq \phi F_p(z)$.*
2. *returns no i such that $|z_i|^p \leq \hat{\phi} F_p(z)$.*

An overestimator is an algorithm that achieves condition (1) with probability 1. An underestimator fulfills condition (2) with probability 1.

Theorem 5.4. *The following bounds hold for $(\hat{\phi}, \phi, \ell_p)$ -heavy hitters:*

- *For all $0 < \phi < 1$, $\hat{\phi} = \phi/2$, and $p \geq 0$, $\Omega(n)$ -space is required in the general turnstile model for both over- and underestimators.*
- *For all $0 < \phi < 1$ and $p \neq 1$, there exists $\hat{\phi}$ such that $\Omega(n)$ space is required in the insertion-only model for both over- and underestimators.*
- *$\Theta(\log n/\phi)$ space is required in the insertion-only model for all $(\phi/2, \phi, 1)$ heavy-hitters.*

Theorem 5.5. *For $\varepsilon = O(1/\log n)$, $\Omega(n)$ space is necessary to ε -overestimate or ε -underestimate the empirical entropy $H(z)$ in the insertion-only model.*

Open Questions: Our work leaves open several natural questions.

1. Can one characterize the functions f for which $S_{\varepsilon\text{-under}}(f) = S_{1\pm\varepsilon}(f)$ or $S_{\varepsilon\text{-over}}(f) = S_{1\pm\varepsilon}(f)$? A complete characterization may be hard, as it could be used to obtain bounds on $S_{1\pm\varepsilon}(\text{LONGEST-INCREASING-SUBSEQUENCE})$ and $S_{1\pm\varepsilon}(\text{EMD})$, two challenging questions in the data stream literature. Even a partial characterization would be interesting.
2. What results hold for estimators $\hat{f}(\sigma)$ for which $\hat{f}(\sigma) \geq f(\sigma)$ always, and with probability at least $2/3$, $\hat{f}(\sigma) \leq f(\sigma)(1 + \varepsilon) + \beta$?

Acknowledgements. We thank Kevin Matulef for several helpful discussions.

References

1. Andoni, A., Ba, K.D., Indyk, P., Woodruff, D.: Efficient sketches for earth-mover distance, with applications. In: Proc. 50th Annual IEEE Symposium on Foundations of Computer Science, pp. 324–330 (2009)
2. Andoni, A., Indyk, P., Krauthgamer, R.: Overcoming the ℓ_1 non-embeddability barrier: Algorithms for product metrics. In: SODA (2009)
3. Buhrman, H., Cleve, R., Wigderson, A.: Quantum vs. classical communication and computation. In: Proc. 30th Annual ACM Symposium on the Theory of Computing, pp. 63–68 (1998)
4. Clarkson, K.L., Hazan, E., Woodruff, D.P.: Sublinear optimization for machine learning. In: Proc. 51st Annual IEEE Symposium on Foundations of Computer Science, pp. 449–457 (2010)
5. Cormode, G., Muthukrishnan, S.: An improved data stream summary: the count-min sketch and its applications. *J. Alg.* 55(1), 58–75 (2005); preliminary version in Proc. 6th Latin American Theoretical Informatics Symposium, pp. 29–38 (2004)
6. Cormode, G., Muthukrishnan, S.: Space efficient mining of multigraph streams. In: Proc. 24th ACM Symposium on Principles of Database Systems, pp. 271–282 (2005)
7. Ergün, F., Jowhari, H.: On distance to monotonicity and longest increasing subsequence of a data stream. In: Proc. 19th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 730–736 (2008)
8. Frank, O.: Estimation of the number of vertices of different degrees in a graph. *Journal of Statistical Planning and Inference* 4(1), 45–50 (1980)
9. Frankl, P., Rödl, V.: Forbidden intersections. *Trans. Amer. Math. Soc.* 300(1), 259–286 (1987)
10. Gál, A., Gopalan, P.: Lower bounds on streaming algorithms for approximating the length of the longest increasing subsequence. In: Proc. 48th Annual IEEE Symposium on Foundations of Computer Science, pp. 294–304 (2007)
11. Gopalan, P., Jayram, T.S., Krauthgamer, R., Kumar, R.: Estimating the sortedness of a data stream. In: Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 318–327 (2007)
12. Indyk, P.: Algorithms for dynamic geometric problems over data streams. In: Proc. 36th Annual ACM Symposium on the Theory of Computing, pp. 373–380 (2004)
13. Jayram, T., Woodruff, D.P.: The data stream space complexity of cascaded norms. In: FOCS, pp. 765–774 (2009)
14. Kushilevitz, E., Nisan, N.: *Communication Complexity*. Cambridge University Press, Cambridge (1997)
15. Mehlhorn, K., Schmidt, E.M.: Las vegas is better than determinism in vlsi and distributed computing (extended abstract). In: Proc. 14th Annual ACM Symposium on the Theory of Computing, pp. 330–337 (1982)
16. Monemizadeh, M., Woodruff, D.P.: 1-pass relative-error l_p sampling with applications. In: Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1143–1160 (2010)
17. Muthukrishnan, S.: *Data Streams: Algorithms and Applications*. Foundations and Trends in Theoretical Computer Science 1(2), 117–236 (2005)

Everywhere-Tight Information Cost Tradeoffs for Augmented Index^{*}

Amit Chakrabarti and Ranganath Kondapally


Department of Computer Science, Dartmouth College
Hanover, NH 03755, USA

Abstract. For a variety of reasons, a number of recent works have studied the classic communication problem INDEX, and its variant AUGMENTED-INDEX, from a *tradeoff* perspective: how much communication can Alice (the player holding the n data bits) save if Bob (the player holding the index) communicates a nontrivial amount? Recently, Magniez et al. (STOC, 2010), Chakrabarti et al. (FOCS, 2010) and Jain and Nayak gave *information cost* tradeoffs for this problem, where the amount of communication is measured as the amount of information revealed by one player to the other. The latter two works showed that reducing Alice’s communication to sublinear requires at least a constant amount of communication from Bob.

Here, we show that the above result is just one point on a more general tradeoff curve. That is, we extend the earlier result to show that, for all b , either Bob reveals $\Omega(b)$ information to Alice, or else Alice reveals $n/2^{O(b)}$ information to Bob. This tradeoff lower bound is easily seen to be everywhere-tight, by virtue of an easy two-round deterministic protocol. Our lower bound applies to constant-error randomized protocols, with information measured under an “easy” distribution on inputs.

1 Introduction

The INDEX problem is perhaps the most basic and often-encountered problem in communication complexity. In this problem, Alice is given a string $x \in \{0, 1\}^n$, and Bob an index $k \in [n]$ (we use $[n]$ as shorthand for $\{1, 2, \dots, n\}$); they must determine x_k , the k th bit of x . They may use randomization and err with probability $\leq \varepsilon$ on each input. The problem is sometimes also referred to as SET-MEMBERSHIP, based on interpreting Alice’s string as (the characteristic vector of) a subset of $[n]$, and the function to be computed as “is $k \in x$?”

We note two aspects of the importance of this problem. **Firstly**, it is *the* starting point for studying the effects of *rounds* and *communication order* on communication complexity. The problem is hard for one-way communication, where messages go only from Alice to Bob (and Bob announces the output); it is easy to show that Alice can do no better than sending $\Omega(n)$ bits to Bob. However, it becomes exponentially easier when interactive communication is allowed, or even when the communication is one-way, but it is Bob who does the talking: Bob need only send Alice $\lceil \log n \rceil$ bits.  All of

^{*} Work supported in part by NSF Grant IIS-0916565 and a McLane Family Fellowship.

¹ Throughout this paper “log” denotes the logarithm to the base 2.

this is a fairly easy exercise; it is, in fact, Exercise 4.20 in the textbook of Kushilevitz and Nisan [15], to which we refer the reader for thorough coverage of the basics of communication complexity. And these results can be generalized to handle several natural variants and relaxations of the basic problem. **Secondly**, the INDEX problem and its variants are important tools in proving a number of lower bounds in the data stream model and on data structures. For instance, the works of Jayram et al. [13] and Kane et al. [14] on the 1-pass space complexity of approximating frequency moments and L_p norm (for $1 \leq p \leq 2$), of Feigenbaum et al. [9] on lower bounds for various graph streaming problems including connectivity and diameter approximation, Woodruff's lower bound for approximating frequent items in a data stream [20], and cell probe lower bounds of Miltersen et al. [17] and Pătraşcu [18].

Thanks to its fundamental role and great importance, the INDEX problem and its variants — in particular, the AUGMENTED-INDEX problem [8,14], which we study here — have been subjected to plenty of scrutiny in recent research. Although the basic $\Omega(n)$ -vs- $O(\log n)$ result quoted above is an easy exercise, recent work has required much more sophisticated analysis, and has arguably led to some interesting mathematics. Our work here should be understood in this context. Our goal is to provide the tightest analysis yet of the most relaxed version of this problem. We now turn to a review of the most relevant past work on the problem, following which we can state our new results precisely.

2 Preliminaries, Context, and Our Result

The starting point of the deeper results on INDEX is to consider *tradeoff* lower bounds: if both Alice and Bob are allowed to send out nontrivial numbers of bits — a and b , respectively — then what pairs of values (a, b) are achievable? The basic result quoted above only considers the special cases $b = O(1)$ and $a = O(1)$. It is not hard to give a simple two-round protocol where Bob sends Alice $b \leq \log n$ bits (e.g., the most significant b bits of his input), Alice then replies with an appropriate subset of $a = \lceil n/2^b \rceil$ bits of her input, from which Bob then reads off x_k . This raises the question of whether this tradeoff is the best possible. Over the years, we have seen successively stronger ways of answering “Yes” to this question, by lower-bounding various relaxed versions of the INDEX problem (note that relaxing the problem makes it formally easier to obtain upper bounds, and hence strengthens any lower bound we can prove). The relaxations considered are as follows.

Augmentation. We allow Bob some side information. Specifically, at the start of the protocol, the $(k - 1)$ -bit prefix of x , which we denote $x_{1:k-1}$, is revealed to Bob. The resulting problem is called AUGMENTED-INDEX.

Information. We charge each player only for the amount of information their messages reveal to the other player (see Definition 1), and not for the number of bits they send. The resulting cost measure is called *information cost* (as opposed to the basic measure, which is *communication cost*). This a relaxation because, by basic information theory, a message can reveal at most as much information as its length.

Verification. We give Bob a check bit $c \in \{0, 1\}$, and require the players to compute the function $\text{AIV}(x, k, c) = x_k \oplus c$; the name AIV is shorthand for “augmented

index verification.” We then measure information cost under an input distribution that ensures that x_k always equals c . Such a distribution should be “easy” on the players, because the problem’s distributional complexity under it is zero. Thus, in a sense, the task facing the players is reduced to cheaply *verifying* that x_k in fact equals c . The catch is that their protocol must continue to achieve low error for all inputs (x, k, c) .

These relaxations have not been chosen arbitrarily; rather, each has been motivated by a specific application. Augmentation has been used in the study of sketching complexity [2] and for proving tight space lower bounds for a wide range of problems in the data stream model [7,8,14]. Information cost was formally introduced by Chakrabarti et al. [6] as a technique for proving direct sum theorems in communication and, in the context of INDEX, was introduced by Jain et al. [12] as a means to quantify privacy loss. The use of “easy” input distributions to measure information cost (i.e., the relaxation to “verification” problems) was pioneered by Bar-Yossef et al. [3] as an important step in proving more sophisticated direct sum theorems.² Such direct sum arguments are at the heart of a number of tight lower bounds for data stream problems [5,4,10,16]; they are also implicit in several data structure lower bounds by way of communication lower bounds on LOPSIDED-SET-DISJOINTNESS [18].

We now summarize the key previous lower bounds results on INDEX and its variants. Ablayev [1] analyzed the one-way communication complexity of the basic INDEX problem using information-theoretic ideas. Miltersen et al. [17] extended the one-way lower bound to AUGMENTED-INDEX (see also [2]). Further, Miltersen et al. proved the lower bound $a \geq n/2^{O(b)}$ for any randomized protocol for INDEX in which Alice and Bob send a and b bits respectively. Jain et al. [12] extended this tradeoff to information cost (with a and b now representing the respective information costs), but under a uniform input distribution, which is a “hard” distribution for INDEX.

Magniez et al. [16] were the first to formally consider “easy” distributions in this context. They proved a hybrid tradeoff lower bound in a restricted communication setting; specifically, they proved that in a two-round protocol for AIV with an Alice \rightarrow Bob \rightarrow Alice communication pattern and $O(1/n^2)$ error, either Alice sends $\Omega(n)$ bits or Bob reveals $\Omega(\log n)$ information. Their main motivation was proving a tight one-pass space lower bound for recognizing Dyck languages in the data stream model. Chakrabarti et al. [4] generalized their result, removing the restriction on the communication pattern, and handling $O(1)$ error. They showed the following specific tradeoff: either $a = \Omega(n)$ or $b = \Omega(1)$, where now a and b are both information costs. Jain and Nayak [11] gave an alternate proof of this result. It is also worth noting that, in earlier work, Pătraşcu [18] implicitly dealt with the easy-distribution issue, but only for (non-augmented) INDEX: one can infer from his work the result that, for constant δ , either $b \geq \delta \log n$ or $a \geq n^{1-O(\delta)}$.

Our main result is a culmination of this moderately long line of research. Whereas the most recent previous result [4,11] provides only one point on the tradeoff curve for

² This brief discussion does not do justice to the ingenuity of studying information cost under “easy” distributions and the power of this idea in obtaining direct sum results. We urge the interested reader to study the relevant discussion in Bar-Yossef et al. [3] for a more complete picture.

AIV, namely, when Bob reveals a tiny constant amount of information, we now provide a complete characterization of the tradeoff.

Theorem 1 (Main Theorem, informal). *Let $b \leq (\log n)/100$ be a positive integer. In any $\frac{1}{3}$ -error protocol for AIV, either Alice reveals $n/2^{O(b)}$ information, or Bob reveals at least b information. Here, information is measured according to an “easy” input distribution.*

While our focus is on proving this fundamental theorem for its own sake, we note that as an easy application, we obtain a cell-probe lower bound for the well-studied partial sums problem [19]. In the partial sums problem, we are given an input array $A[1 \dots n]$ of bits. We preprocess it into a data structure so that given a query $i \in [n]$, we can determine $\text{RANK}(i) = \sum_{k=1}^i A[k]$. We want to minimize the space used by the data structure in memory (organized as an array of *cells*) while making as few memory lookups as possible to answer the query. Using our Main Theorem, we can prove a space/query time tradeoff for this problem in the cell-probe model.

Corollary 1. *Consider the cell-probe model with word size w . Let S denote the space used by an algorithm for the partial sums problem and let t be the number of cell-probes made by the algorithm. Then $tw \geq n/S^{O(t)}$.*

Proof. The following is a simple reduction from AIV to the partial sums problem.

Let D be a t -cell-probe (randomized) algorithm for the partial sums problem with space usage S . Alice and Bob use D to solve $\text{AIV}(x, k, c)$. Alice uses her input string x as the input array and preprocesses it into a data structure as specified by D . Bob on input $(x_{1:k-1}, k, c)$ queries, as specified by D , the preprocessed data structure for $\text{RANK}(k)$. For each cell probe, Bob sends Alice the addresses of the cells to probe. This requires $\log S$ bits of communication. Alice then sends Bob the contents of the cells specified by Bob. This requires w bits of communication. After t cell probes, Bob can compute $x_k = \text{RANK}(k) - \text{RANK}(k-1)$ and output $\text{AIV}(x, k, c)$. In total, Bob communicated $t \log S$ bits and Alice communicated tw bits. The corollary now follows from the Main Theorem. \square

3 Formal Version of Main Theorem and Proof Outline

Let $\mathcal{D} := \{0, 1\}^n \times [n] \times \{0, 1\}$. The augmented index verification function, $\text{AIV} : \mathcal{D} \rightarrow \{0, 1\}$, is defined by

$$\text{AIV}(x, k, c) := x_k \oplus c, \text{ for } (x, k, c) \in \mathcal{D}.$$

We also use AIV to denote the communication problem wherein Alice receives x and Bob receives k, c , and $x_{1:k-1}$, and their goal is to compute $\text{AIV}(x, k, c)$ correctly, with high probability. Let μ, μ_0 denote the uniform distributions on $\mathcal{D}, \mathcal{D}_0$ respectively, where

$$\mathcal{D}_0 := \{(x, k, c) \in \mathcal{D} : \text{AIV}(x, k, c) = 0\} = \{(x, k, c) \in \mathcal{D} : x_k = c\}.$$

Let $(X, K, C) \sim \mu$. Let μ_0 denote the conditional distribution $\mu \mid (X_K = C)$.

Definition 1 (Information costs). Let P be a randomized protocol for AIV and η a distribution on \mathcal{D} . Consider the (correlated) random variables X, K, C, R, T , where $(X, K, C) \sim \eta$, R is the public random string in P , and T is the transcript of messages sent between Alice and Bob as they execute P . The information cost incurred by Alice (resp. Bob) in P under η is defined as the amount of information revealed by Alice to Bob (resp. Bob to Alice) through T , and is denoted by $\text{icost}_\eta^A(P)$ (resp. $\text{icost}_\eta^B(P)$). To be precise,

$$\text{icost}_\eta^A(P) := I(T : X \mid X_{1:K-1}, K, C, R); \quad \text{icost}_\eta^B(P) := I(T : K, C \mid X, R).$$

Theorem 2 (Main Theorem, formal). Let n and b be positive integers, with n large enough and $b \leq (\log n)/100$. There exist positive constants ε and c such that if P is a randomized protocol for AIV_n with error at most ε under μ , then either $\text{icost}_{\mu_0}^A(P) \geq n/2^{cb}$ or $\text{icost}_{\mu_0}^B(P) \geq b$. In particular, the same tradeoff holds if P has worst case two-sided error at most ε .

Proof. The high-level outline of our proof is similar to that in [4]. The proof has two stages. In the first stage we assume the contrary and then zoom in on a specific setting, \mathfrak{R} , of the public random string of P (thereby reducing it to a private-coin protocol) and a single transcript, t , that has two properties: (1) It is *fat* (to borrow a term from [4]), meaning that from each player’s point of view, the entropy remaining in the other player’s input, conditioned on \mathfrak{R} and t , is high. (2) It has low *corruption*, meaning that on inputs consistent with t , the protocol is mostly correct. This stage is formalized in Lemma 1 below.

In the second stage, we consider the distribution $\rho_{\mathfrak{R},t}$ induced on \mathcal{D} by the conditioning on \mathfrak{R} and t . This distribution is special because it arises from a communication protocol: it satisfies the *weak rectangle property* given in Lemma 3 below. As a result, we can show that it cannot simultaneously satisfy these fatness and corruption properties; see Lemma 2 below. This gives us our contradiction, and completes the proof. \square

We remark that the lion’s share of our technical innovation lies in the second stage. We shall explain, at the appropriate juncture, why techniques in the earlier proofs [4,11] did not suffice.

Definition 2 (Fatness, Witness Set, Corruption). A transcript t of a private-coin protocol for AIV is said to (D, δ, r) -fat if there exists a witness set $\mathcal{K} \subseteq [n]$, with $|\mathcal{K}| \geq n - n/2^{Dr}$, such that

$$\forall k \in \mathcal{K} : H(\tilde{X} \mid \tilde{X}_{1:k-1}, \tilde{C}, \tilde{K} = k, \tilde{X}_{\tilde{K}} = \tilde{C}) \geq n - k - n/2^{Dr}, \text{ and} \quad (1)$$

$$H(\tilde{K}, \tilde{C} \mid \tilde{X}, \tilde{X}_{\tilde{K}} = \tilde{C}) \geq \log n - \delta r, \quad (2)$$

where $(\tilde{X}, \tilde{K}, \tilde{C}) \sim \rho_t$ and ρ_t is the distribution obtained by conditioning μ on the transcript t . It is said to ε -corrupt if

$$\text{out}(t) = 0, \text{ and } \mathbb{E}[\text{AIV}(\tilde{X}, \tilde{K}, \tilde{C})] \leq \varepsilon. \quad (3)$$

Lemma 1 (First Stage of Proof of Main Theorem). Suppose P is a private-coin protocol for AIV_n such that $\text{err}_\mu(P) \leq \varepsilon$, $\text{icost}_{\mu_0}^A(P) \leq n/2^{Dr}$, and $\text{icost}_{\mu_0}^B(P) \leq \delta r$. Then P has a transcript that is $(D/100, 100\delta, r)$ -fat and (100ε) -corrupt.

Proof. This stage of the proof consists of a sequence of averaging arguments and applications of Markov inequalities. It proceeds along the lines of [4], and is only slightly more complicated: there are some new twists because our notion of fatness is more nuanced than theirs. A complete proof is included in the full version of the paper. \square

Lemma 2 (Fat Transcript Lemma; Second Stage of Proof of Main Theorem). *There exist positive real constants D, δ, ε such that it is impossible for a transcript of a private-coin protocol for AIV_n to be both (D, δ, r) -fat and ε -corrupt.*

As mentioned above, the proof of this lemma requires us to use the special structure of ρ_t . The following lemma gives us the crucial structural property we need: it is ultimately a consequence of the basic *rectangle property* of deterministic communication protocols [15, Ch. 1]. It can be viewed as a weak version of the rectangle property (or rather, its randomized analogue due to Bar-Yossef et al. [3, Lemma 6.7]), where the weakness is because Alice and Bob share some of the input to the communication problem.

Lemma 3 (Weak Rectangle Property; Lemma 6 of [4]). *Let $\mathcal{X} = \{0, 1\}^n$ and $\mathcal{Y} = \{(w, k, c) \in \{0, 1\}^* \times [n] \times \{0, 1\} : |w| = k - 1\}$. Let P be a private-coin protocol in which Alice receives a string $x \in \mathcal{X}$ while Bob receives $(w, k, c) \in \mathcal{Y}$, with the promise that $w = x_{1:k-1}$. Then, for every transcript t of P , there exist functions $f_t : \mathcal{X} \rightarrow \mathbb{R}^+$ and $g_t : \mathcal{Y} \rightarrow \mathbb{R}^+$ such that*

$$\forall (x, k, c) \in \{0, 1\}^n \times [n] \times \{0, 1\} : \rho_t(x, k, c) = f_t(x)g_t(x_{1:k-1}, k, c). \quad \square$$

We now turn to the main technical proof in this paper.

Proof (Proof of the Fat Transcript Lemma). Suppose to the contrary that for every choice of D, δ , and ε , there exists a transcript t — and hence, a distribution $\tilde{\rho} = \rho_t$ — such that (1), (2) and (3) do hold, with \mathcal{K} as the witness set. Throughout this proof, we let $(\tilde{X}, \tilde{K}, \tilde{C}) \sim \tilde{\rho}$ and we let $(X, K, C) \sim \rho$ be an independent collection of random variables, where the distribution $\rho := \tilde{\rho} \mid (\tilde{X}_{\tilde{K}} = \tilde{C})$.

Let $f = f_t$ and $g = g_t$ be the functions given by Lemma 3. Let \hat{f} denote the probability distribution on $\{0, 1\}^n$ obtained by normalizing f to the probability simplex, and let $Y \sim \hat{f}$ be a random variable independent of all aforementioned ones.

Define the *mystery* of a string $x \in \{0, 1\}^n$ to be

$$M(x) := \mathbb{E}_K[\mathbb{H}(Y_K \mid Y_{1:K-1} = x_{1:K-1}) \mid X = x].$$

Intuitively, x has a high degree of mystery if, for a typical index k , the prefix $x_{1:k-1}$ is “about as likely” to be followed by a 0 as a 1, when drawing strings from the distribution \hat{f} .

Analyzing this quantity is the key to obtaining a contradiction. On the one hand, there cannot be too much mystery overall if the protocol is correct (indeed, we can say that the protocol’s goal is to “remove the mystery” in x). To be precise, we can show that

$$\varepsilon \geq \mathbb{E}[\text{AIV}(\tilde{X}, \tilde{K}, \tilde{C})] \geq (1 - \varepsilon) \mathbb{H}_b^{-1}(\mathbb{E}[M(X)]), \quad (4)$$

where the first inequality restates (3). Here $\mathbb{H}_b(p) = -p \log p - (1 - p) \log(1 - p)$ is the binary entropy function, and $\mathbb{H}_b^{-1} : [0, 1] \rightarrow [0, \frac{1}{2}]$ is its inverse.

On the other hand, the fatness properties (1) and (2) suggest that most strings x should have high mystery. Indeed, let $\mathcal{X} := \{x \in \{0, 1\}^n : H(K \mid X = x) \geq \log n - D\delta r\}$ be the set of “typical” strings x , for which Alice does not learn too much about Bob’s input. For each $k \in [n]$, we can define a certain set of excluded strings $\mathcal{Z}_k \subseteq \{0, 1\}^n$. Then, we can show that

$$\forall k \in \mathcal{K} \ \forall x \in \mathcal{X} \setminus \mathcal{Z}_k : M(x) \geq \frac{1}{2} \Pr[K \geq k \mid X = x] - 2\delta. \tag{5}$$

Finally, we can show that these conditions on k and x fail only with low probability and that K is often large. This will then imply that, for the setting $\delta = 1/D$, we have

$$\mathbb{E}[M(X)] \geq \frac{1}{2} - \frac{6}{D}, \tag{6}$$

which contradicts (4), for small enough ε and large enough D . □

4 Proofs of Technical Lemmas

We now turn to proving the claims made within the above proof of the Fat Transcript Lemma.

4.1 Derivation of (4)

Let $f = f_t$ and $g = g_t$ be the functions given by Lemma 3, and let $\rho = \tilde{\rho} \mid (\tilde{X}_{\tilde{K}} = \tilde{C})$. In the summations below, c, u, v, w, x , and k range over $\{0, 1\}, \{0, 1\}^{k-1}, \{0, 1\}^{n-k}, \{0, 1\}^{n-k+1}, \{0, 1\}^n$, and $[n]$, respectively. We use notation of the form “ $u0v$ ” to denote the concatenation of the string u , the length-1 string “0”, and the string v . We have

$$\begin{aligned} \mathbb{E}[\text{AIV}(\tilde{X}, \tilde{K}, \tilde{C})] &= \sum_k \sum_x \sum_c \tilde{\rho}(x, k, c) \cdot \text{AIV}(x, k, c) \\ &= \sum_k \sum_u \sum_{b \in \{0,1\}} \sum_v \sum_c \tilde{\rho}(ubv, k, c) \cdot \text{AIV}(ubv, k, c). \end{aligned} \tag{7}$$

Observe that for every triple (x, k, c) , we have

$$\tilde{\rho}(x, k, c) \geq \Pr[\tilde{X}_{\tilde{K}} = \tilde{C}] \cdot \rho(x, k, c) \geq (1 - \varepsilon)\rho(x, k, c),$$

where the last step uses (3). Now, noting that $\text{AIV}(ubv, k, c) = 1$ iff $b \neq c$, we can manipulate (7) as follows, using the above inequality at step (8).

$$\begin{aligned} \mathbb{E}[\text{AIV}(\tilde{X}, \tilde{K}, \tilde{C})] &= \sum_k \sum_u \sum_v \left(\tilde{\rho}(u0v, k, 1) + \tilde{\rho}(u1v, k, 0) \right) \\ &= \sum_k \sum_u \left(g(u, k, 1) \sum_v f(u0v) + g(u, k, 0) \sum_v f(u1v) \right) \\ &\geq \sum_k \sum_u \left(g(u, k, 0) + g(u, k, 1) \right) \cdot \min \left\{ \sum_v f(u0v), \sum_v f(u1v) \right\} \\ &= \sum_k \sum_u \left(g(u, k, 0) + g(u, k, 1) \right) \left(\sum_w f(uw) \right) \min \left\{ \frac{\sum_v f(u0v)}{\sum_w f(uw)}, \frac{\sum_v f(u1v)}{\sum_w f(uw)} \right\} \\ &= \sum_k \sum_u \left(\sum_w \sum_c \tilde{\rho}(uw, k, c) \right) \times \\ &\quad \min \{ \Pr[Y_k = 0 \mid Y_{1:k-1} = u], \Pr[Y_k = 1 \mid Y_{1:k-1} = u] \} \\ &\geq (1 - \varepsilon) \sum_k \sum_x \sum_c \rho(x, k, c) \cdot H_b^{-1}(H(Y_k \mid Y_{1:k-1} = x_{1:k-1})) \end{aligned} \tag{8}$$

$$\begin{aligned}
 &= (1 - \varepsilon) \mathbb{E}_{X,K} [\mathbb{H}_b^{-1}(\mathbb{H}(Y_K | Y_{1:K-1} = X_{1:K-1}))] \\
 &\geq (1 - \varepsilon) \mathbb{H}_b^{-1}(\mathbb{E}_{X,K} [\mathbb{H}(Y_K | Y_{1:K-1} = X_{1:K-1})]) \\
 &= (1 - \varepsilon) \mathbb{H}_b^{-1}(\mathbb{E}[M(X)]),
 \end{aligned} \tag{9}$$

where (8) also uses the observation that for any binary random variable Z , $\min\{\Pr[Z = 0], \Pr[Z = 1]\} = \mathbb{H}_b^{-1}(\mathbb{H}(Z))$, and (9) uses Jensen’s inequality and the convexity of \mathbb{H}_b^{-1} .

In the earlier works of Chakrabarti et al. [4] and Jain and Nayak [11], it was critical, from here on, that the marginal of ρ on (x, k) was almost a product distribution. (Jain and Nayak used this fact implicitly.) This in turn depended crucially on Bob’s information cost being less than a small constant. For our more general tradeoff, we do not have this luxury. Instead, we proceed as shown below.

4.2 Derivation of (5)

Recall that our goal is to come up with a set \mathcal{Z}_k of excluded strings for each $k \in [n]$ and then prove that

$$\forall k \in \mathcal{K} \quad \forall x \in \mathcal{X} \setminus \mathcal{Z}_k : M(x) \geq \frac{1}{2} \Pr[K \geq k | X = x] - 2\delta. \tag{5 restated}$$

Here is how we define \mathcal{Z}_k . Below, and throughout the rest of this paper, we use J to denote a random variable distributed *uniformly* in an interval of the form $\{k + 1, k + 2, \dots, n\}$; we write this briefly as $J \in_R [k + 1, n]$.

$$\mathcal{Z}_k := \left\{ x \in \{0, 1\}^n : \mathbb{E}_{J \in_R [k+1, n]} [\mathbb{H}(Y_J | Y_{1:J-1} = x_{1:J-1})] \leq 1 - \frac{Dn}{(n - k)2^{Dr}} \right\}. \tag{10}$$

The following observation will be useful here, and later in the paper.

Lemma 4. *Suppose $r \geq 1$ and $\beta > 0$. Let K be a random variable taking values in $[n]$, and $\mathcal{L} \subseteq [n]$. If $\mathbb{H}(K) \geq \log n - r$, and $|\mathcal{L}| \leq n/2^{\beta r}$, then $\Pr[K \in \mathcal{L}] \leq 2/\beta$.*

Proof. Let $p = \Pr[K \in \mathcal{K}]$. We have

$$\begin{aligned}
 \log n - r &\leq \mathbb{H}(K) \\
 &= p \cdot \mathbb{H}(K | K \in \mathcal{L}) + (1 - p) \cdot \mathbb{H}(K | K \notin \mathcal{L}) + \mathbb{H}_b(p) \\
 &\leq p \cdot (\log n - \beta r) + (1 - p) \cdot \log n + 1 \\
 &= \log n - \beta r p + 1.
 \end{aligned}$$

Solving for p , we get $p \leq (r + 1)/(\beta r) \leq 2/\beta$, since $r \geq 1$. □

Now fix an index $k \in \mathcal{K}$ and a string $x \in \mathcal{X} \setminus \mathcal{Z}_k$. Consider the non-mysterious bit positions in the string x , i.e., the set $\mathcal{B}_x := \{j \in [k + 1, n] : \mathbb{H}(Y_j | Y_{j-1} = x_{1:j-1}) < 1/2\}$. We have

$$\begin{aligned}
 M(x) &= \mathbb{E}[\mathbb{H}(Y_K | Y_{1:K-1} = x_{1:K-1}) | X = x] \\
 &\geq \frac{1}{2} \Pr[K > k \wedge K \notin \mathcal{B}_x | X = x]
 \end{aligned}$$

$$\geq \frac{1}{2}(\Pr[K \geq k \mid X = x] - \Pr[K \in \{k\} \cup \mathcal{B}_x \mid X = x]).$$

To finish the proof, it suffices to show that the probability being subtracted off is at most 4δ . We will now show that K , when conditioned on x , is unlikely to be in \mathcal{B}_x . By definition of \mathcal{Z}_k , for any $x \notin \mathcal{Z}_k$ we have

$$\mathbb{E}_{J \in \mathcal{R}[k+1, n]}[1 - \mathbb{H}(Y_J \mid Y_{1:J-1} = x_{1:J-1})] \leq Dn/(n - k)2^{Dr}.$$

Thus, by Markov's inequality, we can conclude that

$$\Pr[J \in \mathcal{B}_x] = \Pr[1 - \mathbb{H}(Y_J \mid Y_{1:J-1} = x_{1:J-1}) \geq 1/2] \leq 2Dn/(n - k)2^{Dr}$$

and therefore that $|\{k\} \cup \mathcal{B}_x| \leq 1 + 2Dn/2^{Dr} \leq n/2^{D\delta r/2\delta}$. Recall that we have $x \in \mathcal{X}$. By definition of \mathcal{X} , this means that $\mathbb{H}(K \mid X = x) \geq \log n - D\delta r$. Now, by Lemma 4, we have $\Pr[K \in \{k\} \cup \mathcal{B}_x \mid X = x] \leq 2 \cdot (2\delta) = 4\delta$, as required.

4.3 Derivation of (6)

We have just shown that for $k \in \mathcal{K}$ and $x \in \mathcal{X} \setminus \mathcal{Z}_k$, we have $M(x) \geq \frac{1}{2} \Pr[K \geq k \mid X = x] - 2\delta$. Our goal now is to lower bound $\mathbb{E}[M(x)]$.

Define $\mathcal{K}' := \{k \in \mathcal{K} : k \leq n - Dn/2^{Dr}\}$. The next two lemmas use the fatness properties of ρ to conclude that \mathcal{K}' and \mathcal{X} are, in a sense, typical sets.

Lemma 5. *We have $\Pr[K \notin \mathcal{K}'] \leq 4\delta/D$.*

Proof. Definition 2 tells us that $|\mathcal{K}| \geq n - n/2^{Dr}$. Therefore, we have

$$|[n] \setminus \mathcal{K}'| \leq Dn/2^{Dr} + |[n] \setminus \mathcal{K}| \leq Dn/2^{Dr} + n/2^{Dr} \leq n/2^{(D/2\delta) \cdot \delta r}.$$

Eq. (2) gives us $\mathbb{H}(K, C \mid X) \geq \log n - \delta r$, which implies $\mathbb{H}(K) \geq \log n - \delta r$. Now Lemma 4 completes the proof. \square

Lemma 6. *We have $\Pr[X \notin \mathcal{X}] \leq 1/D$.*

Proof. From (2), we have $\mathbb{E}[\log n - \mathbb{H}(K \mid X)] \leq \delta r$. The lemma now follows from the definition of \mathcal{X} and Markov's inequality. \square

To aid our estimation we shall choose, for each string x , a suitable index k at which to apply (5). For each $x \in \{0, 1\}^n$, define its *critical index* to be

$$k_x^* := \min\{k \in \mathcal{K}' : x \notin \mathcal{Z}_k\},$$

where the minimum over an empty set defaults to ∞ . Notice that for all $x \in \mathcal{X}$ we have $M(x) \geq \frac{1}{2} \Pr[K \geq k_x^* \mid X = x] - 2\delta$. Thus,

$$\begin{aligned} \mathbb{E}[M(X)] &\geq \Pr[X \in \mathcal{X}] \cdot \mathbb{E}[M(X) \mid X \in \mathcal{X}] \\ &\geq (1 - \frac{1}{D}) \left(\frac{1}{2} \Pr[K \geq k_X^* \mid X \in \mathcal{X}] - 2\delta \right), \end{aligned} \tag{11}$$

where we have used Lemma 6. Now, we bound (11) as follows.

$$\begin{aligned} \Pr[K \geq k_X^* \mid X \in \mathcal{X}] &= \sum_{x \in \mathcal{X}} \Pr[X = x \wedge K \geq k_x^* \mid X \in \mathcal{X}] \\ &\geq \sum_{k \in \mathcal{K}'} \sum_{x \in \mathcal{X} \setminus \mathcal{Z}_k} \Pr[X = x \wedge K = k \mid X \in \mathcal{X}] \end{aligned}$$

$$\begin{aligned}
 &= \sum_{k \in \mathcal{K}'} \Pr[K = k] \sum_{x \in \mathcal{X} \setminus \mathcal{Z}_k} \Pr[X = x \mid K = k, X \in \mathcal{X}] \\
 &= \sum_{k \in \mathcal{K}'} \Pr[K = k] \cdot \Pr[X \notin \mathcal{Z}_k \mid K = k, X \in \mathcal{X}] \\
 &= \Pr[X \notin \mathcal{Z}_k \wedge K \in \mathcal{K}' \mid X \in \mathcal{X}] \\
 &\geq \Pr[X \notin \mathcal{Z}_k \wedge K \in \mathcal{K}'] - \Pr[X \notin \mathcal{X}] \\
 &= \Pr[K \in \mathcal{K}'] \cdot \Pr[X \notin \mathcal{Z}_k \mid K \in \mathcal{K}'] - \Pr[X \notin \mathcal{X}] \\
 &\geq \left(1 - \frac{4\delta}{D}\right) \left(1 - \frac{1}{D}\right) - \frac{1}{D} \geq 1 - \frac{6}{D}, \tag{12}
 \end{aligned}$$

where (12) follows from Lemmas 5, 6 and the following claim. Plugging this into (11), we get

$$\mathbb{E}[M(X)] \geq \frac{1}{2} - \frac{6}{D},$$

with the choice $\delta = 1/D$.

Claim. We have $\Pr[X \in \mathcal{Z}_k \mid K \in \mathcal{K}'] \leq 1/D$.

Proof. Recall that our distribution ρ is conditioned on $X_K = C$. Using the chain rule for entropy, we have

$$\begin{aligned}
 \frac{H(X \mid X_{1:K-1}, K = k, C)}{n - k} &= \frac{1}{n - k} \sum_{j=k+1}^n H(X_j \mid X_{1:j-1}, K = k, C) \\
 &= \mathbb{E}_{J \in_R[k+1, n]} [H(X_J \mid X_{1:J-1}, K = k)] \\
 &= \mathbb{E}_{J \in_R[k+1, n]} [\mathbb{E}_u [H(X_J \mid X_{1:J-1} = u, K = k) \mid K = k]]. \tag{13}
 \end{aligned}$$

We could remove the conditioning on C in the latter two expressions because, for $J > k$, the conditions $K = k$ and $X_K = C$ together determine C .

Our next step is conceptually critical. We will switch the random variable inside the entropy expression in (13) from X to Y using the following claim, which makes use of the weak rectangle property, Lemma 3.

Claim. For any $j, k \in [n]$ with $j > k$, and any $u \in \{0, 1\}^{k-1}$, we have the following equivalence of conditional distributions:

$$X \mid (X_{1:j-1} = u, K = k) \equiv Y \mid (Y_{1:j-1} = u).$$

In other words, for any $z \in \{0, 1\}^n$, $\Pr[X = z \mid X_{1:j-1} = u, K = k] = \Pr[Y = z \mid Y_{1:j-1} = u]$.

Proof. It is easy to see that if $z_{1:j-1} \neq u$, then both the probabilities are equal to 0. If not, we have $u_k = z_k$. Let $f = f_t$ and $g = g_t$ be the functions in Lemma 3. By that lemma, we can write $\rho(z, k, u_k) = \alpha f(z)g(u_{1:k-1}, k, u_k)$, where α is a normalization factor. Letting w range over $\{0, 1\}^{n-j+1}$ below, we have

$$\begin{aligned}
 \Pr[X = z \mid X_{1:j-1} = u, K = k] &= \rho(z, k, u_k) / \sum_w \rho(uw, k, u_k) \\
 &= \frac{\alpha f(z)g(u_{1:k-1}, k, u_k)}{\sum_w \alpha f(uw)g(u_{1:k-1}, k, u_k)} = \frac{f(z)}{\sum_w f(uw)} = \Pr[Y = z \mid Y_{1:j-1} = u].
 \end{aligned}$$

□

Making the switch from X to Y in (I3), we have

$$\begin{aligned} \frac{H(X \mid X_{1:K-1}, K = k, C)}{n - k} &= \mathbb{E}_{J \in \mathcal{R}[k+1, n]} [\mathbb{E}_u [H(Y_J \mid Y_{1:J-1} = u) \mid K = k]] \\ &= \mathbb{E}_{J \in \mathcal{R}[k+1, n]} [\mathbb{E}_X [H(Y_J \mid Y_{1:J-1} = X_{1:J-1}) \mid K = k]] \\ &= \mathbb{E}_X [\mathbb{E}_{J \in \mathcal{R}[k+1, n]} [H(Y_J \mid Y_{1:J-1} = X_{1:J-1})] \mid K = k] \end{aligned} \quad (14)$$

By (II), for any $k \in \mathcal{K}'$, we have

$$\frac{H(X \mid X_{1:K-1}, K = k, C)}{n - k} \geq 1 - \frac{n}{(n - k)2^{Dr}}.$$

Using this in (I4), we get

$$\mathbb{E}_X [1 - \mathbb{E}_{J \in \mathcal{R}[k+1, n]} [H(Y_J \mid Y_{1:J-1} = X_{1:J-1})] \mid K = k] \leq \frac{n}{(n - k)2^{Dr}}.$$

Since the expression inside the outer expectation is non-negative, by Markov's inequality, the probability that it exceeds $Dn/(n - k)2^{Dr}$ is at most $1/D$. That is, $\Pr[X \in \mathcal{Z}_k \mid K = k] \leq 1/D$. Hence, $\Pr[X \in \mathcal{Z}_K \mid K \in \mathcal{K}'] \leq 1/D$, which completes the proof of the claim. \square

References

1. Ablayev, F.: Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science* 175(2), 139–159 (1996)
2. Bar-Yossef, Z., Jayram, T.S., Krauthgamer, R., Kumar, R.: The sketching complexity of pattern matching. In: *Proc. 8th International Workshop on Randomization and Approximation Techniques in Computer Science*, pp. 261–272 (2004)
3. Bar-Yossef, Z., Jayram, T.S., Kumar, R., Sivakumar, D.: An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.* 68(4), 702–732 (2004)
4. Chakrabarti, A., Cormode, G., Kondapally, R., McGregor, A.: Information cost tradeoffs for augmented index and streaming language recognition. In: *Proc. 51st Annual IEEE Symposium on Foundations of Computer Science*, pp. 387–396 (2010)
5. Chakrabarti, A., Khot, S., Sun, X.: Near-optimal lower bounds on the multi-party communication complexity of set disjointness. In: *Proc. 18th Annual IEEE Conference on Computational Complexity*, pp. 107–117 (2003)
6. Chakrabarti, A., Shi, Y., Wirth, A., Yao, A.C.: Informational complexity and the direct sum problem for simultaneous message complexity. In: *Proc. 42nd Annual IEEE Symposium on Foundations of Computer Science*, pp. 270–278 (2001)
7. Clarkson, K.L., Woodruff, D.P.: Numerical linear algebra in the streaming model. In: *Proc. 41st Annual ACM Symposium on the Theory of Computing*, pp. 205–214 (2009)
8. Do Ba, K., Indyk, P., Price, E., Woodruff, D.P.: Lower bounds for sparse recovery. In: *Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1190–1197 (2010)
9. Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., Zhang, J.: Graph distances in the data-stream model. *SIAM J. Comput.* 38(6), 1709–1727 (2008); Preliminary version in *Proc. 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 745–754 (2005)
10. Gronemeier, A.: Asymptotically optimal lower bounds on the NIH-multi-party information complexity of the AND-function and disjointness. In: *Proc. 26th International Symposium on Theoretical Aspects of Computer Science*, pp. 505–516 (2009)

11. Jain, R., Nayak, A.: The space complexity of recognizing well-parenthesized expressions in the streaming model: the index function revisited. Technical Report Revision #1 to TR10-071, Electronic Colloquium on Computational Complexity (July 2010), <http://eccc.hpi-web.de/>
12. Jain, R., Radhakrishnan, J., Sen, P.: A property of quantum relative entropy with an application to privacy in quantum communication. *J. ACM* 56(6) (2009)
13. Jayram, T.S., Kumar, R., Sivakumar, D.: The one-way communication complexity of gap hamming distance. *Theor. Comput.* 4(1), 129–135 (2008)
14. Kane, D.M., Nelson, J., Woodruff, D.P.: On the exact space complexity of sketching and streaming small norms. In: Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1161–1178 (2010)
15. Kushilevitz, E., Nisan, N.: *Communication Complexity*. Cambridge University Press, Cambridge (1997)
16. Magniez, F., Mathieu, C., Nayak, A.: Recognizing well-parenthesized expressions in the streaming model. In: Proc. 41st Annual ACM Symposium on the Theory of Computing, pp. 261–270 (2010)
17. Miltersen, P.B., Nisan, N., Safra, S., Wigderson, A.: On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.* 57(1), 37–49 (1998); Preliminary version in Proc. 27th Annual ACM Symposium on the Theory of Computing, pp. 103–111 (1995)
18. Pătraşcu, M.: Unifying the landscape of cell-probe lower bounds. Manuscript (2010), <http://people.csail.mit.edu/mip/papers/structures/paper.pdf>
19. Pătraşcu, M., Viola, E.: Cell-probe lower bounds for succinct partial sums. In: Proc. 21st Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 117–122 (2010)
20. Woodruff, D.P.: Optimal space lower bounds for all frequency moments. In: Proc. 15th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 167–175 (2004)

A Canonical Form for Testing Boolean Function Properties

Dana Dachman-Soled* and Rocco A. Servedio**

Columbia University, New York, NY, 10027, U.S.A.
{dg2342,ras2105}@columbia.edu

Abstract. In a well-known result Goldreich and Trevisan (2003) showed that every testable graph property has a “canonical” tester in which a set of vertices is selected at random and the edges queried are the complete graph over the selected vertices. We define a similar-in-spirit canonical form for Boolean function testing algorithms, and show that under some mild conditions property testers for Boolean functions can be transformed into this canonical form.

Our first main result shows, roughly speaking, that every “nice” family of Boolean functions that has low noise sensitivity and is testable by an “independent tester,” has a canonical testing algorithm. Our second main result is similar but holds instead for families of Boolean functions that are closed under ID-negative minors. Taken together, these two results cover almost all of the constant-query Boolean function testing algorithms that we know of in the literature, and show that all of these testing algorithms can be automatically converted into a canonical form.

Keywords: property testing, Boolean functions.

1 Introduction

Property testing has emerged as an intensively studied area of theoretical computer science, with close connections to topics such as sublinear-time algorithms and PCPs in complexity theory. Several distinct strands of research have emerged corresponding to different types of objects to be tested: graphs, Boolean functions, error-correcting codes, probability distributions, etc. Each of these sub-areas has developed its own set of tools and techniques, and results often have different “flavors” across these different areas. Given this state of affairs, a natural goal is to obtain a more unified view of property testing by uncovering deeper underlying similarities between general results on testing different kinds of objects. This high-level goal provides the impetus behind the current work.

The aim of this paper is to obtain “canonical” testers for testable Boolean function properties, similar to the canonical testers for graph properties of Goldreich and Trevisan [GT03]. Specialized to properties that are testable with

* Supported in part by an FFSEAS Presidential Fellowship.

** Supported by NSF grants CCF-0347282, CCF-0523664 and CNS-0716245, and by DARPA award HR0011-08-1-0069.

a constant number of queries independent of n , the [GT03] result is essentially as follows: Let \mathcal{P} be any graph property that has a $q(\epsilon)$ -query testing algorithm, independent of the number of vertices n . Then \mathcal{P} is efficiently testable by an algorithm that follows a simple prescribed “canonical form:” it draws $q(\epsilon)$ vertices at random, queries all $\binom{q(\epsilon)}{2}$ edges between those vertices, does some deterministic computation on the resulting data and outputs “accept” or “reject.”

We ask the following natural question: is there a similar “canonical form” for Boolean function property testing algorithms? Such a result would presumably say that any property of Boolean functions that is constant-query testable is in fact constant-query testable by an algorithm that works roughly as follows: it tosses some fair coins, exhaustively queries the function on “all inputs defined by those coin tosses,” does some deterministic computation on the resulting data, and outputs its verdict. We elaborate below, where we give a precise definition of a “canonical Boolean function testing algorithm”. But first, as motivation, it is useful to explain how we may view any Boolean function property testing algorithm as a probability distribution over query strings.

Let \mathcal{P} be any class of Boolean functions that has a testing algorithm A with query complexity $q(\epsilon)$ independent of the number of variables n . (We may assume that A is nonadaptive, since otherwise we can make A nonadaptive in a standard way; this exponentially increases the query complexity but it still does not depend on n .) We view $A = (A_1, A_2)$ as having two stages: First, A_1 generates all $q(\epsilon)$ query strings and then queries them. Next, A_2 does some computation on the results and outputs its verdict. This computation may *a priori* be randomized, but a simple argument (see Section 4.2.3 of [GT03]) shows that without loss of generality it may be assumed to be deterministic.

The sequence of query strings generated in the first stage can be viewed as a draw from some distribution \mathcal{D} over $\{0, 1\}^{q(\epsilon) \times n}$, and since A is nonadaptive, this distribution does not depend on the function f . But in general this distribution may be quite complex and difficult to understand. Is there a simple “canonical form” for the query generation stage of every Boolean function testing algorithm?

A Canonical Form for Boolean Function Testing Algorithms. In the [GT03] result, there is only one type of distribution over queries for all testing algorithms (and this distribution is very simple) – any difference between two $q(\epsilon)$ -query testing algorithms comes from the deterministic computation they do on the data. We would like an analogous result for testing Boolean functions, which similarly involves only one kind of (simple) distribution over queries. Thus motivated, we consider the following canonical form for Boolean function testers:

- **First stage (query generation):** Let z^1, \dots, z^k be independent and uniform random strings from $\{0, 1\}^n$. This defines a natural partition of $[n]$ into 2^k blocks (see Section 3 for details). We say that a string $x = x_1 \dots x_n \in \{0, 1\}^n$ “respects the partition” if within each block B_b all variables x_i are set to the same value. The 2^{2^k} strings in $\{0, 1\}^n$ that respect the partition are the queries the canonical tester makes.
- **Second stage:** With these 2^{2^k} query-answer pairs in hand, the algorithm does some (deterministic) computation and outputs “accept” or “reject.”

Some examples of known testers that can easily be converted to canonical form as described above include the tester of [BLR93] for $GF(2)$ linear functions and the tester of [AKK⁺03] for degree k polynomials over $GF(2)$. Let us consider the [AKK⁺03] tester and see how to convert it to canonical form. The [AKK⁺03] tester works by choosing $k+1$ strings $z^1, \dots, z^{k+1} \in \{0, 1\}^n$ uniformly at random and then querying all points in the induced subspace. If a string x is in the induced subspace of z^1, \dots, z^{k+1} then it must also “respect the partition” induced by z^1, \dots, z^{k+1} . So to convert the [AKK⁺03] tester to our canonical form, all we have to do is ask some more queries.

A natural first hope is to generalize the above examples and show that *every* Boolean function property that is testable using constantly many queries has a “canonical form” constant-query testing algorithm of the above sort. However, E. Blais [Bl10] has observed that there is a simple property that is testable with $O(1/\epsilon)$ queries but does not have a constant-query canonical tester of the above sort: this is the property of being a symmetric Boolean function. Let SYM be the set of all symmetric Boolean functions (i.e., all functions where $f(x)$ is determined by $|x|$, the Hamming weight of x). SYM can be tested with a constant number of queries with the following algorithm:

- Pick $O(1/\epsilon)$ pairs of points $(x^i, y^i) \in \{0, 1\}^n \times \{0, 1\}^n$ by choosing x uniformly at random from $\{0, 1\}^n$ then choosing y uniformly at random from all inputs with the same weight as x .
- Check that for each pair $f(x^i) = f(y^i)$. Accept if this holds; otherwise reject.

It is clear that if $f \in \text{SYM}$ then the above test accepts with probability 1. On the other hand, for any f that is ϵ -far from SYM, with probability at least ϵ the string x^i is one of the “bad” inputs that has the minority output in its level, and with probability at least $1/2$ the string y^i is one of the inputs with the majority output for the same level. So with probability at least $\epsilon/2$, (x^i, y^i) is a witness to the fact that f is not symmetric, and $O(1/\epsilon)$ queries are sufficient to reject f with probability at least $2/3$.

To show that SYM cannot be tested by a constant-query canonical tester, it suffices to show that for $k = o(\log \log n)$, with high probability each of the $2^{2^k} = n^{o(1)}$ queries generated by the tester has different Hamming weight. This can be established by a straightforward but somewhat tedious argument which we omit here (the main ingredients are the observation that each query string x generated by the canonical tester has Hamming weight distributed according to a binomial distribution $B(n, \frac{j}{2^k})$ for some integer j , together with standard anti-concentration bounds on binomial distributions with sufficiently large variance).

The example above shows that, unlike the graph testing setting, it is not the case that *every* constant-query Boolean function tester can be “canonicalized.” So in order to obtain meaningful results on “canonicalizing” Boolean function testers, one must restrict the types of properties and/or testers that are considered; this is precisely what we do in our results, as explained below.

Our Results. Our main results are that certain “nice” testing algorithms, for certain “nice” types of Boolean function properties, can automatically be

converted into the above-described canonical form. Roughly speaking, the testing algorithms we can handle are ones for which every distribution $\mathcal{D}_{x^1, \dots, x^t}$ in the query generation phase is a product of n Bernoulli distributions over the n coordinates (with some slight additional technical restrictions that we describe later). This is a restricted class of algorithms, but it includes many different testing algorithms that have been proposed and analyzed in the Boolean function property testing literature. We call such testing algorithms “Independent testers” (see Section 2.1 for a precise definition), and we give two results showing that independent testers for certain types of properties can be “canonicalized.”

Our first result applies to classes C that are closed under negating variables and contain only functions with low *noise sensitivity*. We say such a class C is *closed under Noisy-Neg minors* (see Definition 1). For such classes C we show:

Theorem 1 (Informal). *If C is closed under Noisy-Neg minors and has a (two-sided) independent tester, then C has a (two-sided) canonical tester.*

Our second result applies to classes C that are closed under identification of variables, negation of variables, and adding or removing irrelevant variables. Following [HR05], we say that such a class is *closed under ID-Neg minors* (see Definition 2). For such classes C we show the following:

Theorem 2 (Informal). *If C is closed under ID-Neg minors and has a one-sided independent tester, then C has a one-sided canonical tester.*

These two results allow us to give “canonical” versions of many different Boolean function property testing algorithms in the literature, including the algorithms of [PRS02, BLR93, AKK⁺03, FKR⁺04, Bla09, DLM⁺07, MOR10] for the classes of dictators, $GF(2)$ -linear functions, $GF(2)$ -deg- d functions, J -juntas, decision lists, size- s decision trees, size- s branching programs, s -term DNFs, size- s Boolean formulas, s -sparse $GF(2)$ polynomials, size- s Boolean circuits, functions with Fourier degree d and halfspaces. One exception comes from [GOS⁺09]; that work gives testing algorithms for the class of Boolean functions with Fourier dimension d (i.e. for “ d -juntas of parities”) and for the class of Boolean functions with s -sparse Fourier spectra. It is easy to see that these classes are not closed under Noisy-Neg minors, since each class contains all parity functions, and the testers of [GOS⁺09] do not have 1-sided error. (However, we note that inspection of the testers provided in [GOS⁺09] shows that they can be straightforwardly “canonicalized” just like the [AKK⁺03] tester discussed in the introduction.)

Our Approach. Developing a canonical tester for Boolean function properties seems to be significantly more challenging than for graph properties. The high-level idea behind the [GT03] graph testing canonicalization result is that if k edges have been queried so far, then all “untouched” vertices (that are not adjacent to any of the k queried edges) are equally good candidates for the next vertex to be involved in the query set. For Boolean function testing the situation is more complicated because of the structure imposed by the Boolean hypercube; for example, if the two strings 0^n and 1^n have been queried so far,

then it is clearly not the case that all possible 3rd query strings are “created equal” in relation to these first two query strings.

A natural first effort to canonicalize a Boolean function property tester is to design a canonical tester that makes its queries in the first stage, and then simply directly uses those queries to “internally” simulate a run of the independent tester in its second stage. Ideally, in such an “internal” simulation, each time the original independent tester makes a query the canonical tester would use a query-response pair obtained in its first stage that corresponds reasonably well to the string queried by the independent tester. However, this naive approach does not seem to suffice, since an independent tester can easily make queries which do not correspond well to any query made by the canonical tester. As a simple example, the first query of an independent tester could independently set each variable x_i to 1 with probability $1/3$. The number of 1s in the first query of this independent tester is distributed as a draw from the binomial distribution $B(n, 1/3)$, but the number of 1s in any query made by a q -query canonical tester is distributed as a draw from the binomial distribution $B(n, p)$, where p is of the form (integer)/ 2^q . If q is a constant independent of n , these two distributions have variation distance nearly 1.

The high-level idea of both our constructions is that instead of trying to approximately simulate an execution of the independent tester on the n -variable function f (which it cannot do), the canonical tester *perfectly* simulates an execution of the independent tester on a different function f' over n' relevant variables. Since this simulation is perfect, the canonical tester successfully tests whether f' has property C . For the case of Noisy-Neg minors the analysis shows that w.h.p. the independent tester’s view looks the same whether the target function is f' or f . Therefore, a “good” answer for f' must also be a good answer for f . For the case of ID-Neg minors, the analysis shows that because of the way f' is determined, we have that (1) if f belongs to C then so does f' ; and (2) if f is far from C , then f' is at least slightly far from C . Along with the fact that the canonical tester tests f' successfully, these conditions imply that the canonical tester tests f successfully.

2 Preliminaries

A *Boolean function property* is simply a class of Boolean functions. Throughout the paper we write \mathcal{F}_n to denote the class of all 2^{2^n} Boolean functions mapping $\{0, 1\}^n$ to $\{0, 1\}$. We write C_n to denote a class of n -variable Boolean functions.

We adopt all the standard definitions of Boolean function property testing (see e.g. [PRS02](#), [EKR⁺04](#), [MORS10](#)) and do not repeat them here because of space limitations. As mentioned in the Introduction, we may view any nonadaptive testing algorithm T as consisting of two phases: an initial “query generation phase” T_1 in which the query strings are selected (at the end of this phase the queries are performed), and a subsequent “computation” phase T_2 in which some computation is performed on the query-answer pairs and the algorithm either accepts or rejects. Throughout the paper we will describe and analyze testing algorithms in these terms.

The Classes We Consider. Our first main result deals with classes of Boolean functions closed under Noisy-Neg minors; we give the relevant definitions below.

Definition 1. (Noise Sensitivity of f) Let $f : \{0, 1\}^n \rightarrow \{0, 1\}$, let $\epsilon \in [0, 1/2]$, and let (x, y) be a pair of $(1 - 2\epsilon)$ -correlated random inputs (i.e. x is uniform from $\{0, 1\}^n$ and y is formed by independently setting each y_i to equal x_i with probability $1 - 2\epsilon$, and to be uniform random otherwise). The noise sensitivity of f at noise rate ϵ is defined to be $NS_\epsilon(f) := \Pr[f(x) \neq f(y)]$.

(Noise Sensitivity of a class C) Let $C = \cup_{n \geq 1} C_n$ be a class of Boolean functions. We define $NS_\epsilon(C) := \max_n \max_{f \in C_n} \{NS_\epsilon(f)\}$, the noise sensitivity of C at noise rate ϵ , to be the maximum noise sensitivity of any $f \in C$.

(C is closed under Noisy-Neg minors) Let $C = \cup_{n \geq 1} C_n$ be a class of Boolean functions. We say that C is closed under Noisy-Neg Minors if C is closed under negating input variables and there is a function $g(\epsilon)$ (not depending on n) which is such that $\lim_{\epsilon \rightarrow 0^+} g(\epsilon) = 0$ and $NS_\epsilon(C) \leq g(\epsilon)$.

Our second main result deals with classes C closed under ID-Neg Minors.

Definition 2. (ID-Neg Minors) Let $f \in \mathcal{F}_n$ and let $f' \in \mathcal{F}_{n'}$. We say that f' is an ID-Neg Minor of f if f' can be produced from f by a (possibly empty) sequence of the following operations: (i) Adding/Removing irrelevant variables (recall that variable x_i is irrelevant if there is no input string where flipping x_i changes the value of f); (ii) Identifying input variables (e.g. the function $f(x_1, x_1, x_3)$ is obtained by identifying variable x_2 with x_1); and (iii) Negating input variables.

(C is closed under ID-Neg Minors) Let $C = \cup_{n \geq 1} C_n$ be a class of Boolean functions, let $f \in \mathcal{F}_n$, and let $f' \in \mathcal{F}_{n'}$. We say that C is closed under ID-Neg Minors if the following holds: If $f \in C_n$ and f' is an ID-Neg Minor of f , then $f' \in C$.

As examples, the class of $GF(2)$ degree- d polynomials is closed under ID-Neg minors, and the class of halfspaces is closed under Noisy-Neg minors.

We close this preliminaries section with two definitions that will be useful:

Definition 3. Let f be a function in \mathcal{F}_n and let F_+, F_- be two disjoint subsets of $[n]$. We define $\text{Noisy}(f, F_+, F_-) \in \mathcal{F}_n$ to be the function $\text{Noisy}(f, F_+, F_-)(x_1, \dots, x_n) = f(t_1, \dots, t_n)$, where $t_i := 1$ if $i \in F_+$, $t_i := 0$ if $i \in F_-$; and $t_i := x_i$ otherwise.

Intuitively, given a function f to test, our canonical tester for classes C that are closed under Noisy-Neg minors will choose F_+, F_- according to some distribution (defined later) and will instead test $f' = \text{Noisy}(f, F_+, F_-)$.

Definition 4. Let $f \in \mathcal{F}_n$, F_+ and F_- be two disjoint subsets of $[n]$, and id be an element of F_+ . For $n' = n - |F_+| - |F_-| + 1$, we define $\text{ID-Neg}(f, F_+, F_-, id) \in \mathcal{F}_{n'}$ to be the function $\text{ID-Neg}(f, F_+, F_-, id)(x_1, \dots, x_{n'}) = f(t_1, \dots, t_n)$, where $t_i := x_{id}$ if $i \in F_+$; $t_i := \bar{x}_{id}$ if $i \in F_-$; and $t_i := x_i$ otherwise.

Similarly to the case above, given a target function f our canonical tester for classes C that are closed under ID-Neg minors will choose F_+, F_-, id according to some distribution (defined later) and will instead test the target function $f' = \text{ID-Neg}(f, F_+, F_-, id)$.

2.1 The Testing Algorithms We Can Canonicalize: Independent Testers

Definition 5. A $q(\epsilon)$ -query independent tester for class C is a probabilistic oracle machine $T = (T_1, T_2)$ which takes as input a distance parameter ϵ and is given access to a black-box oracle for an arbitrary function $f : \{0, 1\}^n \rightarrow \{0, 1\}$.

(First Stage) The query generation algorithm T_1 chooses $q(\epsilon)$ query strings in the following way: To choose the i -th string, the algorithm partitions the set $[n]$ into 2^{i-1} blocks. The block $B_{b_1, \dots, b_{i-1}}$ contains those indices that were set to b_j in the j th query string x^j for all $j = 1, \dots, i - 1$. For each block $B_{b_1, \dots, b_{i-1}}$, for each $m \in B_{b_1, \dots, b_{i-1}}$, the algorithm sets x_m^i to 1 with probability p_{b_1, \dots, b_i} and to 0 with probability $1 - p_{b_1, \dots, b_i}$. The resulting string x^i is the i -th query string. After choosing all the strings, T_1 queries all $q(\epsilon)$ strings $x^1, \dots, x^{q(\epsilon)}$ and gets back responses $f(x^1), \dots, f(x^{q(\epsilon)})$.

(Second Stage) The computation stage T_2 gets as input the $q(\epsilon)$ query-answer pairs $(x_1, f(x_1)), \dots, (x_{q(\epsilon)}, f(x_{q(\epsilon)}))$, does some deterministic computation on this input, and outputs either “accept” or “reject.”

In an independent tester the query generation algorithm T_1 must satisfy the following conditions:

- For each string $b = (b_1, \dots, b_t)$ the probability $p_b = p_b(\epsilon)$ is a value $0 \leq p_b \leq 1$ (which may depend on ϵ but is independent of n).
- For each t , the 2^t values p_{b_1, \dots, b_t} (as b ranges over $\{0, 1\}^t$) are all rational numbers, and (over all t) the denominator of each of these rational numbers is at most $c = c(\epsilon)$ (c may depend on ϵ but is independent of n). We say that $c(\epsilon)$ is the granularity of the independent tester T .

If T is a one-sided tester then for any $f : \{0, 1\}^n \rightarrow \{0, 1\}$, if f belongs to C then $\Pr[T^f = \text{“accept”}] = 1$, and if f is ϵ -far from C then $\Pr[T^f = \text{“reject”}] \geq r(\epsilon)$, where $r(\epsilon) > 0$ is a positive-valued function of ϵ only. We say that $r(\epsilon)$ is the rejection parameter of the tester.

If T is a two-sided tester then for any $f : \{0, 1\}^n \rightarrow \{0, 1\}$, if f belongs to C then $\Pr[T^f = \text{“accept”}] = 1 - a(\epsilon)$, and if f is ϵ -far from C then $\Pr[T^f = \text{“reject”}] \geq r(\epsilon)$ where a and r are functions of ϵ only and for $0 < \epsilon < 1/2$, $a(\epsilon) < r(\epsilon)$. We say that $a(\epsilon)$ and $r(\epsilon)$ are the acceptance and rejection parameters of the tester respectively.

Given an independent tester T as described above, we let $\text{Prod}(\epsilon)$ denote the product of the denominators of all probabilities $p_{b_1, \dots, b_t}(\epsilon)$ where t ranges over all possible values $1, 2, \dots, q(\epsilon)$ and $b = (b_1, \dots, b_t)$ ranges over all t -bit strings. If the tester T is $c(\epsilon)$ -granular, it is easy to see that $\text{Prod}(\epsilon)$ is at most $c(\epsilon)^{2^{q(\epsilon)+1}}$. It is clear that each subset B_{b_1, \dots, b_t} of $[n]$ described above has size binomially distributed according to $B(n, \ell/\text{Prod}(\epsilon))$ for some integer ℓ .

3 A Canonical Form for Testers, and Our Main Results

Before stating our main results precisely, we give a precise description of the canonical form mentioned in the introduction.

Definition 6. Let $q' : [0, 1] \rightarrow \mathbb{N}$. A q' -canonical tester for class C is a probabilistic oracle machine $T = (T_1, T_2)$ which takes as input a distance parameter ϵ and is given access to a black-box oracle for an arbitrary function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, and performs as follows.

Given input parameter ϵ , the query generation algorithm T_1 works as follows.

1. $z^1, \dots, z^{q'(\epsilon)}$ are selected to be independent uniformly random n -bit strings. These strings define a partition \mathcal{B} of $[n]$ into $2^{q'(\epsilon)}$ blocks: an element $i \in [n]$ lies in block $B_{b_1, \dots, b_{q'(\epsilon)}}$ if the i -th bit of string z^j equals b_j for all $j = 1, \dots, q'(\epsilon)$.
2. Let $Q^{\mathcal{B}} \subseteq \{0, 1\}^n$ be the set of all strings x such that the following condition holds: $\forall i, j \in [n]$, if i and j are in the same partition subset $B_{b_1, \dots, b_{q'(\epsilon)}} \in \mathcal{B}$ then $x_i = x_j$.
3. Using the oracle for f , T_1 queries all $2^{2^{q'(\epsilon)}}$ strings $x \in Q^{\mathcal{B}}$.

The computation stage T_2 is given the $2^{2^{q'(\epsilon)}}$ query-answer pairs $[(x, f(x))]_{x \in Q^{\mathcal{B}}}$, does some deterministic computation, and outputs either “accept” or “reject.”

The success criteria for one-sided (two-sided, respectively) canonical testers are entirely similar to the criteria for independent testers. We note that a q' -canonical tester makes $2^{2^{q'(\epsilon)}}$ queries when run with input parameter ϵ .

3.1 Main Results

As our main results, we show that (i) any class that is closed under Noisy-Neg minors and is constant-query testable by a (two-sided) independent tester is also constant-query testable by a (two-sided) canonical tester; and (ii) any class that is closed under ID-Neg minors and is constant-query testable by a one-sided independent tester is also constant-query testable by a one-sided canonical tester. More precisely, we prove the following:

Theorem 3. Let C be any class of functions closed under Noisy-Neg Minors and let $g(\epsilon)$ be as in Definition 1. Let T be a $q(\epsilon)$ -query independent tester for property C with acceptance and rejection parameters $a(\epsilon), r(\epsilon)$. Let $q'_2(\epsilon)$ be the smallest integer value that satisfies the following bound:

$$NS_{\frac{\text{Prod}(\epsilon)}{2}, \frac{1}{2^{q'_2(\epsilon)}}}(C) \leq \frac{r(\epsilon) - a(\epsilon)}{16q(\epsilon)}.$$

Let $\eta' = \frac{2^{q'_2(\epsilon)} \bmod \text{Prod}(\epsilon)}{2^{q'_2(\epsilon)}}$ where Prod is as defined in Section 2.1 and let $q'_1(\epsilon) = \left\lceil \frac{32}{NS_{\eta'}(C)} \ln \frac{8}{r(\epsilon) - a(\epsilon)} \right\rceil$. Let $q'(\epsilon) = q'_2(\epsilon) \cdot (q'_1(\epsilon) + 1)$. Then there is a q' -canonical

tester $\text{Canon}(T)$ for C with acceptance and rejection parameters $a'(\epsilon), r'(\epsilon)$, where $a'(\epsilon) = \frac{3}{4}a(\epsilon) + \frac{1}{4}r(\epsilon)$, and $r'(\epsilon) = \frac{1}{4}a(\epsilon) + \frac{3}{4}r(\epsilon)$.

Theorem 4. *Let C be any class of functions closed under ID-Neg Minors. Let T be a one-sided independent tester for property C that has query complexity $q(\epsilon)$, granularity $c(\epsilon)$, and rejection parameter $r(\epsilon)$. Let $\epsilon_1 = \frac{r(\epsilon)}{4q(\epsilon)}$ and let $q'(\epsilon)$ be defined as $q'(\epsilon) = \lceil \log(\text{Prod}(\epsilon) \cdot \text{Prod}(\epsilon_1)) \rceil$ where Prod is as described in Section 2.1. Then there is a one-sided q' -canonical tester $\text{Canon}(T)$ for property C which, on input parameter ϵ , has rejection parameter $(\frac{3r(\epsilon)/4}{1-r(\epsilon)/4}) \cdot r(\epsilon_1)$.*

Throughout the rest of the paper whenever we write “ T ” or “ C ” without any other specification, we are referring to the tester and property from Theorem 3 or Theorem 4 (which one will be clear from context).

4 Overview of the Proofs of Theorems 3 and 4

In this section we give a high-level explanation of our arguments and of the constructions of our canonical testers. Full details and complete proofs of Theorems 3 and 4, as well as implications of our results for various specific function classes and testing algorithms, are given in the full version.

We first note that an execution of the Independent Tester $T = (T_1, T_2)$ (see Definition 5) with input parameter ϵ creates a $2^{q(\epsilon)}$ -way partition of the n variables by independently assigning each variable to a randomly chosen subset in the partition with the appropriate probability (of course these probabilities need not all be equal). All queries made by the independent tester then respect this partition.

Consider the following first attempt at making a canonical tester $\text{Canon}(T) = (\text{Canon}(T)_1, \text{Canon}(T)_2)$ from an independent tester T . First, $\text{Canon}(T)_1$ partitions the n variables into $2^{q'}$ subsets of expected size $n/2^{q'}$, as specified in Definition 6, and makes all corresponding queries; it then passes both the queries and the responses to the second stage, $\text{Canon}(T)_2$. The value q' will be such that $2^{q'}$ equals $\text{Prod}(\epsilon) \cdot k + \text{rem}$, where $0 \leq \text{rem} < \text{Prod}(\epsilon)$, and k is a positive integer. In the second stage, $\text{Canon}(T)_2$ chooses the first $\text{Prod}(\epsilon) \cdot k$ subsets of $\text{Canon}(T)_1$'s partition (let us say these subsets collectively contain n' variables) and ignores the variables in the last rem subsets. For the n' variables contained in these first $\text{Prod}(\epsilon) \cdot k$ subsets, $\text{Canon}(T)_2$ can perfectly simulate a partition created by an execution of the independent tester T on these n' variables with parameter ϵ , by “coalescing” these $\text{Prod}(\epsilon) \cdot k$ subsets into $2^{q'(\epsilon)}$ subsets of the appropriate expected sizes. (To create a subset whose size is binomially distributed according to $B(n', \ell/\text{Prod}(\epsilon))$, $\text{Canon}(T)_2$ “coalesces” a collection of $k\ell$ of the $\text{Prod}(\epsilon)$ subsets.) To simulate each of the $q = q(\epsilon)$ queries that T makes, $\text{Canon}(T)_2$ sets the n' variables as T_1 would set them given this partition.

Obviously, the problem with the above simulation is how to set the extra variables in the remaining rem subsets in each of the q queries. The n' variables described above are faithfully simulating the distribution over query strings that

T would make if it were run on an n' -variable function with input parameter ϵ , but of course the actual queries that $\text{Canon}(T)$ makes have the additional rem variables, and the corresponding responses are according to the n -variable function f . Thus, we have no guarantee that T_2 will answer correctly w.h.p. when executed on the query-response strings generated by the simulator $\text{Canon}(T)_2$ as described above. Nevertheless, the simulation described above is a good starting point for our actual construction.

The underlying idea of our canonical testers is that instead of (imperfectly) simulating an execution of the independent tester on the actual n -variable target function f , the canonical tester *perfectly* simulates an execution of the independent tester on a related function f' . Our analysis shows that due to the special properties of the independent tester and of the classes we consider, the response of the independent tester on function f' is also a legitimate response for f .

Below we describe the construction of a canonical tester for two different types of independent testers and classes. The first construction shows how to transform T , where T is a two-sided independent tester for a class C that is closed under Noisy-Neg Minors, into $\text{Canon}(T)$, a two-sided canonical tester for class C . The second construction shows how to transform T , where T is a one-sided independent tester for a class C closed under ID-Neg minors, into $\text{Canon}(T)$, a one-sided canonical tester for C .

4.1 Two-Sided Independent Testers and Classes Closed under Noisy-Neg Minors

We first note that it is easy to construct an algorithm that approximates $NS_\eta(f)$ of a target function f by non-adaptively drawing pairs of points (x, y) where x is chosen uniformly at random and y is $1 - 2\eta$ correlated with x . It is also easy to see that if η is a rational number c_1/c_2 where c_2 is a power of 2, then the distribution over queries made by such an algorithm can be simulated using a canonical tester.

For ease of understanding we view our first canonical tester as having two parts (it will be clear that these two parts can be straightforwardly combined to obtain a canonical tester that follows the template of Definition 6). The first part is an algorithm that approximates $NS_{\eta'}(f)$ and rejects any f for which $NS_{\eta'}(f)$ is noticeably higher than $NS_{\eta'}(C)$ (here η' is a parameter of the form (integer)/(power of 2) that will be specified later).

The second part of the tester simulates the partition generated by the independent tester T as described at the start of Section 4. Let F_+ contain the variables assigned to the first $\text{rem}/2$ subsets from the rem “remaining” subsets, and let F_- contain the variables assigned to the last $\text{rem}/2$ of those subsets. As a thought experiment, we may imagine that the variables in $F_+ \cup F_-$ are each independently assigned to a randomly selected one of the $\text{Prod}(\epsilon)$ partition subsets with the appropriate probability. In this thought experiment, we have perfectly simulated a partition generated by running T_1 over an n -variable function. We now define f' based on the subsets F_+ and F_- . The function f' is

simply the restriction of f under which all variables in F_+ are fixed to 1 and all variables in F_- are fixed to 0.

Now, we would like $\text{Canon}(T)$ to generate the q query-answer pairs for f that T_1 would make given the partition from the thought experiment described above. While $\text{Canon}(T)$ cannot do this, a crucial observation is that $\text{Canon}(T)$ can perfectly simulate q query-answer pairs that T_1 would make given the above-described partition where the answers are generated according to f' . Moreover, our analysis will show (using the fact that C is closed under negation) that we may assume w.l.o.g. that each of these q queries is individually uniformly distributed over $\{0, 1\}^n$.

Thus for each individual (uniform random) query string x , $f'(x)$ is equivalent to $f(y)$ where y is a random string that is $(1 - 2\eta')$ -correlated with x , where $\eta' = \text{rem}/2^{q'}$. Now since $NS_{\eta'}(C)$ depends only on η' , by choosing η' small enough (and q' large enough), by a union bound with high probability $f(x)$ equals $f'(x)$ for all the queries x that were generated. Since this is the case, then T_2 must with high probability generate the same output on target function f and f' . So since T is (by hypothesis) an effective tester for f it must be the case that T 's responses on f' are also "good" for f .

4.2 One-Sided Independent Testers and Classes Closed under ID-Neg Minors

Our second canonical tester construction also begins by simulating a partition of the independent tester T over n' variables as described above. However, now we will think of the parameters as being set somewhat differently: we view the canonical tester as partitioning the n variables into $2^{q'(\epsilon)}$ subsets where now $q' = q'(\epsilon)$ is such that $2^{q'}$ equals $\text{Prod}(\epsilon_1) \cdot k + \text{rem}$, where $\epsilon_1 \ll \epsilon$ (more precisely $\epsilon_1 = \frac{r(\epsilon)}{4q(\epsilon)}$, though this exact expression is not important for now), $0 \leq \text{rem} < \text{Prod}(\epsilon)$, and k is a positive integer. The canonical tester then defines a new function f' over n' variables by applying an operator \mathcal{F}^{ϵ_1} to the set X of $2^{2^{q'(\epsilon)}}$ query strings that $\text{Canon}(T)_1$ generates; we now describe how this operator acts. Let F_+ contain the variables assigned to the first $\text{rem}/2$ subsets from the rem "remaining" subsets, and let F_- contain the variables assigned to the last $\text{rem}/2$ of the rem subsets. Given f , the function f' obtained by applying \mathcal{F}^{ϵ_1} to X is chosen in the following way: f' is the same as f except that

- A variable x_{id} is chosen by taking the lexicographically first element of F_+ ;
- All variables in F_+ (F_-) are identified with x_{id} (\bar{x}_{id}).

$\text{Canon}(T)$ places id at random in one of the remaining partition subsets and then selects the appropriate set of query strings that T_1 would make given the simulated partition over n' variables described above, and constructs query-answer pairs for these strings in which the answers are the corresponding values of f' on these strings (similar to the key observation in Section 4.1, it is indeed possible for $\text{Canon}(T)$ to do this). Finally, $\text{Canon}(T)$ passes these queries and responses to T_2 and responds as T_2 does.

The proof of correctness proceeds in two parts. First, we show that with high probability $\text{Canon}(T)$ successfully tests target function f' . (This is an easy consequence of the fact, mentioned above, that $\text{Canon}(T)$ perfectly simulates T 's partition over the n' variables.) Second, we show that (1) if $f \in C$ and C is closed under ID-Neg minors then $f' \in C$; and (2) if f is ϵ -far from C then w.h.p. f' is ϵ_1 -far from C , where the value of ϵ_1 depends only on ϵ . We note that (2) does not hold in general for f, f' where f' is an arbitrary ID-Neg minor of f . However, our analysis shows that assuming that there exists a one-sided independent tester for class C , (2) holds for f' chosen in the way described above.

References

- [AKK⁺03] Alon, N., Kaufman, T., Krivelevich, M., Litsyn, S., Ron, D.: Testing low-degree polynomials over $\text{GF}(2)$. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) *RANDOM 2003 and APPROX 2003*. LNCS, vol. 2764, pp. 188–199. Springer, Heidelberg (2003)
- [Bla09] Blais, E.: Testing juntas nearly optimally. In: *Proc. 41st Annual ACM Symposium on Theory of Computing (STOC)*, pp. 151–158 (2009)
- [Bla10] Blais, E.: Personal communication (2010)
- [BLR93] Blum, M., Luby, M., Rubinfeld, R.: Self-testing/correcting with applications to numerical problems. *J. Comp. Sys. Sci.* 47, 549–595 (1993)
- [DLM⁺07] Diakonikolas, I., Lee, H., Matulef, K., Onak, K., Rubinfeld, R., Servedio, R., Wan, A.: Testing for concise representations. In: *Proc. 48th Ann. Symposium on Computer Science (FOCS)*, pp. 549–558 (2007)
- [FKR⁺04] Fischer, E., Kindler, G., Ron, D., Safra, S., Samorodnitsky, A.: Testing juntas. *J. Computer & System Sciences* 68(4), 753–787 (2004)
- [GOS⁺09] Gopalan, P., O'Donnell, R., Servedio, R., Shpilka, A., Wimmer, K.: Testing Fourier dimensionality and sparsity. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikolettseas, S., Thomas, W. (eds.) *ICALP 2009*. LNCS, vol. 5555, pp. 500–512. Springer, Heidelberg (2009)
- [GT03] Goldreich, O., Trevisan, L.: Three theorems regarding testing graph properties. *Random Structures and Algorithms* 23(1), 23–57 (2003)
- [HR05] Hellerstein, L., Raghavan, V.: Exact learning of DNF formulas using DNF hypotheses. *J. Comp. Syst. Sci.* 70(4), 435–470 (2005)
- [MORS10] Matulef, K., O'Donnell, R., Rubinfeld, R., Servedio, R.: Testing halfspaces. *SIAM J. Comp.* 39(5), 2004–2047 (2010)
- [PRS02] Parnas, M., Ron, D., Samorodnitsky, A.: Testing Basic Boolean Formulae. *SIAM J. Disc. Math.* 16, 20–46 (2002)

Independent Sets in Random Graphs from the Weighted Second Moment Method

Varsha Dani¹ and Cristopher Moore^{1,2}

¹ Computer Science Department, University of New Mexico
² Santa Fe Institute

Abstract. We prove new lower bounds on the likely size of the maximum independent set in a random graph with a given constant average degree. Our method is a weighted version of the second moment method, where we give each independent set a weight based on the total degree of its vertices.

1 Introduction

We are interested in the likely size of the largest independent set S in a random graph with a given average degree. It is easy to see that $|S| = \Theta(n)$ whenever the average degree is constant, and Shamir and Spencer [8] showed using Azuma's inequality that, for any fixed n , $|S|$ is tightly concentrated around its mean. Moreover, Bayati, Gamarnik, and Tetali [3] recently showed that $|S|/n$ converges to a limit with high probability. Thus for each constant c there is a constant $\alpha_{\text{crit}} = \alpha_{\text{crit}}(c)$ such that

$$\lim_{n \rightarrow \infty} \Pr[G(n, p = c/n) \text{ has an independent set of size } \alpha n] = \begin{cases} 1 & \alpha < \alpha_{\text{crit}} \\ 0 & \alpha > \alpha_{\text{crit}} \end{cases}.$$

By standard arguments this holds in $G(n, m = cn/2)$ as well.

Our goal is to bound α_{crit} as a function of c , or equivalently to bound

$$c_{\text{crit}} = \sup \{c : \alpha_{\text{crit}}(c) \geq \alpha\},$$

as a function of α . For $c \leq e$, a greedy algorithm of Karp and Sipser [7] asymptotically finds a maximal independent set, and analyzing this algorithm with differential equations yields the exact value of α_{crit} . For larger c , Frieze [6] determined α_{crit} to within $o(1/c)$, where o refers to the limit where c is large. These bounds were improved by Coja-Oghlan and Efthymiou [5] who prove detailed results on the structure of the set of independent sets.

We further improve these bounds. Our method is a weighted version of the second moment method, inspired by the work of Achlioptas and Peres [2] on random k -SAT, where each independent set is given a weight depending on the total degree of its vertices. In addition to improving bounds on this particular problem, our hope is that this advances the art and science of inventing random variables that counteract local sources of correlation in random structures.

We work in a modified version of the $G(n, m)$ model which we call $\tilde{G}(n, m)$. For each of the m edges, we choose two vertices u, v uniformly and independently and connect them. This may lead to a few multiple edges or self-loops. A vertex with a self-loop cannot belong to an independent set. In the sparse case where $m = cn/2$ for constant c , with constant positive probability $\tilde{G}(n, m = cn/2)$ has no multiple edges or self-loops, in which case it is uniform in the usual model $G(n, m = cn/2)$ where edges are chosen without replacement from distinct pairs of vertices. Thus any property which holds with high probability for $\tilde{G}(n, m)$ also holds with high probability for $G(n, m)$, and any bounds we prove on α_{crit} in $\tilde{G}(n, m)$ also hold in $G(n, m)$.

We review the first moment upper bound on α_{crit} from Bollobás [4]. Let X denote the number of independent sets of size αn in $\tilde{G}(n, m)$. Then

$$\Pr[X > 0] \leq \mathbb{E}[X].$$

By linearity of expectation, $\mathbb{E}[X]$ is the sum over all $\binom{n}{\alpha n}$ sets of αn vertices of the probability that a given one is independent. The m edges (u, v) are chosen independently and for each one $u, v \in S$ with probability α^2 , so

$$\mathbb{E}[X] = \binom{n}{\alpha n} (1 - \alpha^2)^m.$$

In the limit $n \rightarrow \infty$, Stirling’s approximation $n! = (1 + o(1))\sqrt{2\pi n} n^n e^{-n}$ gives

$$\binom{n}{\alpha n} = \Theta\left(\frac{1}{\sqrt{n}} e^{nh(\alpha)}\right), \tag{1}$$

where h is the entropy function

$$h(\alpha) = -\alpha \ln \alpha - (1 - \alpha) \ln(1 - \alpha),$$

and where Θ hides constants that depend smoothly on α . Thus

$$\mathbb{E}[X] = \Theta\left(\frac{1}{\sqrt{n}} e^{n(h(\alpha) + (c/2) \ln(1 - \alpha^2))}\right).$$

For each c , the α such that

$$h(\alpha) + (c/2) \ln(1 - \alpha^2) = 0 \tag{2}$$

is an upper bound on $\alpha_{\text{crit}}(c)$, since for larger α the expectation $\mathbb{E}[X]$ is exponentially small.

We find it more convenient to parametrize our bounds in terms of the function $c_{\text{crit}}(\alpha)$. Then [2] gives the following upper bound,

$$c_{\text{crit}}(\alpha) \leq 2 \frac{\alpha \ln \alpha + (1 - \alpha) \ln(1 - \alpha)}{\ln(1 - \alpha^2)} \leq 2 \frac{\ln(1/\alpha) + 1}{\alpha}. \tag{3}$$

We will prove the following nearly-matching lower bound.

Theorem 1.1. *For any constant $x > 4/e$, for sufficiently small α*

$$c_{\text{crit}}(\alpha) \geq 2 \frac{\ln(1/\alpha) + 1}{\alpha} - \frac{x}{\sqrt{\alpha}}. \tag{4}$$

Coja-Oghlan and Efthymiou [5] bounded c_{crit} within a slightly larger factor $O(\sqrt{\ln(1/\alpha)/\alpha})$.

Inverting (3) and Theorem 1.1 gives the following bounds on $\alpha_{\text{crit}}(c)$. The lower bound is a significant improvement over previous results:

Corollary 1.2. *For $z > 0$, let $W(z)$ denote the unique positive root x of the equation $xe^x = z$. Then for any constant $y > 4\sqrt{2}/e$,*

$$\frac{2}{c} W\left(\frac{ec}{2}\right) - y \frac{\sqrt{\ln c}}{c^{3/2}} \leq \alpha_{\text{crit}} \leq \frac{2}{c} W\left(\frac{ec}{2}\right),$$

where the lower bound holds for sufficiently large c .

If we like we can expand $W(ec/2)$ asymptotically in c ,

$$\begin{aligned} W\left(\frac{ec}{2}\right) &= \ln c - \ln \ln c + 1 - \ln 2 + \frac{\ln \ln c}{\ln c} - \frac{1 - \ln 2}{\ln c} \\ &\quad + \frac{1}{2} \frac{(\ln \ln c)^2}{(\ln c)^2} - (2 - \ln 2) \frac{\ln \ln c}{(\ln c)^2} + \frac{3 + (\ln 2)^2 - 4 \ln 2}{2(\ln c)^2} + O\left(\frac{(\ln \ln c)^3}{(\ln c)^3}\right). \end{aligned}$$

The first few of these terms correspond to the bound in [6], and we can extract as many additional terms as we wish.

2 The Weighted Second Moment Method

Our proof uses the second moment method. For any nonnegative random variable X , the Cauchy-Schwarz inequality implies that

$$\Pr[X > 0] \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]}. \tag{5}$$

If X counts the number of objects of a certain kind, (5) shows that at least one such object exists as long as the expected number of objects is large and the variance is not too large.

Unfortunately, applying this directly to the number X of independent sets fails utterly. The problem is that for most pairs of sets of size αn , the events that they are independent are highly correlated, unlike the case where the average degree grows sufficiently quickly with n [4, 6]. As a result, $\mathbb{E}[X^2]$ is exponentially larger than $\mathbb{E}[X]^2$, and the second moment method yields an exponentially small lower bound on $\Pr[X > 0]$.

One way to deal with these correlations, used by Frieze in [6], is to partition the vertices into sets V_i of $\lfloor 1/\alpha \rfloor$ vertices each and focus on those independent

sets that intersect each V_i exactly once. In that case, a large-deviations inequality allows us to override the correlations.

Here we pursue a different approach, inspired by the work of Achlioptas and Peres [2] on random k -SAT. The idea is to give each independent set S a weight $w(S)$, depending exponentially on local quantities in the graph. Specifically, we define

$$w(S) = \mu^{\# \text{ of edges } (u, v) \text{ with } u, v \notin S},$$

for some $\mu < 1$. If the number of edges m is fixed, the number of edges where neither endpoint is in S is simply m minus the total degree of the vertices in S . Thus we can also write

$$w(S) = \mu^{m - \sum_{v \in S} \deg(v)}. \tag{6}$$

We will apply the second moment method to the total weight of all independent sets of size αn ,

$$X = \sum_{\substack{S \subseteq V, |S| = \alpha n \\ S \text{ independent}}} w(S).$$

If we tune μ properly, then for particular α^*, c^* we have $\mathbb{E}[X^2] = \Theta(\mathbb{E}[X]^2)$, in which case $\Pr[X > 0]$ is bounded above zero. In that case $c_{\text{crit}}(\alpha^*) \geq c^*$, or equivalently $\alpha_{\text{crit}}(c^*) \geq \alpha^*$.

Why is this the right type of weight? Intuitively, one of the main sources of correlations between independent sets is the temptation to occupy low-degree vertices. For instance, any two maximal independent sets contain all the degree-zero vertices, giving them a large overlap. If X simply counts the independent sets of size αn , the resulting correlations make X 's variance exponentially large compared to the square of its expectation, and the second moment fails.

Weighting each S as in (6) counteracts this temptation, punishing sets that occupy low-degree vertices by reducing their weight exponentially. As we will see below, when μ is tuned to a particular value, making this punishment condign, these correlations disappear in the sense that the dominant contribution to $\mathbb{E}[X^2]$ comes from pairs of sets S, T of size αn such that $|S \cap T| = \alpha^2 n + O(\sqrt{n})$, just as if S and T were chosen independently from among all sets of size αn .

This is analogous to the situation for k -SAT, where satisfying assignments are correlated because of the temptation to give each variable the truth value that agrees with the majority of its literals in the formula. By giving each satisfying assignment a weight $\eta^{\# \text{ of true literals}}$ and tuning η properly, we make the dominant contribution to $\mathbb{E}[X^2]$ come from pairs of satisfying assignments which agree on $n/2 + O(\sqrt{n})$ variables, just as if they were chosen independently [2].

Proceeding, let us compute the first and second moments of our random variable X . We extend the weight function $w(S)$ to all sets $S \subseteq V$ by setting $w(S) = 0$ if S is not independent. That is,

$$X = \sum_{\substack{S \subseteq V \\ |S| = \alpha n}} w(S)$$

where

$$w(S) = \prod_{(u,v) \in E} w_{u,v}(S)$$

and

$$w_{u,v}(S) = \begin{cases} \mu & \text{if } u, v \notin S \\ 1 & \text{if } u \in S, v \notin S \text{ or vice versa} \\ 0 & \text{if } u, v \in S. \end{cases} \tag{7}$$

We start by computing $\mathbb{E}[X]$. Fix a set S of size αn . Since the m edges are chosen independently,

$$\mathbb{E}[w(S)] = w_1(\alpha, \mu)^m \quad \text{where} \quad w_1(\alpha, \mu) = \mathbb{E}_{u,v}[w_{u,v}(S)].$$

For each edge (u, v) in $\tilde{G}(n, m)$, u and v are chosen randomly and independently, so the probabilities of the three cases in (7) are $(1 - \alpha)^2$, $2\alpha(1 - \alpha)$, and α^2 respectively. Thus

$$w_1(\alpha, \mu) = (1 - \alpha)^2 \mu + 2\alpha(1 - \alpha).$$

By linearity of expectation,

$$\mathbb{E}[X] = \sum_{\substack{S \subseteq V \\ |S| = \alpha n}} \mathbb{E}[w(S)] = \binom{n}{\alpha n} w_1(\alpha, \mu)^m.$$

Using Stirling’s approximation (II) and substituting $m = cn/2$ gives

$$\mathbb{E}[X] = \Theta\left(\frac{1}{\sqrt{n}} e^{nf_1(\alpha)}\right) \quad \text{where} \quad f_1(\alpha) = h(\alpha) + \frac{c}{2} \ln w_1(\alpha, \mu). \tag{8}$$

As before, Θ hides constant factors that depend smoothly on α .

Next we compute the second moment. We have

$$\mathbb{E}[X^2] = \mathbb{E}\left[\sum_S w(S) \sum_T w(T)\right] = \sum_{S,T} \mathbb{E}[w(S) w(T)]$$

where S and T are subsets of V of size αn . The expectation of $w(S) w(T)$ does not depend on the specific choice of S and T , but it does depend on the size of their intersection. We say that S and T have *overlap* ζ if $|S \cap T| = \zeta n$. Again using the independence of the edges, we have

$$\mathbb{E}[w(S) w(T)] = w_2(\alpha, \zeta, \mu)^m \quad \text{where} \quad w_2(\alpha, \zeta, \mu) = \mathbb{E}_{u,v}[w_{u,v}(S) w_{u,v}(T)].$$

For each edge (u, v) of \tilde{G} , the probability that it has no endpoints in S or T is $(1 - 2\alpha + \zeta)^2$, in which case it contributes μ^2 to $w_{u,v}(S) w_{u,v}(T)$. The probability that it has one endpoint in S and none in T or vice versa is $2(2\alpha - 2\zeta)(1 - 2\alpha + \zeta)$, in which case it contributes μ . Finally, the probability that it has one endpoint

in S and one in T is $2(\alpha - \zeta)^2 + 2\zeta(1 - 2\alpha + \zeta)$, in which case it contributes 1. With the remaining probability it has both endpoints in S or T , in causing them to be non-independent and contributing zero. Thus

$$w_2(\alpha, \zeta, \mu) = (1 - 2\alpha + \zeta)^2 \mu^2 + 4(\alpha - \zeta)(1 - 2\alpha + \zeta)\mu + 2(\alpha - \zeta)^2 + 2\zeta(1 - 2\alpha + \zeta)$$

Observe that when $\zeta = \alpha^2$, as it typically would be if S and T were chosen independently and uniformly, we have

$$w_2 = w_1^2. \tag{9}$$

The number of pairs of sets S, T of size αn and intersection of size $z = \zeta n$ is the multinomial

$$\binom{n}{\zeta n, (\alpha - \zeta)n, (\alpha - \zeta)n, (1 - 2\alpha + \zeta)n} = \binom{n}{\alpha n} \binom{\alpha n}{\zeta n} \binom{(1 - \alpha)n}{(\alpha - \zeta)n},$$

and linearity of expectation gives

$$\mathbb{E}[X^2] = \sum_{z=0}^{\alpha n} \binom{n}{z, \alpha n - z, \alpha n - z, (1 - 2\alpha)n + z} w_2(\alpha, \zeta, \mu)^m.$$

This sum is dominated by the terms where $\zeta = z/n$ is bounded inside the interval $(0, \alpha)$. Stirling’s approximation then gives

$$\binom{n}{\zeta n, (\alpha - \zeta)n, (\alpha - \zeta)n, (1 - 2\alpha + \zeta)n} = \Theta\left(\frac{e^{n[h(\alpha) + \alpha h(\zeta/\alpha) + (1 - \alpha)h(\frac{\alpha - \zeta}{1 - \alpha})]}}{n^{3/2}}\right),$$

where Θ hides constants that vary slowly with α and ζ . Thus the contribution to $\mathbb{E}[X^2]$ of pairs of sets with overlap $\zeta \in (0, \alpha)$ is

$$\frac{1}{n^{3/2}} e^{nf_2(\alpha, \zeta, \mu)} \tag{10}$$

where

$$f_2(\alpha, \zeta, \mu) = h(\alpha) + \alpha h\left(\frac{\zeta}{\alpha}\right) + (1 - \alpha)h\left(\frac{\alpha - \zeta}{1 - \alpha}\right) + \frac{c}{2} \ln w_2(\alpha, \zeta, \mu).$$

Combining (10) with (8), we can write

$$\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2} = \Theta\left(\frac{1}{\sqrt{n}} \sum_{z=0}^{\alpha n} e^{n\phi(z/n)}\right), \tag{11}$$

where

$$\begin{aligned} \phi(\zeta) &= f_2(\alpha, \zeta, \mu) - 2f_1(\alpha, \mu) \\ &= \alpha h\left(\frac{\zeta}{\alpha}\right) + (1 - \alpha)h\left(\frac{\alpha - \zeta}{1 - \alpha}\right) - h(\alpha) + \frac{c}{2} \ln \frac{w_2(\alpha, \zeta, \mu)}{w_1(\alpha, \mu)^2}. \end{aligned}$$

Using (9) and the fact that the entropy terms cancel, we have

$$\phi(\alpha^2) = 0.$$

In other words, the contribution to $\mathbb{E}[X^2]$ from pairs of sets with overlap α^2 is proportional to $\mathbb{E}[X]^2$.

We can now replace the sum in (11) with an integral,

$$\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2} = \Theta\left(\frac{1}{\sqrt{n}} \sum_{z=0}^{\alpha n} e^{n\phi(z/n)}\right) = \Theta\left(\sqrt{n} \int_0^\alpha e^{n\phi(\zeta)} d\zeta\right),$$

and evaluate this integral using Laplace’s method as in [1, Lemma 3]. Its asymptotic behavior depends on the maximum value of ϕ ,

$$\phi_{\max} = \max_{\zeta \in [0, \alpha]} \phi(\zeta).$$

If $\phi'' < 0$ at the corresponding ζ_{\max} , then it is dominated by an interval of width $\Theta(1/\sqrt{n})$ around ζ_{\max} and

$$\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2} = \Theta(e^{n\phi_{\max}}).$$

If $\phi_{\max} = \phi(\alpha^2) = 0$, then $\mathbb{E}[X^2] = \Theta(\mathbb{E}[X]^2)$ and the second moment method succeeds. Thus our goal is to show that ϕ is maximized at α^2 .

For this to happen, we at least need $\zeta = \alpha^2$ to be a local maximum of ϕ . In particular, we need

$$\phi'(\alpha^2) = 0. \tag{12}$$

Differentiating, we find that (12) holds if

$$\mu = \frac{1 - 2\alpha}{1 - \alpha}.$$

Henceforth, we will fix μ to this value. In that case we have

$$w_1 = 1 - \alpha \quad \text{and} \quad w_2 = (1 - \alpha)^2 + \frac{(\zeta - \alpha^2)^2}{(1 - \alpha)^2},$$

so

$$\phi(\zeta) = \alpha h\left(\frac{\zeta}{\alpha}\right) + (1 - \alpha) h\left(\frac{\alpha - \zeta}{1 - \alpha}\right) - h(\alpha) + \frac{c}{2} \ln\left(1 + \frac{(\zeta - \alpha^2)^2}{(1 - \alpha)^4}\right)$$

The remainder of this paper is dedicated to showing that for sufficiently small α as a function of c or vice versa, ϕ is indeed maximized at α^2 .

3 Finding and Bounding the Maxima

Using $\ln(1+x) \leq x$, we write $\phi(\zeta) \leq \psi(\zeta)$ where

$$\psi(\zeta) = \alpha h\left(\frac{\zeta}{\alpha}\right) + (1-\alpha)h\left(\frac{\alpha-\zeta}{1-\alpha}\right) - h(\alpha) + \frac{c}{2} \frac{(\zeta - \alpha^2)^2}{(1-\alpha)^4}. \tag{13}$$

Note that

$$\psi(\alpha^2) = \phi(\alpha^2) = 0.$$

Our goal is to show for an appropriate c that $\zeta = \alpha^2$ is in fact the global maximum of ψ , and therefore of ϕ . In what follows, asymptotic symbols such as O and o refer to the limit $\alpha \rightarrow 0$, or equivalently the limit $c \rightarrow \infty$. Error terms may be positive or negative unless otherwise stated.

The first two derivatives of $\psi(\zeta)$ are

$$\psi'(\zeta) = \frac{c(\zeta - \alpha^2)}{(1-\alpha)^4} + 2\ln(\alpha - \zeta) - \ln \zeta - \ln(1 - 2\alpha + \zeta) \tag{14}$$

$$\psi''(\zeta) = \frac{c}{(1-\alpha)^4} - \frac{2}{\alpha - \zeta} - \frac{1}{\zeta} - \frac{1}{1 - 2\alpha + \zeta} \tag{15}$$

The second derivative $\psi''(\zeta)$ tends to $-\infty$ at $\zeta = 0$ and $\zeta = \alpha$. Setting $\psi''(\zeta) = 0$ yields a cubic equation in ζ which has one negative root and, for sufficiently small α , two positive roots in the interval $[0, \alpha]$. Thus for each α and sufficiently small α , there are $0 < \zeta_1 < \zeta_2 < \alpha$ where

$$\psi''(\zeta) \begin{cases} < 0 & 0 < \zeta < \zeta_1 \\ > 0 & \zeta_1 < \zeta < \zeta_2 \\ < 0 & \zeta_2 < \zeta < \alpha. \end{cases}$$

It follows that ψ can have at most two local maxima. One is in the interval $[0, \zeta_1]$, and the following lemma shows that for the relevant α and c this is α^2 :

Lemma 3.1. *If $c = o(1/\alpha^2)$ then for sufficiently small α , $\zeta_1 > \alpha^2$ and $\psi(\alpha^2)$ is a local maximum.*

The other local maximum is in the interval $[\zeta_2, \alpha]$, and we denote it ζ_3 . To locate it, first we bound ζ_2 :

Lemma 3.2. *If*

$$c = (2 + o(1)) \frac{\ln(1/\alpha)}{\alpha},$$

then

$$\frac{\zeta_2}{\alpha} = 1 - \delta_2 \quad \text{where} \quad \delta_2 = \frac{1 + o(1)}{\ln(1/\alpha)}.$$

Thus ζ_2/α , and therefore ζ_3/α , tends toward 1 as $\alpha \rightarrow 0$.

We can now locate ζ_3 when α is close to its critical value.

Lemma 3.3. *If*

$$c = \frac{1}{\alpha} (2 \ln(1/\alpha) + 2 - o(1)),$$

then

$$\frac{\zeta_3}{\alpha} = 1 - \delta_3 \quad \text{where} \quad \delta_3 = \frac{1 + o(1)}{e} \sqrt{\alpha}.$$

Lemma 3.4. *For any constant $x > 4/e$, if*

$$c = \frac{2 \ln(1/\alpha) + 2 - x\sqrt{\alpha}}{\alpha},$$

then $\psi(\zeta_3) < 0$ for sufficiently small α .

4 Proofs

Proof of Lemma 3.1. Setting $\zeta = \alpha^2$ in (I5) gives

$$\psi''(\alpha^2) < \frac{c}{(1 - \alpha)^4} - \frac{1}{\alpha^2}.$$

If $c = o(1/\alpha^2)$ this is negative for sufficiently small α , in which case $\zeta_1 > \alpha^2$ and $\psi(\alpha^2)$ is a local maximum. \square

Proof of Lemma 3.2. For any constant b , if

$$\frac{\zeta}{\alpha} = 1 - \delta \quad \text{where} \quad \delta = \frac{b}{\ln(1/\alpha)} \tag{16}$$

then (I5) gives

$$\psi''(\zeta) = \left(2 - \frac{2}{b} + o(1)\right) \frac{\ln(1/\alpha)}{\alpha} - O(1/\alpha).$$

If $b \neq 1$, for sufficiently small α this is negative if $b < 1$ and positive if $b > 1$. Therefore $\zeta_2/\alpha = 1 - \delta_3$ where $\delta_3 = (1 + o(1))/\ln(1/\alpha)$. \square

Proof of Lemma 3.3. Lemma 3.2 tells us that $\zeta_3 = \alpha(1 - \delta)$ for some

$$\delta < \frac{1 + o(1)}{\ln(1/\alpha)}.$$

Setting $\zeta = \alpha(1 - \delta)$ in (I4) and using

$$\frac{1}{(1 - \alpha)^4} = 1 + O(\alpha) \quad \text{and} \quad -\ln(1 - x) = O(x)$$

gives, after some algebra,

$$\psi'(\zeta) = \alpha c + \ln \alpha + 2 \ln \delta + O(\alpha \delta c) + O(\alpha^2 c).$$

For any constant b , setting

$$\delta = \frac{b\sqrt{\alpha}}{e}$$

gives

$$\psi'(\zeta) = \alpha c + 2 \ln \alpha + 2 \ln b - 2 + O(\alpha^{3/2}c),$$

and setting

$$c = \frac{2 \ln(1/\alpha) + 2 - \varepsilon}{\alpha}$$

then gives

$$\psi'(\zeta) = 2 \ln b - \varepsilon + o(1).$$

If $\varepsilon = o(1)$ and $b \neq 1$, for sufficiently small α this is negative if $b < 1$ and positive if $b > 1$. Therefore $\zeta_3/\alpha = 1 - \delta_3$ where $\delta_3 = (1 + o(1))\sqrt{a}/e$. \square

Proof of Lemma 3.4. Setting $\zeta = \alpha(1 - \delta)$ where $\delta = b\sqrt{a}/e$ in (13) and using the Taylor series

$$\frac{1}{(1 - \alpha)^4} = 1 + 4\alpha + O(\alpha^2) \quad \text{and} \quad -\ln(1 - x) = x + x^2/2 + O(x^3)$$

gives, after a fair amount of algebra,

$$\begin{aligned} \psi(\zeta) &= \alpha(\ln \alpha - 1) - \left(\frac{2b \ln \alpha - 4b + 2b \ln b}{e}\right) \alpha^{3/2} + \left(\frac{c + 1}{2} - \frac{b^2}{2e^2}\right) \alpha^2 \\ &\quad - \frac{b}{e} \alpha^{5/2} c + \left(\frac{b^2}{2e^2} + 1\right) \alpha^3 c + O(\alpha^{7/2}c) + O(\alpha^{5/2}). \end{aligned}$$

Setting

$$c = \frac{2 \ln(1/\alpha) + 2 - x\sqrt{\alpha}}{\alpha}$$

for constant x causes the terms proportional to $\alpha \ln \alpha$, α , and $\alpha^{3/2} \ln \alpha$ to cancel, leaving

$$\psi(\zeta) = \left(\frac{2b(1 - \ln b)}{e} - \frac{x}{2}\right) \alpha^{3/2} + O(\alpha^2).$$

The coefficient of $\alpha^{3/2}$ is maximized when $b = 1$, and is negative whenever $x > 4/e$. In that case, $\psi(\zeta_3) < 0$ for sufficiently small α , completing the proof. \square

Proof of Corollary 1.2. First note that

$$\alpha_0 = \frac{2}{c} W\left(\frac{ec}{2}\right) \tag{17}$$

is the root of the equation

$$c = 2 \frac{\ln(1/\alpha_0) + 1}{\alpha_0},$$

since we can also write it as

$$e^{c\alpha/2} = \frac{e}{\alpha_0},$$

and multiplying both sides by $c\alpha_0/2$ gives

$$\frac{c\alpha_0}{2} e^{c\alpha_0/2} = \frac{ec}{2},$$

in which case (17) follows from the definition of W .

The root α of

$$c = 2 \frac{\ln(1/\alpha) + 1}{\alpha} - \frac{x}{\sqrt{\alpha}}$$

is then at least

$$\frac{2}{c} W\left(\frac{ec}{2}\right) + (x + o(1)) \frac{\partial \alpha_0}{\partial c} \sqrt{\frac{c}{2 \ln c}}$$

since $\alpha = (1 + o(1))2 \ln c/c$ and $\partial^2 \alpha_0 / \partial^2 c \geq 0$. Since

$$\frac{\partial \alpha_0}{\partial c} = -(1 + o(1)) \frac{2 \ln c}{c^2},$$

the statement follows from Theorem 1.1 □

Acknowledgments. We are grateful to Amin Coja-Oghlan, Alan Frieze, David Gamarnik, Yuval Peres, Alex Russell, and Joel Spencer for helpful conversations, and to the anonymous reviews for their comments.

References

- [1] Achlioptas, D., Moore, C.: Random k -SAT: Two moments suffice to cross a sharp threshold. *SIAM J. Comput.* 36, 740–762 (2006)
- [2] Achlioptas, D., Peres, Y.: The Threshold for Random k -SAT is $2k \log 2 - O(k)$. *J. AMS* 17, 947–973 (2004)
- [3] Bayati, M., Gamarnik, D., Tetali, P.: Combinatorial approach to the interpolation method and scaling limits in sparse random graphs. In: *Proc. STOC*, pp. 105–114 (2010)
- [4] Bollobás, B.: *Random graphs*, 2nd edn. Cambridge studies in advanced mathematics. Cambridge University Press, Cambridge (2001)
- [5] Coja-Oghlan, A., Efthymiou, C.: On independent sets in random graphs. In: *Proc. SODA*, pp. 136–144 (2011)
- [6] Frieze, A.M.: On the independence number of random graphs. *Discrete Math.* 81, 171–175 (1990)
- [7] Karp, R.M., Sipser, M.: Maximum matching in sparse random graphs. In: *Proc. FOCS*, pp. 364–375 (1981)
- [8] Shamir, E., Spencer, J.: Sharp concentration of the chromatic number on random graphs $G(n, p)$. *Combinatorica* 7, 121–129 (1987)

Extractors and Lower Bounds for Locally Samplable Sources

Anindya De* and Thomas Watson**

Computer Science Division
University of California, Berkeley, CA, USA
{anindya,tom}@cs.berkeley.edu

Abstract. We consider the problem of extracting randomness from sources that are efficiently samplable, in the sense that each output bit of the sampler only depends on some small number d of the random input bits. As our main result, we construct a deterministic extractor that, given any d -local source with min-entropy k on n bits, extracts $\Omega(k^2/nd)$ bits that are $2^{-n^{\Omega(1)}}$ -close to uniform, provided $d \leq o(\log n)$ and $k \geq n^{2/3+\gamma}$ (for arbitrarily small constants $\gamma > 0$). Using our result, we also improve a result of Viola (FOCS 2010), who proved a $1/2 - O(1/\log n)$ statistical distance lower bound for $o(\log n)$ -local samplers trying to sample input-output pairs of an explicit boolean function, assuming the samplers use at most $n + n^{1-\delta}$ random bits for some constant $\delta > 0$. Using a different function, we simultaneously improve the lower bound to $1/2 - 2^{-n^{\Omega(1)}}$ and eliminate the restriction on the number of random bits.

Keywords: extractors, lower bounds, locally samplable sources.

1 Introduction

Randomness extraction is the following general problem. Given a sample from an imperfect physical source of randomness, which is modeled as a probability distribution on bit strings of length n , we wish to apply an efficient deterministic algorithm to the sample to produce an output which is almost uniformly distributed (and thus is suitable for use by a randomized algorithm). Of course, to extract randomness from a source, the source needs to “contain” a certain amount of randomness in the first place. It is well established that the most suitable measure of the amount of randomness in a source is its *min-entropy* (defined below). However, even if the source is known to have at least $n - 1$ bits of min-entropy, no algorithm can extract even a single bit that is guaranteed to be close to uniformly distributed. To deal with this problem, researchers have

* This material is based upon work supported by the National Science Foundation under Grant No. CCF-1017403.

** This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0946797 and by the National Science Foundation under Grant No. CCF-1017403.

constructed *seeded extractors*, which have access to a short uniformly random seed that is statistically independent of the source and which acts as a catalyst for the extraction process.

However, there is a sense in which seeded extractors are overkill: They are guaranteed to work for completely arbitrary sources that have high enough min-entropy. It is reasonable to assume the physical source of randomness has some limited structure, in which case deterministic (that is, seedless) extraction may become viable. There are several classes of sources for which researchers have constructed good deterministic extractors. One such class is independent sources, where the n bits are partitioned into blocks which are assumed to be statistically independent of each other [7,9,11,42,23,21,3,18]. Other such classes include so-called bit-fixing sources [6,17,13,22], affine sources [12,5,22,8,28,19], polynomial sources [11], and algebraic varieties [10].

Trevisan and Vadhan [24] considered deterministic extractors for sources that are samplable by efficient algorithms given uniform random bits. One may initially be concerned that extracting randomness from such sources is somehow circular or vacuous: We are assuming uniform random bits are used to sample the source, and our goal then is to “undo” the sampling and get uniform random bits back. The point is that this class of sources is just a model for physical sources. This is motivated by the following postulate about the universe: A physical source of randomness is generated by an efficient process in nature, so it is reasonable to model the source as being sampled by an efficient algorithm.

Trevisan and Vadhan constructed extractors for the class of sources samplable by general *time-bounded* algorithms, but their constructions are conditional on standard complexity-theoretic conjectures. It is common in other areas of research that proving unconditional limits on the power of time-bounded algorithms is beyond the reach of current techniques. Thus we consider more restricted types of algorithms, such as small-space algorithms and bounded-depth circuits, which are combinatorially simple enough for us to prove unconditional results. It is natural to try to construct unconditional deterministic extractors for sources samplable by such restricted algorithms. Kamp et al. [16] succeeded in doing so for small-space samplers.

However, it is an open problem to construct an unconditional deterministic extractor for sources samplable by AC^0 -type circuits.¹ A basic obstacle is that this requires that input-output pairs of the extractor cannot be sampled by such circuits, and it is not even known how to construct an explicit function with the latter property. For example, although the parity function is known not to have subexponential-size constant-depth circuits [15], input-output pairs can be sampled very efficiently: Just take uniformly random bits x_1, \dots, x_n and output $x_1, x_1 \oplus x_2, x_2 \oplus x_3, \dots, x_{n-1} \oplus x_n, x_n$.

Our goal in this paper is to expand the frontier of unconditional deterministic randomness extraction for sources with low-complexity samplers. We succeed in constructing extractors for sources samplable by small-depth circuits with *bounded* fan-in gates, which corresponds to the class NC^0 . This is equivalent to

¹ Viola [27] solved this problem in independent and concurrent work; see below.

requiring that each output bit of the sampler only depends on a small number of input bits. We call such sources *locally samplable*.

1.1 Results

A distribution on a finite set S is said to have *min-entropy* at least k if each element of S occurs with probability at most 2^{-k} . The *statistical distance* between two distributions D_1 and D_2 on a finite set S is defined to be $\|D_1 - D_2\| = \max_{T \subseteq S} |\Pr_{D_1}[T] - \Pr_{D_2}[T]|$. If $\|D_1 - D_2\| \leq \epsilon$ then we also say D_1 and D_2 are ϵ -close. If $f : S \rightarrow S'$ and D is a distribution on S , then we let $f(D)$ denote the distribution on S' obtained by drawing a sample from D and applying f to it. When we mention a distribution multiple times in an expression, all instantiations refer to a single sample from the distribution; for example, $(D, f(D))$ denotes the distribution obtained by sampling $w \sim D$ and outputting the pair $(w, f(w))$. We use U_n to denote the uniform distribution on $\{0, 1\}^n$. If \mathcal{C} is a class of distributions on $\{0, 1\}^n$, then a function $\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ is called a (k, ϵ) -*extractor for \mathcal{C}* if for every distribution $D \in \mathcal{C}$ with min-entropy at least k , $\|\text{Ext}(D) - U_m\| \leq \epsilon$. Informally, when we say an extractor is *explicit* we mean that an efficient algorithm with the desired behavior is exhibited.

We define a d -*local sampler* to be a function $f : \{0, 1\}^r \rightarrow \{0, 1\}^n$ such that each output bit depends on at most d input bits. In other words, for every $j \in \{1, \dots, n\}$ there exists a subset $I_j \subseteq \{1, \dots, r\}$ with $|I_j| \leq d$ and a function $f_j : \{0, 1\}^{|I_j|} \rightarrow \{0, 1\}$ such that the j^{th} output bit of f is obtained by evaluating f_j on the input bits indexed by I_j . The output distribution of the sampler is $f(U_r)$. We say a distribution D on $\{0, 1\}^n$ is a d -*local source* if there exists a d -local sampler (with any input length r) whose output distribution is D .

Theorem 1. *For every constant $\gamma > 0$ there exists a constant $\beta > 0$ such that there exists an explicit (k, ϵ) -extractor for the class of d -local sources with output length $m = k^2/8nd$ and error $\epsilon = 2^{-n^\beta}$, provided $k \geq n^{2/3+\gamma}$ and $d \leq \beta \log n$.*

Theorem 2. *For every constant $\gamma > 0$ there exists a constant $\beta > 0$ such that there exists an explicit (k, ϵ) -extractor for the class of 1-local sources with output length $m = k - o(k)$ and error $\epsilon = 2^{-n^\beta}$, provided $k \geq n^{1/2+\gamma}$.*

Theorem 3. *There exists a universal constant $\beta > 0$ and an explicit function $F : \{0, 1\}^n \rightarrow \{0, 1\}$ such that for every d -local source D on $\{0, 1\}^{n+1}$ with $d \leq \beta \log n$, $\|D - (U_n, F(U_n))\| \geq 1/2 - 2^{-n^\beta}$.*

1.2 Techniques

The proof of Theorem 1 has three steps.

The first step is to construct a certain extractor for 1-local sources (which in particular yields Theorem 2). To do this, we observe that extractors for so-called low-weight affine sources also work for 1-local sources. Then we construct an extractor for low-weight affine sources, building on and improving Rao’s extractor

from [22]. While Rao’s extractor handles affine sources of min-entropy at least k and weight at most k^γ for some constant $\gamma > 0$, our improvement handles sources with weight at most $k^{1-\gamma}$ for *any* constant $\gamma > 0$. The key ingredient in our improvement is the strong condenser of Guruswami, Umans, and Vadhan [14]. We present this step in Section 3.

The second step is to show that extractors for 1-local sources also work for $o(\log n)$ -local sources. To do this, we relate the problem to a concept we call *superindependent matchings* in bipartite graphs, and we prove a combinatorial lemma about the existence of such matchings. We present this step in Section 4.

The third step is to increase the output length of the extractor using the technique of “obtaining an independent seed” introduced by Gabizon et al. [13]. Combining step 1 and step 2 yields an extractor with output length $\Omega(k^2/nd^{32d})$. To increase the output length to $\Omega(k^2/nd)$, we adapt the technique from [13]. A key ingredient in our argument is a lemma due to Vadhan [25], which is a strengthened version of a classic lemma due to Nisan and Zuckerman [20]. While the result of [13] achieves output length $k - o(k)$ for bit-fixing sources, we lose a factor of $\Omega(k/n)$ in the output length due to the use of Vadhan’s lemma, and we lose another factor of $\Omega(1/d)$ since conditioning on p bits of the output of a d -local sampler could cause a loss of pd bits of min-entropy. We present this step in Section 5.

Viola [26] proved a version of Theorem 3 where the statistical distance lower bound is only $1/2 - O(1/\log n)$, and the d -local sampler is restricted to use at most $n + n^{1-\delta}$ random bits for any constant $\delta > 0$. His function F is what he calls “majority mod p ”. Using a different function F (namely, any bit of the extractor underlying Theorem 1), we simultaneously improve the lower bound to $1/2 - 2^{-n^{\Omega(1)}}$ and eliminate the restriction on the number of random bits. Our proof of Theorem 3 uses ideas similar to Viola’s, but is actually somewhat simpler given the extraction property of F . In [26], Viola also showed that for symmetric functions F , one cannot hope to get such a strong lower bound for samplers that are polynomial-size constant-depth circuits. Our extractor function F is not symmetric. We present the proof of Theorem 3 in Section 6.

In independent and concurrent work, Viola [27] obtained extractors for d -local sources with $d \leq n^{o(1)}$ and for sources sampled by AC^0 -type circuits. The high level idea behind the extractor remains the same: Show the given source is close to a convex combination of 1-local sources, and use the extractor in [22]. However, the proofs in [27] are much more complicated than in this paper.

2 Preliminaries

In this paper we work with bipartite graphs $G = (L, R, E)$, where L, R are disjoint finite sets (the left and right nodes) and E is a set of unordered pairs where one element comes from L and the other from R . The distance between two nodes is the number of edges on a shortest path between them.

To every function $f : \{0, 1\}^r \rightarrow \{0, 1\}^n$ we associate a bipartite graph $G = (L, R, E)$ where $L = \{1, \dots, r\} \times \{\text{in}\}$, $R = \{1, \dots, n\} \times \{\text{out}\}$, and $\{(i, \text{in}), (j, \text{out})$

j th output bit of f depends on the i th input bit of f (that is, for some setting of all input bits except the i th, the j th output bit equals the i th input bit or its complement). Note that we include no unnecessary edges, and the graph is unique. We use $I_j \times \{\text{in}\}$ to denote the set of neighbors of (j, out) and $J_i \times \{\text{out}\}$ to denote the set of neighbors of (i, in) . Observe that if $f(U_r)$ has min-entropy at least k , then there are at least k non-isolated nodes in L , and in particular $r \geq k$.

We say f is a d -local sampler if each node in R has degree at most d , and we say a distribution on $\{0, 1\}^n$ is a d -local source if it equals $f(U_r)$ for some d -local sampler f (with any input length r). We say f is a (d, c) -local sampler if each node in R has degree at most d and each node in L has degree at most c , and we say a distribution on $\{0, 1\}^n$ is a (d, c) -local source if it equals $f(U_r)$ for some (d, c) -local sampler f (with any input length r).

Suppose Y is a finite set of indices, $(p_y)_{y \in Y}$ is a distribution on Y , and for each $y \in Y$, D_y is a distribution on a finite set S . Then the convex combination $\sum_{y \in Y} p_y D_y$ is defined to be the distribution on S obtained by sampling y according to $(p_y)_{y \in Y}$ and then outputting a sample from D_y .

Lemma 1. *Suppose $\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ is any function and suppose $D = \sum_{y \in Y} p_y D_y$ is a distribution on $\{0, 1\}^n$. Then for every $\epsilon \geq 0$,*

$$\|\text{Ext}(D) - U_m\| \leq \epsilon + \Pr_{y \sim (p_y)_{y \in Y}} \left[\|\text{Ext}(D_y) - U_m\| > \epsilon \right].$$

Corollary 1. *Suppose every distribution in \mathcal{C} with min-entropy at least k can be written as a convex combination $\sum_{y \in Y} p_y D_y$ where*

$$\Pr_{y \sim (p_y)_{y \in Y}} [D_y \text{ is in } \mathcal{C}' \text{ and has min-entropy at least } k'] \geq 1 - \delta.$$

Then every (k', ϵ') -extractor for \mathcal{C}' is also a (k, ϵ) -extractor for \mathcal{C} where $\epsilon = \epsilon' + \delta$.

Lemma 2. *Every d -local source with min-entropy at least k is a convex combination of (d, c) -local sources with min-entropy at least $k - nd/c$.*

Proof. Consider an arbitrary d -local sampler $f : \{0, 1\}^r \rightarrow \{0, 1\}^n$ whose output distribution has min-entropy at least k , and let $G = (L, R, E)$ be the associated bipartite graph. Since $|E| \leq nd$, there are at most nd/c nodes in L with degree greater than c ; without loss of generality these nodes are $\{r - \ell + 1, \dots, r\} \times \{\text{in}\}$ for some $\ell \leq nd/c$. For each string $y \in \{0, 1\}^\ell$, define $f_y : \{0, 1\}^{r-\ell} \rightarrow \{0, 1\}^n$ as $f_y(x) = f(x, y)$ (hardwiring the last ℓ bits to y). Then $f(U_r) = \sum_{y \in \{0, 1\}^\ell} \frac{1}{2^\ell} f_y(U_{r-\ell})$. Moreover, each $f_y(U_{r-\ell})$ is a (d, c) -local source with min-entropy at least $k - nd/c$, since if some $z \in \{0, 1\}^n$ and $y^* \in \{0, 1\}^\ell$ satisfied $\Pr_{x \sim U_{r-\ell}} [f_{y^*}(x) = z] > 1/2^{k-nd/c}$ then we would have

$$\begin{aligned} \Pr_{x \sim U_{r-\ell}, y \sim U_\ell} [f(x, y) = z] &\geq \Pr_{y \sim U_\ell} [y = y^*] \cdot \Pr_{x \sim U_{r-\ell}} [f(x, y^*) = z] \\ &> \frac{1}{2^\ell} \cdot \frac{1}{2^{k-nd/c}} \\ &\geq 1/2^k \end{aligned}$$

contradicting that $f(U_r)$ has min-entropy at least k . □

We also use seeded extractors. A function $\text{Ext} : \{0, 1\}^n \times \{0, 1\}^t \rightarrow \{0, 1\}^m$ is called a *seeded (k, ϵ) -extractor* if for every distribution D on $\{0, 1\}^n$ with min-entropy at least k , $\|\text{Ext}(D, U_t) - U_m\| \leq \epsilon$ where U_t is independent of D .

If $z \in \{0, 1\}^n$ and $J \subseteq \{1, \dots, n\}$, then we let $z|_J \in \{0, 1\}^{|J|}$ denote the substring of z indexed by the coordinates in J . If D is a distribution on $\{0, 1\}^n$ and $J \subseteq \{1, \dots, n\}$, then we let $D|_J$ denote the marginal distribution on the coordinates in J . Finally, all logarithms in this paper are base 2.

3 1-Local Sources

An *affine source* is a distribution on $\{0, 1\}^n$ which is uniform over an affine subspace (where $\{0, 1\}^n$ is viewed as a vector space over \mathbb{F}_2). If the subspace has dimension k then it has size 2^k and hence the source has min-entropy k . The distribution can be sampled by picking $x_1, \dots, x_k \in \{0, 1\}$ uniformly at random and outputting $z_0 + x_1 z_1 + \dots + x_k z_k$ where $z_0 \in \{0, 1\}^n$ is a shift vector and $z_1, \dots, z_k \in \{0, 1\}^n$ are a basis of the associated linear subspace. The source is said to be a *weight- c affine source* if there exist basis vectors z_1, \dots, z_k each of which has Hamming weight at most c .

Observation 1. *Every $(1, c)$ -local source is also a weight- c affine source.*

Rao [22] constructed extractors for low-weight affine sources.

Theorem 4 ([22]). *There exist universal constants $C, \gamma > 0$ such that for all $k \geq \log^C n$ there exists an explicit $(k, 2^{-k^{\Omega(1)}})$ -extractor with output length $m = k - o(k)$ for the class of weight- k^γ affine (and in particular, $(1, k^\gamma)$ -local) sources.*

We improve Rao’s result to obtain the following theorem.

Theorem 5. *There exists a universal constant $C > 0$ such that for every constant $\gamma > 0$ and all $k \geq \log^{C/\gamma} n$ there exists an explicit $(k, 2^{-k^{\Omega(1)}})$ -extractor with output length $m = k - o(k)$ for the class of weight- $k^{1-\gamma}$ affine (and in particular, $(1, k^{1-\gamma})$ -local) sources.*

We present the proof of Theorem 5 in the full version of the paper. Our proof closely follows Rao’s proof of Theorem 4, but we insert an application of a linear strong condenser from [14]. Also, where Rao uses a small-bias generator, we instead use a parity-check function for a BCH code. These changes allow us to get better parameters.

We now explain how Theorem 2 follows from Theorem 5, Lemma 2, and Corollary 1. We first note the following immediate corollary of Theorem 5

Corollary 2. *For every constant $\gamma > 0$ there exists a constant $\beta > 0$ such that for all $k \geq n^{1/2+\gamma}$ there exists an explicit $(k, 2^{-n^\beta})$ -extractor with output length $m = k - o(k)$ for the class of weight- $n^{1/2}$ affine (and in particular, $(1, n^{1/2})$ -local) sources.*

Lemma 2 implies that every 1-local source with min-entropy at least $k \geq n^{1/2+\gamma}$ is a convex combination of $(1, n^{1/2})$ -local sources with min-entropy at least $k - n^{1/2} \geq k - o(k)$. Theorem 2 then follows from Corollary 1 (with $\delta = 0$) and Corollary 2.

Bourgain [5], Yehudayoff [28], and Li [19] constructed extractors for linear min-entropy affine sources (of arbitrary weight), achieving better error but worse output length than Theorem 5. This can be used to improve the error in Theorem 1 and Theorem 2 when $k \geq \Omega(n)$ and $d \leq O(1)$. We omit the details.

4 d-Local Sources

The following theorem shows that to get extractors for d -local sources, it suffices to construct extractors for 1-local sources.

Theorem 6. *Every (k', ϵ') -extractor for $(1, 2nd/k)$ -local sources is also a (k, ϵ) -extractor for d -local sources, where $k' = k^2/4nd^32^d$ and $\epsilon = \epsilon' + e^{-k'/4}$.*

Assuming $k \geq n^{2/3+\gamma}$ (for constant $\gamma > 0$) and $d \leq \beta \log n$ (for small enough constant $\beta > 0$) in Theorem 6, we find that it suffices to have a (k', ϵ') -extractor for $(1, c)$ -local sources where $k' \geq n^{1/3+\gamma}$ and $c = 2nd/k \leq n^{1/3} \leq (k')^{1-\gamma}$. Such an extractor is given by Theorem 5, with error $\epsilon' = 2^{-n^{\Omega(1)}}$ (and thus $\epsilon = \epsilon' + e^{-k'/4} \leq 2^{-n^{\Omega(1)}}$). This already yields a version of Theorem 1 with output length $k' - o(k') = \Omega(k^2/nd^32^d)$.

4.1 Superindependent Matchings

We first prove a combinatorial lemma that is needed for the proof of Theorem 6.

Definition 1. *Given a bipartite graph $G = (L, R, E)$, we say a set of edges $M \subseteq E$ is a superindependent matching if there is no path of length at most two in G from an endpoint of an edge in M to an endpoint of a different edge in M .*

Lemma 3. *Suppose $G = (L, R, E)$ is a bipartite graph with no isolated nodes and such that each node in L has degree at most c and each node in R has degree at most d . Then G has a superindependent matching of size at least $|L|/d^2c$.*

Proof. Let M be a largest superindependent matching in G , and suppose for contradiction that $|M| < |L|/d^2c$. Note that for each node in R , the number of nodes in L within distance three in G is at most $d(1+(c-1)(d-1)) \leq d^2c$. Thus the number of nodes in L within distance three of the right endpoints of edges in M is at most $|M| \cdot d^2c < |L|$. Hence there exists a node $u \in L$ at distance greater than three from the right endpoint of every edge in M . Since G has no isolated nodes, there exists a node $v \in R$ such that $\{u, v\} \in E$. Note that there is no path of length at most two from either u or v to an endpoint of an edge in M , since otherwise a simple case analysis would show that u is within distance three of the right endpoint of an edge in M . Thus $M \cup \{\{u, v\}\}$ is a superindependent matching, contradicting the maximality of M . \square

4.2 Proof of Theorem 6

Suppose that $\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ is a (k', ϵ') -extractor for $(1, 2nd/k)$ -local sources. By Corollary 1 (with $\delta = 0$) and Lemma 2 it suffices to show that Ext is a $(k/2, \epsilon)$ -extractor for (d, c) -local sources where $c = 2nd/k$. The plan is to show that every (d, c) -local source with min-entropy at least $k/2$ is a convex combination of $(1, c)$ -local sources most of which have min-entropy at least k' , and then apply Corollary 1 again.

So consider an arbitrary (d, c) -local sampler $f : \{0, 1\}^r \rightarrow \{0, 1\}^n$ whose output distribution has min-entropy at least $k/2$, and let $G = (L, R, E)$ be the associated bipartite graph. If we obtain \tilde{G} from G by removing any isolated nodes, then \tilde{G} still has at least $k/2$ nodes on its left side. Applying Lemma 3 to \tilde{G} tells us that G has a superindependent matching M of size at least $k/(2d^2c)$. Let $\ell = |M|$, and without loss of generality assume that the left endpoints of M are $L' = \{1, \dots, \ell\} \times \{\text{in}\}$. We write inputs to f as (x, y) where $x \in \{0, 1\}^\ell$ and $y \in \{0, 1\}^{r-\ell}$. Since M is superindependent, each node in R is adjacent to at most one node in L' . Thus if we define $f_y : \{0, 1\}^\ell \rightarrow \{0, 1\}^n$ as $f_y(x) = f(x, y)$ (hardwiring the last $r - \ell$ input bits to y) then for each y , f_y is a $(1, c)$ -local sampler. Observe that $f(U_r) = \sum_{y \in \{0, 1\}^{r-\ell}} \frac{1}{2^{r-\ell}} f_y(U_\ell)$.

Let $G_y = (L', R, E_y)$ denote the bipartite graph associated with f_y . The min-entropy of $f_y(U_\ell)$ is the number of nodes in L' that are non-isolated in G_y . Although each node in L' is non-isolated in G (since $M \subseteq E$), edges incident to L' may disappear when we hardwire y . We claim that with high probability over y , plenty of nodes in L' are still non-isolated in G_y and hence $f_y(U_\ell)$ has high min-entropy. For $i \in \{1, \dots, \ell\}$ let $(j_i, \text{out}) \in R$ be the neighbor of (i, in) in M , and let $I_{j_i} \times \{\text{in}\}$ be the set of neighbors of (j_i, out) in G . Since the j_i^{th} output bit of f depends on the i^{th} input bit, there exists a string $w_i \in \{0, 1\}^{|I_{j_i}|-1}$ such that hardwiring the input bits corresponding to $I_{j_i} \setminus \{i\}$ to w_i leaves the edge $\{(i, \text{in}), (j_i, \text{out})\}$ in place, and in particular ensures that (i, in) is non-isolated. Since M is superindependent, the sets I_{j_i} for $i \in \{1, \dots, \ell\}$ are pairwise disjoint and in particular, each $I_{j_i} \setminus \{i\} \subseteq \{\ell + 1, \dots, r\}$. We assume the bits of y are indexed starting at $\ell + 1$, so for example $y|_{\{\ell+1\}}$ is the first bit of y . By the disjointness, we find that the events $y|_{I_{j_i} \setminus \{i\}} = w_i$ (for $i \in \{1, \dots, \ell\}$) are fully independent over $y \sim U_{r-\ell}$. Moreover, each of these events occurs with probability at least $1/2^{d-1}$ since $|w_i| \leq d - 1$. Thus we have

$$\begin{aligned} & \Pr_{y \sim U_{r-\ell}} [f_y(U_\ell) \text{ does not have min-entropy at least } k'] \\ &= \Pr_{y \sim U_{r-\ell}} \left[\left| \{i \in \{1, \dots, \ell\} : (i, \text{in}) \text{ is non-isolated in } G_y\} \right| < k' \right] \\ &\leq \Pr_{y \sim U_{r-\ell}} \left[\left| \{i \in \{1, \dots, \ell\} : y|_{I_{j_i} \setminus \{i\}} = w_i\} \right| < k' \right] \\ &\leq e^{-k/8d^2c2^d} \end{aligned}$$

by a standard Chernoff bound.

To summarize, we have shown that every (d, c) -local source with min-entropy at least $k/2$ is a uniform convex combination of $(1, c)$ -local sources, at most

Ingredients:

$\text{Ext}' : \{0, 1\}^n \rightarrow \{0, 1\}^{m'}$

$\text{Ext}'_1 : \{0, 1\}^n \rightarrow \{0, 1\}^s$ is the first s bits of Ext'

$\text{Ext}'_2 : \{0, 1\}^n \rightarrow \{0, 1\}^{m'-s}$ is the last $m' - s$ bits of Ext'

$\text{Samp} : \{0, 1\}^s \rightarrow \binom{\{1, \dots, n\}}{p}$

$\text{SExt} : \{0, 1\}^p \times \{0, 1\}^{m'-s} \rightarrow \{0, 1\}^m$

Result:

$\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}^m$ defined as $\text{Ext}(z) = \text{SExt}(z|_{\text{Samp}(\text{Ext}'_1(z))}, \text{Ext}'_2(z))$

Fig. 1. Increasing the output length of an extractor for d -local sources

$e^{-k/8d^2c2^d}$ fraction of which do not have min-entropy at least k' . It now follows from Corollary 1 that Ext is a $(k/2, \epsilon)$ -extractor for (d, c) -local sources. This finishes the proof of Theorem 6.

5 Increasing the Output Length

Combining the results from Section 3 and Section 4 yields an extractor for d -local sources with output length $\Omega(k^2/nd^32^d)$, provided $d \leq o(\log n)$ and the min-entropy k is at least $n^{2/3+\gamma}$. In this section we show how to improve the output length to $\Omega(k^2/nd)$. We now present our general theorem on increasing the output length of extractors for d -local sources (Theorem 7 and Figure 1), which uses the technique of “obtaining an independent seed”. As in [13], the strategy is to take the output of a deterministic extractor and use part of it to sample a set of coordinates of the source, which are then plugged into a seeded extractor, using the other part of the deterministic extractor’s output as the seed. A key ingredient (which was not used in [13]) is a fundamental lemma of Nisan and Zuckerman [20], which roughly says that if we sample the coordinates appropriately, then the min-entropy rate of the marginal distribution on those coordinates is almost as high as the min-entropy rate of the whole source. However, the original Nisan-Zuckerman lemma loses a logarithmic factor in the min-entropy rate. We use a strengthened version of the lemma, due to Vadhan [25], which only loses a constant factor.

We use $\binom{\{1, \dots, n\}}{p}$ to denote the set of subsets of $\{1, \dots, n\}$ of size p .

Definition 2. We say $\text{Samp} : \{0, 1\}^s \rightarrow \binom{\{1, \dots, n\}}{p}$ is a (μ, η) -sampler if for every $g : \{1, \dots, n\} \rightarrow [0, 1]$ with $\frac{1}{n} \sum_{j=1}^n g(j) \geq \mu$ it holds that

$$\Pr_{\sigma \sim U_s} \left[\frac{1}{p} \sum_{j \in \text{Samp}(\sigma)} g(j) < \mu/2 \right] \leq \eta.$$

Theorem 7. There exists a constant $\alpha > 0$ such that the following holds. Suppose Ext' is a (k', ϵ') -extractor for d -local sources, Samp is a $(k/2n \log(4n/k), \eta)$ -sampler, and SExt is a seeded $(pk/4n, \epsilon'')$ -extractor. Then Ext is a (k, ϵ) -extractor for d -local sources, where $k = k' + pd$ and $\epsilon = \epsilon'(2^{s+1} + 1) + 2\sqrt{\eta + 2^{-\alpha k}} + \epsilon''$.

Due to space constraints, we defer the proof of Theorem 7, as well as the parameter-setting to derive Theorem 1, to the full version.

6 Improved Lower Bounds for Sampling Input-Output Pairs

For this section, we define a (d, c, k) -local sampler to be a (d, c) -local sampler with at least k non-isolated nodes on the left side of its associated bipartite graph (that is, it makes nontrivial use of at least k random bits). We say a distribution on $\{0, 1\}^n$ is a (d, c, k) -local source if it equals $f(U_r)$ for some (d, c, k) -local sampler f (with any input length r). Note that a (d, c, k) -local source might not have min-entropy at least k .

Theorem 8. *Suppose $\text{Ext} : \{0, 1\}^n \rightarrow \{0, 1\}$ is a $(0, \epsilon)$ -extractor for $(d, 8d, n/4)$ -local sources, where $d < n/8$. Then for every d -local source D on $\{0, 1\}^{n+1}$ we have $\|D - (U_n, \text{Ext}(U_n))\| \geq 1/2 - \epsilon - 2^{-n/2}$.*

It might seem suspicious that we are assuming Ext is a $(0, \epsilon)$ -extractor. We are not, in fact, extracting from sources with 0 min-entropy — it is possible to derive a lower bound on the min-entropy of any $(d, 8d, n/4)$ -local source. The point is that for Theorem 8, we do not care about the min-entropy, only the number of non-isolated input nodes.

In the proof of Theorem 6, we implicitly showed that any particular bit of the extractor from Theorem 5 (for min-entropy $n^{0.9}$ and weight $\log n$) is a $(0, \epsilon)$ -extractor for $(d, 8d, n/4)$ -local sources with error $\epsilon = 2^{-n^{\Omega(1)}}$, provided $d \leq \beta \log n$ for some small enough constant $\beta > 0$. Theorem 3 follows immediately from this and Theorem 8.

Proof (of Theorem 8). Consider an arbitrary d -local sampler $f : \{0, 1\}^r \rightarrow \{0, 1\}^{n+1}$, and let $G = (L, R, E)$ be the associated bipartite graph. Since $|E| \leq (n + 1)d$, there are at most $(n + 1)/8$ nodes in L with degree greater than $8d$. Also, at most $d \leq (n - 1)/8$ nodes in L are adjacent to $(n + 1, \text{out})$. Without loss of generality, the nodes in L that either have degree greater than $8d$ or are adjacent to $(n + 1, \text{out})$ are $\{r - \ell + 1, \dots, r\} \times \{\text{in}\}$ for some $\ell \leq (n + 1)/8 + (n - 1)/8 = n/4$. For each string $y \in \{0, 1\}^\ell$, define $f_y : \{0, 1\}^{r-\ell} \rightarrow \{0, 1\}^{n+1}$ as $f_y(x) = f(x, y)$ (hardwiring the last ℓ bits to y) and let $G_y = (L', R, E_y)$ be the associated bipartite graph, where $L' = \{1, \dots, r - \ell\} \times \{\text{in}\}$. Observe that $f(U_r) = \sum_{y \in \{0, 1\}^\ell} \frac{1}{2^\ell} f_y(U_{r-\ell})$. We define the tests

$$T_1 = \{z \in \{0, 1\}^{n+1} : \exists x \in \{0, 1\}^{r-\ell}, y \in \{0, 1\}^\ell \text{ such that } f(x, y) = z \text{ and } \{i \in \{1, \dots, r - \ell\} : (i, \text{in}) \text{ is non-isolated in } G_y\} < n/4\}$$

and $T_2 = \{z \in \{0, 1\}^{n+1} : \text{Ext}(z_{\{1, \dots, n\}}) \neq z_{\{n+1\}}\}$ (in other words, T_2 is the complement of the support of $(U_n, \text{Ext}(U_n))$). We define the test $T = T_1 \cup T_2$.

Proposition 1. $\Pr_{f(U_r)}[T] \geq 1/2 - \epsilon$.

Proposition 2. $\Pr_{(U_n, \text{Ext}(U_n))}[T] \leq 2^{-n/2}$.

Combining the two propositions, we have $|\Pr_{f(U_r)}[T] - \Pr_{(U_n, \text{Ext}(U_n))}[T]| \geq 1/2 - \epsilon - 2^{-n/2}$, thus witnessing that $\|f(U_r) - (U_n, \text{Ext}(U_n))\| \geq 1/2 - \epsilon - 2^{-n/2}$.

Proof (of Proposition 7). It suffices to show that $\Pr_{f_y(U_{r-\ell})}[T] \geq 1/2 - \epsilon$ holds for each $y \in \{0, 1\}^\ell$. If $|\{i \in \{1, \dots, r-\ell\} : (i, \text{in}) \text{ is non-isolated in } G_y\}| < n/4$ then of course $\Pr_{f_y(U_{r-\ell})}[T_1] = 1$. Otherwise, $f_y(U_{r-\ell})$ is a $(d, 8d, n/4)$ -source on $\{0, 1\}^{n+1}$. Note that $(n+1, \text{out})$ is isolated in G_y ; we define $b_y \in \{0, 1\}$ to be the fixed value of the $(n+1)^{\text{st}}$ output bit of f_y , and we define $f'_y : \{0, 1\}^{r-\ell} \rightarrow \{0, 1\}^n$ to be the first n output bits of f_y . Since $f'_y(U_{r-\ell})$ is a $(d, 8d, n/4)$ -source on $\{0, 1\}^n$, we have $\|\text{Ext}(f'_y(U_{r-\ell})) - U_1\| \leq \epsilon$ and thus $\Pr_{b \sim \text{Ext}(f'_y(U_{r-\ell}))}[b \neq b_y] \geq 1/2 - \epsilon$. In other words, $\Pr_{f_y(U_{r-\ell})}[T_2] \geq 1/2 - \epsilon$. \square

Proof (of Proposition 8). By definition, $\Pr_{(U_n, \text{Ext}(U_n))}[T_2] = 0$. Note that $|T_1| \leq 2^{n/2}$ since each string in T_1 can be described by a string of length at most $\ell + n/4 \leq n/2$, namely an appropriate value of y along with the bits of x such that the corresponding nodes in L' are non-isolated in G_y . Since $(U_n, \text{Ext}(U_n))$ is uniform over a set of size 2^n , we get $\Pr_{(U_n, \text{Ext}(U_n))}[T_1] \leq 2^{n/2}/2^n = 2^{-n/2}$. \square

This finishes the proof of Theorem 8. \square

Acknowledgments. We thank Zeev Dvir, Omer Reingold, Salil Vadhan, and Avi Wigderson for helpful email exchanges. In particular, A.D. would like to thank Omer and Salil for answering his innumerable queries about extractors and Salil for suggesting the use of GUV condensers. We had helpful conversations about this work with Umilma Mahadev, Luca Trevisan, and Gregory Valiant.

References

1. Barak, B., Impagliazzo, R., Wigderson, A.: Extracting randomness using few independent sources. *SIAM Journal on Computing* 36(4), 1095–1118 (2006)
2. Barak, B., Kindler, G., Shaltiel, R., Sudakov, B., Wigderson, A.: Simulating independence: New constructions of condensers, Ramsey graphs, dispersers, and extractors. *Journal of the ACM* 57(4) (2010)
3. Barak, B., Rao, A., Shaltiel, R., Wigderson, A.: 2-source dispersers for sub-polynomial entropy and Ramsey graphs beating the Frankl-Wilson construction. In: *Proceedings of the 38th ACM Symposium on Theory of Computing*, pp. 671–680 (2006)
4. Bourgain, J.: More on the sum-product phenomenon in prime fields and its applications. *International Journal of Number Theory* 1, 1–32 (2005)
5. Bourgain, J.: On the construction of affine-source extractors. *Geometric and Functional Analysis* 1, 33–57 (2007)
6. Chor, B., Friedman, J., Goldreich, O., Håstad, J., Rudich, S., Smolensky, R.: The bit extraction problem or t -resilient functions. In: *Proceedings of the 26th IEEE Symposium on Foundations of Computer Science*, pp. 396–407 (1985)

7. Chor, B., Goldreich, O.: Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing* 17(2), 230–261 (1988)
8. DeVos, M., Gabizon, A.: Simple affine extractors using dimension expansion. In: *Proceedings of the 25th IEEE Conference on Computational Complexity*, pp. 50–57 (2010)
9. Dodis, Y., Elbaz, A., Oliveira, R., Raz, R.: Improved randomness extraction from two independent sources. In: Jansen, K., Khanna, S., Rolim, J.D.P., Ron, D. (eds.) *RANDOM 2004 and APPROX 2004*. LNCS, vol. 3122, pp. 334–344. Springer, Heidelberg (2004)
10. Dvir, Z.: Extractors for varieties. In: *Proceedings of the 24th IEEE Conference on Computational Complexity*, pp. 102–113 (2009)
11. Dvir, Z., Gabizon, A., Wigderson, A.: Extractors and rank extractors for polynomial sources. *Computational Complexity* 18(1), 1–58 (2009)
12. Gabizon, A., Raz, R.: Deterministic extractors for affine sources over large fields. *Combinatorica* 28(4), 415–440 (2008)
13. Gabizon, A., Raz, R., Shaltiel, R.: Deterministic extractors for bit-fixing sources by obtaining an independent seed. *SIAM Journal on Computing* 36(4), 1072–1094 (2006)
14. Guruswami, V., Umans, C., Vadhan, S.: Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM* 56(4) (2009)
15. Hastad, J.: Almost optimal lower bounds for small depth circuits. In: *Proceedings of the 18th ACM Symposium on Theory of Computing*, pp. 6–20 (1986)
16. Kamp, J., Rao, A., Vadhan, S., Zuckerman, D.: Deterministic extractors for small-space sources. *Journal of Computer and System Sciences* 77(1), 191–220 (2011)
17. Kamp, J., Zuckerman, D.: Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM Journal on Computing* 36(5), 1231–1247 (2007)
18. Li, X.: Improved constructions of three source extractors. In: *Proceedings of the 26th IEEE Conference on Computational Complexity*, pp. 126–136 (2011)
19. Li, X.: A new approach to affine extractors and dispersers. In: *Proceedings of the 26th IEEE Conference on Computational Complexity*, pp. 137–147 (2011)
20. Nisan, N., Zuckerman, D.: Randomness is linear in space. *Journal of Computer and System Sciences* 52(1), 43–52 (1996)
21. Rao, A.: Extractors for a constant number of polynomially small min-entropy independent sources. *SIAM Journal on Computing* 39(1), 168–194 (2009)
22. Rao, A.: Extractors for low-weight affine sources. In: *Proceedings of the 24th IEEE Conference on Computational Complexity*, pp. 95–101 (2009)
23. Raz, R.: Extractors with weak random seeds. In: *Proceedings of the 37th ACM Symposium on Theory of Computing*, pp. 11–20 (2005)
24. Trevisan, L., Vadhan, S.: Extracting randomness from samplable distributions. In: *Proceedings of the 41st IEEE Symposium on Foundations of Computer Science*, pp. 32–42 (2000)
25. Vadhan, S.: Constructing locally computable extractors and cryptosystems in the bounded-storage model. *Journal of Cryptology* 17(1), 43–77 (2004)
26. Viola, E.: The complexity of distributions. In: *Proceedings of the 51st IEEE Symposium on Foundations of Computer Science*, pp. 202–211 (2010)
27. Viola, E.: Extractors for circuit sources. Technical Report TR11-056, Electronic Colloquium on Computational Complexity (2011)
28. Yehudayoff, A.: Affine extractors over prime fields (2010) (manuscript)

A Deterministic Algorithm for the Frieze-Kannan Regularity Lemma

Domingos Dellamonica^{1,*}, Subrahmanyam Kalyanasundaram²,
Daniel Martin^{3,**}, Vojtěch Rödl^{1,***}, and Asaf Shapira^{4,†}

¹ Department of Mathematics and Computer Science,
Emory University, Atlanta, GA 30322
{ddellam,rodl}@mathcs.emory.edu

² School of Computer Science, Georgia Institute of Technology,
Atlanta, GA 30332
subruk@cc.gatech.edu

³ Center for Mathematics, Computer Science and Cognition, Universidade Federal do
ABC, Santo André, SP 09210-170 Brazil
daniel.martin@ufabc.edu.br

⁴ School of Mathematics and School of Computer Science,
Georgia Institute of Technology, Atlanta, GA 30332
asafico@math.gatech.edu

Abstract. The Frieze-Kannan regularity lemma is a powerful tool in combinatorics. It has also found applications in the design of approximation algorithms and recently in the design of fast combinatorial algorithms for boolean matrix multiplication. The algorithmic applications of this lemma require one to efficiently construct a partition satisfying the conditions of the lemma.

Williams [24] recently asked if one can construct a partition satisfying the conditions of the Frieze-Kannan regularity lemma in deterministic sub-cubic time. We resolve this problem by designing an $\tilde{O}(n^\omega)$ time algorithm for constructing such a partition, where $\omega < 2.376$ is the exponent of fast matrix multiplication. The algorithm relies on a spectral characterization of vertex partitions satisfying the properties of the Frieze-Kannan regularity lemma.

1 Introduction

1.1 Background and Motivation

The Regularity Lemma of Szemerédi [21] is one of the most powerful tools in tackling combinatorial problems in various areas like extremal graph theory, additive combinatorics and combinatorial geometry. For a detailed discussion of

* Supported in part by a Fulbright-CAPES scholarship.

** Supported in part by CNPq 475064/2010-0.

*** Supported in part by NSF grant DMS0800070.

† Supported in part by NSF Grant DMS-0901355.

these applications, we refer the reader to [15]. The regularity lemma guarantees that the vertex set of any (dense) graph $G = (V, E)$ can be partitioned into a *bounded* number of vertex sets V_1, \dots, V_k such that almost all the bipartite graphs (V_i, V_j) are pseudo-random (see Section 1.2 for precise definitions). Hence, one can think of Szemerédi's regularity lemma as saying that any graph can be approximated by a finite structure. This aspect of the regularity lemma has turned out to be extremely useful for designing approximation algorithms, since in some cases one can approximate certain properties of a graph (say, the Max-Cut of the graph) by investigating its regular partition (which is of constant size). In order to apply this algorithmic scheme one should be able to efficiently construct a partition satisfying the condition of the lemma. While Szemerédi's proof of his lemma was only existential, it is known how to efficiently construct a partition satisfying the conditions of the lemma. The first to achieve this goal were Alon et al. [2] who showed that this task can be carried out in time $O(n^\omega)$, where here and throughout this paper ω is the exponent of fast matrix multiplication. The algorithm of Coppersmith and Winograd [7] gives $\omega < 2.376$. The $O(n^\omega)$ algorithm of Alon et al. [2] was later improved by Kohayakawa, Rödl and Thoma [14] who gave a deterministic $O(n^2)$ algorithm.

The main drawback of Szemerédi's regularity lemma is that the constants involved are huge; Gowers [13] proved that in some cases the number of parts in a Szemerédi regular partition grows as a tower of exponents of height polynomial in the error parameter ε . It is thus natural to ask if one can find a slightly weaker regularity lemma which would be applicable, while at the same time not involve such huge constants. Such a lemma was indeed considered in [20] for bipartite graphs and in [8] for arbitrary graphs. Subsequently, Frieze and Kannan [9,10] devised an elegant regularity lemma of this type. They formulated a slightly weaker notion of regularity (see Definition 1) which we will refer to as FK-regularity. They proved that any graph has an FK-regular partition involving drastically fewer parts compared to Szemerédi's lemma. They also showed that an FK-regular partition can still be used in some of the cases where Szemerédi's lemma was used. The notion of FK-regularity has been studied extensively in the past decade. For example, it is a key part of the theory of graph limits developed in recent years, see the survey of Lovász [17]. Finally, FK-regularity was a key tool in the recent breakthrough of Bansal and Williams [4], where they obtained new bounds for combinatorial boolean matrix multiplication.

As in the case of Szemerédi's regularity lemma, in order to algorithmically apply the FK-regularity lemma, one needs to be able to efficiently construct a partition satisfying the conditions of the lemma. Frieze and Kannan also showed that this task can be performed in *randomized* $O(n^2)$ time. Alon and Naor [3] have shown that one can construct such a partition in deterministic polynomial time. The algorithm of Alon and Naor [3] requires solving a semi-definite program (SDP) and hence is not very efficient¹. The fast boolean matrix multiplication of Bansal and Williams [4] applies the randomized algorithm of [9,10]

¹ In fact, after solving the SDP, the algorithm of [3] needs time $O(n^3)$ to round the SDP solution.

for constructing an FK-regular partition. In an attempt to derandomize their matrix multiplication algorithm, Williams [24] asked if one can construct an FK-regular partition in deterministic time $O(n^{3-c})$ for some $c > 0$. Our main result in this paper answers this question by exhibiting a deterministic $\tilde{O}(n^\omega)$ time algorithm. Furthermore, as part of the design of this algorithm, we also show that one can find an approximation² to the first eigenvalue of a symmetric matrix in *deterministic* time $\tilde{O}(n^\omega)$.

Besides the above algorithmic motivation for our work, a further combinatorial motivation comes from the study of pseudo-random structures. Different notions of pseudo-randomness have been extensively studied in the last decade, both in theoretical computer science and in discrete mathematics. A key question that is raised in such cases is: Does there exist a *deterministic* condition which guarantees that a certain structure (say, graph or boolean function) behaves like a typical *random* structure? A well known result of this type is the discrete Cheeger’s inequality [1], which relates the expansion of a graph to the spectral gap of its adjacency matrix. Other results of this type relate the pseudo-randomness of functions over various domains to certain norms (the so-called Gowers norms). We refer the reader to the surveys of Gowers [12] and Trevisan [22] for more examples and further discussion on different notions of pseudo-randomness. An FK-regular partition is useful since it gives a pseudo-random description of a graph. Hence, it is natural to ask if one can characterize this notion of pseudo-randomness using a deterministic condition. The work of Alon and Naor [3] gives a condition which can be checked in polynomial time. However, as we mentioned before, verifying this condition requires one to solve a semi-definite program and is thus not efficient. In contrast, our main result in this paper gives a deterministic condition for FK-regularity which can be stated very simply and checked very efficiently.

1.2 The Main Result

We start with more precise definitions related to the regularity lemma. For a pair of subsets $A, B \subseteq V(G)$ in a graph $G = (V, E)$, let $e(A, B)$ denote the number of edges between A and B , counting each of the edges contained in $A \cap B$ twice. The density $d(A, B)$ is defined to be $d(A, B) = \frac{e(A, B)}{|A||B|}$. We will frequently deal with a partition of the vertex set $\mathcal{P} = \{V_1, V_2, \dots, V_k\}$. The *order* of such a partition is the number of sets V_i (k in the above partition). A partition is *equitable* if all sets are of size $\lfloor n/k \rfloor$ or $\lceil n/k \rceil$. We will use the shorthand notation for density across parts, $d_{ij} = d(V_i, V_j)$ whenever $i \neq j$. Also, we set $d_{ii} = 0$ for all i .

The key notion in Szemerédi’s regularity lemma [21] is the following: Let A, B be disjoint sets of vertices. We say that (A, B) is ε -regular if $|d(A, B) - d(A', B')| \leq \varepsilon$ for all $A' \subseteq A$ and $B' \subseteq B$ satisfying $|A'| \geq \varepsilon|A|$ and $|B'| \geq \varepsilon|B|$. It is not hard to show (see [15]) that ε -regular bipartite graphs behave like random graphs in many ways. Szemerédi’s Regularity Lemma [20] states that given $\varepsilon > 0$

² The necessity of approximation when dealing with eigenvalues is due to the non-existence of algebraic roots of high degree polynomials.

there is a constant $T(\varepsilon)$, such that the vertex set of any graph $G = (V, E)$ can be partitioned into k equitable sets V_1, \dots, V_k , where $k \leq T(\varepsilon)$ and all but εk^2 of the pairs (i, j) are such that (V_i, V_j) is ε -regular.

One of the useful aspects of an ε -regular partition of a graph is that it allows one to estimate the number of edges in certain partitions of G . For example, given an ε -regular partition, one can estimate the value of the Max-Cut in G within an error of εn^2 , in time that depends only on the order of the partition (and independent of the order of G !). Hence, one would like the order of the partition to be relatively small. However, as we have mentioned above, Gowers [13] has shown that there are graphs whose ε -regular partitions have size at least $\text{Tower}(1/\varepsilon^{1/16})$, namely a tower of exponents of height $1/\varepsilon^{1/16}$.

To remedy this, Frieze and Kannan [9,10] introduced the following relaxed notion of regularity, which we will call ε -FK-regularity.

Definition 1 (ε -FK-regular). *Let $\mathcal{P} = \{V_1, V_2, \dots, V_k\}$ be a partition of $V(G)$. For subsets $S, T \subseteq V$ and $1 \leq i \leq k$, let $S_i = S \cap V_i$ and $T_i = T \cap V_i$. Define $\Delta(S, T)$ for subsets $S, T \subseteq V$ as follows:*

$$\Delta(S, T) = e(S, T) - \sum_{i \neq j} d_{ij} |S_i| |T_j|. \tag{1}$$

The partition \mathcal{P} is said to be ε -FK-regular if it is equitable and

$$\text{for all subsets } S, T \subseteq V, \quad |\Delta(S, T)| \leq \varepsilon n^2. \tag{2}$$

If $|\Delta(S, T)| > \varepsilon n^2$ then S, T are said to be witnesses to the fact that \mathcal{P} is not ε -FK-regular.

One can think of Szemerédi’s regularity as dividing the graph into parts such that across most of the parts the graph looks like a random graph. In FK-regularity, we just want to partition the graph so that any cut of the graph contains roughly the “expected” number of edges as dictated by the densities d_{ij} . Another way to think about FK-regularity is that we want the bipartite graphs to be ε -regular (in the sense of Szemerédi) only on average.

The main novelty in this (weaker³) notion of regularity is that it allows one to compute useful statistics on the graph (such as estimating Max-Cut) while at the same time having the property that any graph can be partitioned into an ε -FK-regular partition of order $2^{100/\varepsilon^2}$, which is drastically smaller than the tower-type order of a Szemerédi partition. This was proved by Frieze and Kannan in [9,10] where they also gave several algorithmic applications of their version of the regularity lemma. As we have mentioned before, Frieze and Kannan also [9,10] proved that one can construct an ε -FK regular partition of a graph in randomized time $O(n^2)$. Our main result in this paper is the following deterministic algorithmic version of the FK-regularity lemma which answers a question of Williams [24].

³ It is not hard to see that an ε -regular partition (in the sense of Szemerédi’s lemma) is indeed ε -FK-regular.

Theorem 1 (Main Result). *Given $\varepsilon > 0$ and an n vertex graph $G = (V, E)$, one can find in deterministic time $O\left(\frac{1}{\varepsilon^8}n^\omega \log \log n\right)$ an ε -FK-regular partition of G of order at most $2^{10^8/\varepsilon^7}$.*

1.3 Paper Overview

The rest of the paper is organized as follows. As we have mentioned earlier, the relation between pseudo-random properties and spectral properties of graphs goes back to the Cheeger’s Inequality [1]. Furthermore, it was shown in [11] that one can characterize the notion of Szemerédi’s regularity using a spectral condition. In Section 2 we introduce a spectral condition for ε -FK-regularity and show that it characterizes this property. In order to be able to check this spectral condition efficiently, one has to be able to approximately compute the first eigenvalue of a matrix. Hence, in Section 3 we show that this task can be carried out in deterministic time $\tilde{O}(n^\omega)$. We use a deterministic variant of the randomized power iteration method. Since we could not find a reference for this, we include the proof for completeness. Finally, Section 4 contains some concluding remarks and open problems. As in other algorithmic versions of regularity lemmas, the non-trivial task is that of checking whether a partition is regular, and if it is not, then finding sets S, T which violate this property (recall Definition 1). This key result is stated in Corollary 1. Hence, due to space limitations, we omit the (somewhat routine) process of deducing Theorem 1 from Corollary 1 and defer it to the full version of this paper.

2 A Spectral Condition for FK-Regularity

In this section we introduce a spectral condition which “characterizes” partitions which are ε -FK regular. Actually, the condition will allow us to quickly distinguish between partitions that are ε -FK regular from partitions that are not $\varepsilon^3/1000$ -FK regular. As we will show later on, this is all one needs in order to efficiently construct an ε -FK regular partition. Our spectral condition relies on the following characterization of eigenvalues of a matrix. We omit the proof of this standard fact.

Lemma 1 (First eigenvalue). *For a diagonalizable matrix M , the absolute value of the first eigenvalue $\lambda_1(M)$ is given by the following:*

$$|\lambda_1(M)| = \max_{\|\mathbf{x}\|=\|\mathbf{y}\|=1} \mathbf{x}^T M \mathbf{y}.$$

We say that an algorithm computes a δ -approximation to the first eigenvalue of a matrix M if it finds two unit vectors \mathbf{x}, \mathbf{y} achieving $\mathbf{x}^T M \mathbf{y} \geq (1 - \delta)|\lambda_1(M)|$. Our goal in this section is to prove the following theorem.

Theorem 2. *Suppose there is an $S(n)$ time algorithm for computing a $1/2$ -approximation of the first eigenvalue of a symmetric $n \times n$ matrix. Then there is an $O(n^2 + S(n))$ time algorithm which given $\varepsilon > 0$, and a partition \mathcal{P} of the vertices of an n -vertex graph $G = (V, E)$, does one of the following:*

1. Correctly states that \mathcal{P} is ε -FK-regular.
2. Produces sets S, T which witness the fact that \mathcal{P} is not $\varepsilon^3/1000$ -FK-regular.

Let A be the adjacency matrix of the graph $G = (V, E)$, where $V = \{1, 2, \dots, n\} = [n]$. Let $S, T \subseteq V$ be subsets of the vertices and $\mathbf{x}_S, \mathbf{x}_T$ denote the corresponding indicator vectors. We would like to test if a partition $\mathcal{P} = V_1, \dots, V_k$ of V is a ε -FK-regular partition. We define a matrix $D = D(\mathcal{P})$ in the following way. Let $1 \leq i, j \leq n$ and suppose vertex i belongs V_{l_i} in \mathcal{P} and vertex j belongs to V_{l_j} , for some $1 \leq l_i, l_j \leq k$. Then the (i, j) th entry of D is given by $D_{ij} = d_{l_i l_j}$. Thus the matrix D is a block matrix (each block corresponding to the parts in the partition), where each block contains the same value at all positions, the value being the density of edges corresponding to the two parts. Now define $\Delta = A - D$. For $S, T \subseteq V$ and an $n \times n$ matrix M , define

$$M(S, T) = \sum_{i \in S, j \in T} M(i, j) = \mathbf{x}_S^T M \mathbf{x}_T.$$

Notice that for the matrix Δ , the above definition coincides with \square :

$$\Delta(S, T) = A(S, T) - D(S, T) = e(S, T) - \sum_{i, j} d_{ij} |S_i| |T_j|,$$

where $S_i = S \cap V_i$ and $T_j = T \cap V_j$. Summarizing, \mathcal{P} is an ε -FK-regular partition of V if and only if for all $S, T \subseteq V$, we have $|\Delta(S, T)| \leq \varepsilon n^2$.

Let $G = (V, E)$ be an n -vertex graph, let \mathcal{P} be a partition of $V(G)$ and let Δ be the matrix defined above. Notice that by construction, Δ is a symmetric matrix and so it can be diagonalized with real eigenvalues. Lemmas \square and \square below will establish a relation between the first eigenvalue of Δ and the FK-regularity properties of \mathcal{P} .

Lemma 2. *If $|\lambda_1(\Delta)| \leq \gamma n$ then \mathcal{P} is γ -FK-regular.*

Proof. We prove this in contrapositive. Suppose \mathcal{P} is not γ -FK-regular and let S, T be two sets witnessing this fact, that is, satisfying $|\Delta(S, T)| = |\mathbf{x}_S^T \Delta \mathbf{x}_T| > \gamma n^2$. Normalizing the vectors \mathbf{x}_S and \mathbf{x}_T , we have $\tilde{\mathbf{x}}_S = \mathbf{x}_S / \|\mathbf{x}_S\| = \mathbf{x}_S / \sqrt{|S|}$ and $\tilde{\mathbf{x}}_T = \mathbf{x}_T / \|\mathbf{x}_T\| = \mathbf{x}_T / \sqrt{|T|}$. We get

$$|\tilde{\mathbf{x}}_S^T \Delta \tilde{\mathbf{x}}_T| > \gamma n^2 / (\sqrt{|S|} |T|) \geq \gamma n,$$

where the last inequality follows since $|S|, |T| \leq n$. By the characterization of the first eigenvalue, we have that $|\lambda_1(\Delta)| > \gamma n$. \square

Lemma 3. *Suppose two vectors $\mathbf{p}, \mathbf{q} \in [-1, 1]^n$ satisfying $\mathbf{p}^T \Delta \mathbf{q} > 0$ are given. Then, in deterministic time $O(n^2)$, we can find sets $S, T \subseteq [n]$ satisfying $|\Delta(S, T)| \geq \frac{1}{4} \mathbf{p}^T \Delta \mathbf{q}$.*

Proof. Let us consider the positive and negative parts of the vectors \mathbf{p} and \mathbf{q} . Of the four combinations, $(\mathbf{p}^+, \mathbf{q}^+)$, $(\mathbf{p}^+, \mathbf{q}^-)$, $(\mathbf{p}^-, \mathbf{q}^+)$ and $(\mathbf{p}^-, \mathbf{q}^-)$, at least one

pair should give rise to a product at least $\mathbf{p}^T \Delta \mathbf{q}/4$. Let us call this pair the good pair. Suppose the good pair is $\mathbf{p}^+, \mathbf{q}^+$. Let Δ_i, Δ^j denote respectively the i th row and j th column of Δ . We can write $(\mathbf{p}^+)^T \Delta \mathbf{q}^+ = \sum_i p_i^+ \langle \Delta_i, \mathbf{q}^+ \rangle$. Compute the n products, $\langle \Delta_i, \mathbf{q}^+ \rangle$. We put vertex i in S if and only if $\langle \Delta_i, \mathbf{q}^+ \rangle \geq 0$. For this choice of S , we have $\mathbf{x}_S^T \Delta \mathbf{q}^+ \geq (\mathbf{p}^+)^T \Delta \mathbf{q}^+$. Similarly as before, we have $\mathbf{x}_S^T \Delta \mathbf{q}^+ = \sum_j q_j^+ \langle \mathbf{x}_S, \Delta^j \rangle$, therefore depending on the signs of $\langle \mathbf{x}_S, \Delta^j \rangle$, we define whether j belongs to T . Thus we get sets S, T such that $\Delta(S, T) = \mathbf{x}_S^T \Delta \mathbf{x}_T \geq (\mathbf{p}^+)^T \Delta \mathbf{q}^+ \geq \mathbf{p}^T \Delta \mathbf{q}/4$. Notice that this rounding takes $O(n^2)$ time, since we need to perform $2n$ vector products, each of which takes $O(n)$ time.

If exactly one of \mathbf{p}^- or \mathbf{q}^- is part of the good pair, then we could replicate the above argument in a similar manner. Thus we would get $\Delta(S, T) \leq -\mathbf{p}^T \Delta \mathbf{q}/4$. If the good pair is $(\mathbf{p}^-, \mathbf{q}^-)$, we would again get $\Delta(S, T) \geq \mathbf{p}^T \Delta \mathbf{q}/4$. \square

Lemma 4. *If $|\lambda_1(\Delta)| > \gamma n$, then \mathcal{P} is not $\gamma^3/108$ -FK-regular. Furthermore, given unit vectors \mathbf{x}, \mathbf{y} satisfying $\mathbf{x}^T \Delta \mathbf{y} > \gamma n$, one can find sets S, T witnessing this fact in deterministic time $O(n^2)$.*

Proof. As per the previous observation, it is enough to find sets S, T such that $|\Delta(S, T)| > \gamma^3 n^2/108$. By Lemma 3, it is enough to find vectors \mathbf{p} and \mathbf{q} in $[-1, 1]^n$ satisfying $\mathbf{p}^T \Delta \mathbf{q} > \gamma^3 n^2/27$.

Suppose that $|\lambda_1(\Delta)| > \gamma n$ and let \mathbf{x}, \mathbf{y} satisfy $\|\mathbf{x}\| = \|\mathbf{y}\| = 1$ and $\mathbf{x}^T \Delta \mathbf{y} > \gamma n$. Let $\beta > 1$ (β will be chosen to be $3/\gamma$ later on) and define $\hat{\mathbf{x}}, \hat{\mathbf{y}}$ in the following manner:

$$\hat{x}_i = \begin{cases} x_i & \text{if } |x_i| \leq \frac{\beta}{\sqrt{n}} \\ 0 & \text{otherwise} \end{cases}, \quad \hat{y}_i = \begin{cases} y_i & \text{if } |y_i| \leq \frac{\beta}{\sqrt{n}} \\ 0 & \text{otherwise} \end{cases}.$$

We claim that

$$\hat{\mathbf{x}}^T \Delta \hat{\mathbf{y}} > (\gamma - 2/\beta)n. \quad (3)$$

To prove this, note that

$$\begin{aligned} \hat{\mathbf{x}}^T \Delta \hat{\mathbf{y}} &= \mathbf{x}^T \Delta \mathbf{y} - (\mathbf{x} - \hat{\mathbf{x}})^T \Delta \mathbf{y} - \hat{\mathbf{x}}^T \Delta (\mathbf{y} - \hat{\mathbf{y}}) \\ &> \gamma n - (\mathbf{x} - \hat{\mathbf{x}})^T \Delta \mathbf{y} - \hat{\mathbf{x}}^T \Delta (\mathbf{y} - \hat{\mathbf{y}}) \\ &\geq \gamma n - |(\mathbf{x} - \hat{\mathbf{x}})^T \Delta \mathbf{y}| - |\hat{\mathbf{x}}^T \Delta (\mathbf{y} - \hat{\mathbf{y}})|. \end{aligned}$$

Hence, to establish (3) it would suffice to bound $|(\mathbf{x} - \hat{\mathbf{x}})^T \Delta \mathbf{y}|$ and $|\hat{\mathbf{x}}^T \Delta (\mathbf{y} - \hat{\mathbf{y}})|$ from above by n/β . To this end, let $C(\mathbf{x}) = \{i : |x_i| \geq \beta/\sqrt{n}\}$, and note that since $\|\mathbf{x}\| = 1$ we have $|C(\mathbf{x})| \leq n/\beta^2$. Now define Δ' as

$$\Delta'_{ij} = \begin{cases} \Delta_{ij} & \text{if } i \in C(\mathbf{x}) \\ 0 & \text{otherwise} \end{cases}.$$

We now claim that the following holds.

$$\begin{aligned} |(\mathbf{x} - \hat{\mathbf{x}})^T \Delta \mathbf{y}| &= |(\mathbf{x} - \hat{\mathbf{x}})^T \Delta' \mathbf{y}| \leq \|(\mathbf{x} - \hat{\mathbf{x}})^T\| \|\Delta' \mathbf{y}\| \\ &\leq \|\Delta' \mathbf{y}\| \\ &\leq \|\Delta'\|_F \|\mathbf{y}\| \\ &= \|\Delta'\|_F \\ &\leq n/\beta. \end{aligned}$$

Indeed, the first inequality is Cauchy-Schwarz and in the second inequality we use the fact that $\|\mathbf{x} - \hat{\mathbf{x}}\| \leq \|\mathbf{x}\| = 1$. In the third inequality $\|\Delta'\|_F$ denotes $\sqrt{\sum_{i,j} (\Delta'_{ij})^2}$ and the inequality follows from Cauchy-Schwarz. The fourth line is an equality that follows from $\|\mathbf{y}\| = 1$. The last inequality follows from observing that since $|C(\mathbf{x})| \leq n/\beta^2$ the matrix Δ' has only n^2/β^2 non-zero entries, and each of these entries is of absolute value at most 1. It follows from an identical argument that $|\hat{\mathbf{x}}^T \Delta(\mathbf{y} - \hat{\mathbf{y}})| \leq n/\beta$, thus proving (3). After rescaling $\hat{\mathbf{x}}$ and $\hat{\mathbf{y}}$, we get

$$((\sqrt{n}/\beta)\hat{\mathbf{x}})^T \Delta((\sqrt{n}/\beta)\hat{\mathbf{y}}) > (\gamma - 2/\beta)n^2/\beta^2 .$$

Setting $\beta = 3/\gamma$ so that $(\gamma - 2/\beta)/\beta^2$ is maximized, the right hand side of the inequality is $\gamma^3 n^2/27$. Now that we have the necessary vectors $\mathbf{p} = (\sqrt{n}/\beta)\hat{\mathbf{x}}$ and $\mathbf{q} = (\sqrt{n}/\beta)\hat{\mathbf{y}}$, an application of Lemma 3 completes the proof. \square

Proof (Proof of Theorem 2). We start with describing the algorithm. Given $G = (V, E)$, $\varepsilon > 0$ and a partition \mathcal{P} of $V(G)$, the algorithm first computes the matrix $\Delta = A - D$ (in time $O(n^2)$) and then computes unit vectors \mathbf{x}, \mathbf{y} satisfying $\mathbf{x}^T \Delta \mathbf{y} \geq \frac{1}{2} |\lambda_1(\Delta)|$ (in time $S(n)$). If $\mathbf{x}^T \Delta \mathbf{y} \leq \varepsilon n/2$ the algorithm declares that \mathcal{P} is ε -FK-regular, and if $\mathbf{x}^T \Delta \mathbf{y} > \varepsilon n/2$ it declares that \mathcal{P} is not $\varepsilon^3/1000$ -FK-regular and then uses the $O(n^2)$ time algorithm of Lemma 4 in order to produce sets S, T which witness this fact. The running time of the algorithm is clearly $O(n^2 + S(n))$.

As to the algorithm’s correctness, if $\mathbf{x}^T \Delta \mathbf{y} \leq \varepsilon n/2$ then since we have computed a $1/2$ -approximation for $|\lambda_1(\Delta)|$ this means that $|\lambda_1(\Delta)| \leq \varepsilon n$. Hence, by Lemma 2 we have that \mathcal{P} is indeed ε -FK-regular. If $\mathbf{x}^T \Delta \mathbf{y} > \varepsilon n/2$ then by Lemma 4 we are guaranteed to obtain sets S, T which witness the fact that \mathcal{P} is not $\varepsilon^3/(108 \cdot 8) \geq \varepsilon^3/1000$ -FK-regular. \square

3 Finding the First Eigenvalue Deterministically

In order to efficiently apply Theorem 2 from the previous section, we will need an efficient algorithm for approximating the first eigenvalue of a symmetric matrix. Such an algorithm is guaranteed by the following theorem:

Theorem 3. *Given an $n \times n$ symmetric matrix H , and parameter $\delta > 0$, one can find in deterministic time $O(n^\omega \log(\frac{1}{\delta} \log(\frac{n}{\delta})))$ unit vectors \mathbf{x}, \mathbf{y} satisfying*

$$\mathbf{x}^T H \mathbf{y} \geq (1 - \delta) |\lambda_1(H)| .$$

Setting $H = \Delta$ and $\delta = 1/2$ in Theorem 3, and using Theorem 2 we infer the following corollary.

Corollary 1. *There is an $O(n^\omega \log \log n)$ time algorithm, which given $\varepsilon > 0$, an n -vertex graph $G = (V, E)$ and a partition \mathcal{P} of $V(G)$, does one of the following:*

1. *Correctly states that \mathcal{P} is ε -FK-regular.*
2. *Finds sets S, T which witness the fact that \mathcal{P} is not $\varepsilon^3/1000$ -FK-regular.*

As we have mentioned in Section 1, one can derive our main result stated in Theorem 1 from Corollary 1 using the proof technique of Szemerédi [21]. Hence we defer this part of the paper to the full version.

We also note that the proof of Theorem 3 can be modified to approximate the quantity $\max_{\|\mathbf{x}\|=\|\mathbf{y}\|=1} \mathbf{x}^T H \mathbf{y}$ for any matrix H . This quantity is the so-called first singular value of H . But since we do not need this for our specific application to FK-regularity, we state the theorem “only” for symmetric matrices H .

Getting back to the proof of Theorem 3 we first recall that for any matrix H we have $|\lambda_1(H)| = \sqrt{\lambda_1(H^2)}$ (notice that H^2 is positive semi-definite, so all its eigenvalues are non-negative). Hence, in order to compute an approximation to $|\lambda_1(H)|$, we shall compute an approximation to $\lambda_1(H^2)$. Theorem 3 will follow easily once we prove the following.

Theorem 4. *Given an $n \times n$ positive semi-definite matrix M , and parameter $\delta > 0$, there exists an algorithm that runs in $O(n^\omega \log(\frac{1}{\delta} \log(\frac{n}{\delta})))$ time and outputs a vector \mathbf{b} such that*

$$\frac{\mathbf{b}^T M \mathbf{b}}{\mathbf{b}^T \mathbf{b}} \geq (1 - \delta) \lambda_1(M).$$

Proof (Proof of Theorem 3). As mentioned above, $|\lambda_1(H)| = \sqrt{\lambda_1(H^2)}$. Since H^2 is positive semi-definite we can use Theorem 4 to compute a vector \mathbf{b} satisfying

$$\frac{\mathbf{b}^T M \mathbf{b}}{\mathbf{b}^T \mathbf{b}} = \hat{\lambda}_1 \geq (1 - \delta) \lambda_1(M).$$

We shall see that $\sqrt{\hat{\lambda}_1}$ is a $(1 - \delta)$ approximation to the first eigenvalue of H . To recover the corresponding vectors as in Lemma 1, notice that

$$\mathbf{b}^T M \mathbf{b} = \mathbf{b}^T H^2 \mathbf{b} = \|H \mathbf{b}\|^2 = \hat{\lambda}_1 \|\mathbf{b}\|^2 \implies \|H \mathbf{b}\| = \sqrt{\hat{\lambda}_1} \|\mathbf{b}\|.$$

Setting $\mathbf{x} = \frac{H \mathbf{b}}{\sqrt{\hat{\lambda}_1} \|\mathbf{b}\|}$ and $\mathbf{y} = \frac{\mathbf{b}}{\|\mathbf{b}\|}$, we obtain unit vectors \mathbf{x} and \mathbf{y} satisfying

$$\mathbf{x}^T H \mathbf{y} = \sqrt{\hat{\lambda}_1} \geq \sqrt{(1 - \delta) \lambda_1(H^2)} \geq (1 - \delta) |\lambda_1(H)|.$$

The main step that contributes to the running time is the computation of \mathbf{b} using Theorem 4 and hence the running time is $O(n^\omega \log(\frac{1}{\delta} \log(\frac{n}{\delta})))$, as needed. \square

We turn to prove Theorem 4. We shall apply the *power iteration method* to compute an approximation of the first eigenvalue of a positive semi-definite (PSD) matrix. Power iteration is a technique that can be used to compute the largest eigenvalues and is a very widely studied method. For instance, the paper [16] by Kuczyński and Woźniakowski has a very thorough analysis of the method. The earlier work of [18] shows that power iteration is much more effective with PSD matrices. A much simpler (albeit slightly weaker) analysis was given in [23].

A PSD matrix M has all nonnegative eigenvalues. The goal of power iteration is to find the first eigenvalue and the corresponding eigenvector of M . The basic

idea is that an arbitrary vector \mathbf{r} is taken, and is repeatedly multiplied with the matrix M . The vector \mathbf{r} can be seen as a decomposition into components along the direction of each of the eigenvectors of the matrix. With each iteration of multiplication by M , the component of \mathbf{r} along the direction of the first eigenvector gets magnified much more than the component of \mathbf{r} along the direction of the other eigenvectors. This is because the first eigenvalue is larger than the other eigenvalues. One of the key properties that is required of \mathbf{r} is that it has a nonzero component along the first eigenvector. We ensure that by using n different orthogonal basis vectors.

We first need the following key lemma.

Lemma 5. *Let M be a positive semi-definite matrix. Let $\mathbf{a} \in \mathbb{R}^n$ be a unit vector such that $|\langle \mathbf{v}_1, \mathbf{a} \rangle| \geq 1/\sqrt{n}$. Then, for every positive integer s and positive $\delta > 0$, for $\mathbf{b} = M^s \mathbf{a}$, we have*

$$\frac{\mathbf{b}^T M \mathbf{b}}{\mathbf{b}^T \mathbf{b}} \geq \lambda_1 \cdot \left(1 - \frac{\delta}{2}\right) \cdot \frac{1}{1 + n \left(1 - \frac{\delta}{2}\right)^{2s}},$$

where λ_1 denotes the first eigenvalue of M .

The proof of the above lemma is omitted due to lack of space. Now we are ready to analyze the power iteration algorithm and to prove Theorem 4.

Proof (Proof of Theorem 4). Consider the n canonical basis vectors, denoted by \mathbf{e}_i , for $i = 1, \dots, n$. We can decompose the first eigenvector \mathbf{v}_1 of M along these n basis vectors. Since \mathbf{v}_1 has norm 1, there must exist an i such that $|\langle \mathbf{v}_1, \mathbf{e}_i \rangle| \geq 1/\sqrt{n}$, by pigeonhole principle. We can perform power iteration of M , starting at these n basis vectors. We would get n output vectors, and for each output vector \mathbf{x} , we compute $\mathbf{x}^T M \mathbf{x} / (\mathbf{x}^T \mathbf{x})$, and choose the one which gives us the maximum. By Lemma 5, one of these output vectors \mathbf{x} is such that

$$\frac{\mathbf{x}^T M \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \geq \lambda_1 \cdot \left(1 - \frac{\delta}{2}\right) \cdot \frac{1}{1 + n \left(1 - \frac{\delta}{2}\right)^{2s}}.$$

If we use $s = O\left(\frac{1}{\delta} \log\left(\frac{n}{\delta}\right)\right)$, we can eliminate the factor n in the denominator, and the denominator would become $(1 + \frac{\delta}{2})$, giving us an estimate of at least $\lambda_1 \cdot (1 - \delta)$, which is what we required.

To perform the n power iterations efficiently, consider taking the s th power of M . Let $N = M^s = M^s \cdot I$. We can think of this as performing n power iteration algorithms in parallel, each one starting with a different canonical basis vector. For each vector $\mathbf{x} = M^s \mathbf{e}_i$, we need to compute $(\mathbf{x}^T M \mathbf{x}) / (\mathbf{x}^T \mathbf{x})$. For that we compute the products $P = N^T M N$ and $Q = N^T N$. To get the \mathbf{x} that maximizes the answer, we choose $\max\{P_{ii}/Q_{ii} : 1 \leq i \leq n\}$. The maximized ratio is the approximation to the first eigenvalue, and the corresponding i th column of N is the estimation of the maximizing eigenvector.

For the running time analysis, the most time consuming step is taking the s th power of the matrix M . Using repeated squaring, this can be done in $2 \log s$ steps. Since we need $s = O\left(\frac{1}{\delta} \log\left(\frac{n}{\delta}\right)\right)$, the running time required by the entire algorithm is bounded by $O\left(n^\omega \log\left(\frac{n}{\delta}\right)\right)$. □

4 Concluding Remarks and Open Problems

- We have designed an $\tilde{O}(n^\omega)$ time deterministic algorithm for constructing an ε -FK regular partition of a graph. It would be interesting to see if one can design an $O(n^2)$ time deterministic algorithm for this problem. We recall that it is known [14] that one can construct an ε -regular partition of a graph (in the sense of Szemerédi) in deterministic time $O(n^2)$. This algorithm relies on a *combinatorial* characterization of ε -regularity using a co-degree condition. Such an approach might also work for ε -FK regularity, though the co-degree condition in this case might be more involved.
- We have used a variant of the power iteration method to obtain an $\tilde{O}(n^\omega)$ time algorithm for computing an approximation to the first eigenvalue of a symmetric matrix. It would be interesting to see if the running time can be improved to $O(n^2)$. Recall that our approach relies on (implicitly) running n power-iterations in parallel, each of which on one of the n standard basis vectors. One approach to design an $\tilde{O}(n^2)$ algorithm would be to show that given an $n \times n$ PSD matrix M , one can find in time $O(n^2)$ a set of $n^{0.1}$ unit vectors such that one of the vectors \mathbf{v} in the set has an inner product at least $1/\text{poly}(n)$ with the first eigenvector of M . If this can indeed be done, then one replace the fast matrix multiplication algorithm for square matrices that we use in the algorithm, by an algorithm of Coppersmith [6] that multiplies an $n \times n$ matrix by an $n \times n^{0.1}$ matrix in time $\tilde{O}(n^2)$. The modified algorithm would then run in $\tilde{O}(n^2)$.
- Designing an $\tilde{O}(n^2)$ algorithm for finding the first eigenvalue of a PSD matrix would of course yield an $\tilde{O}(n^2)$ algorithm for finding an ε -FK regular partition of a graph (via Theorem 2). In our case, it is enough to find the first eigenvalue up to a δn additive error. So another approach to getting an $\tilde{O}(n^2)$ algorithm for ε -FK regularity would be to show that in time $\tilde{O}(n^2)$ we can approximate the first eigenvalue up to an *additive* error of δn . It might be easier to design such a $\tilde{O}(n^2)$ algorithm than for the multiplicative approximation discussed in the previous item.

Acknowledgement. We would like to thank Nikhil Srivastava for helpful initial discussions.

References

1. Alon, N.: Eigenvalues and expanders. *Combinatorica* 6, 83–96 (1986), doi:10.1007/BF02579166
2. Alon, N., Duke, R.A., Lefmann, H., Rödl, V., Yuster, R.: The algorithmic aspects of the regularity lemma. *J. Algorithms* 16, 80–109 (1994)
3. Alon, N., Naor, A.: Approximating the cut-norm via Grothendieck’s inequality. In: *Proceedings of the Thirty-Sixth Annual ACM Symposium on Theory of Computing, STOC 1984*, pp. 72–80. ACM, New York (2004)
4. Bansal, N., Williams, R.: Regularity lemmas and combinatorial algorithms. In: *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009*, pp. 745–754. IEEE Computer Society, Washington, DC, USA (2009)

5. Bhatia, R.: *Matrix Analysis*. Springer, New York (1997)
6. Coppersmith, D.: Rapid multiplication of rectangular matrices. *SIAM J. Computing* 11, 467–471 (1982)
7. Coppersmith, D., Winograd, S.: Matrix multiplication via arithmetic progressions. In: *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing, STOC 1987*, pp. 1–6. ACM, New York (1987)
8. Duke, R.A., Lefmann, H., Rödl, V.: A fast approximation algorithm for computing the frequencies of subgraphs in a given graph. *SIAM J. Comput.* 24(3), 598–620 (1995)
9. Frieze, A., Kannan, R.: The regularity lemma and approximation schemes for dense problems. In: *Annual IEEE Symposium on Foundations of Computer Science*, p. 12 (1996)
10. Frieze, A., Kannan, R.: Quick approximation to matrices and applications. *Combinatorica* 19, 175–220 (1999)
11. Frieze, A., Kannan, R.: A simple algorithm for constructing Szemerédi’s regularity partition. *Electr. J. Comb.* 6 (1999) (electronic)
12. Gowers, W.T.: Quasirandomness, counting and regularity for 3-uniform hypergraphs. *Comb. Probab. Comput.* 15, 143–184 (2006)
13. Gowers, W.T.: Lower bounds of tower type for Szemerédi’s uniformity lemma. *Geometric And Functional Analysis* 7, 322–337 (1997)
14. Kohayakawa, Y., Rödl, V., Thoma, L.: An optimal algorithm for checking regularity. *SIAM J. Comput.* 32, 1210–1235 (2003); Earlier version in *SODA 2002*
15. Komlós, J., Shokoufandeh, A., Simonovits, M., Szemerédi, E.: The regularity lemma and its applications in graph theory, pp. 84–112. Springer-Verlag New York, Inc., New York (2002)
16. Kuczyński, J., Woźniakowski, H.: Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start. *SIAM Journal on Matrix Analysis and Applications* 13(4), 1094–1122 (1992)
17. Lovász, L.: Very large graphs. In: Jerison, D., Mazur, B., Mrowka, T., Schmid, W., Stanley, R., Yau, S.T. (eds.) *Current Developments in Mathematics* (2008)
18. O’Leary, D.P., Stewart, G.W., Vandergraft, J.S.: Quasirandomness, counting and regularity for 3-uniform hypergraphs. *Mathematics of Computation* 33, 1289–1292 (1979)
19. Rödl, V., Schacht, M.: Regularity lemmas for graphs. In: *Fete of Combinatorics and Computer Science. Bolyai Society Mathematical Studies*, vol. 20, pp. 287–325. Springer, Heidelberg
20. Szemerédi, E.: On sets of integers containing no k elements in arithmetic progressions. *Polska Akademia Nauk. Instytut Matematyczny. Acta Arithmetica* 27, 199–245 (1975)
21. Szemerédi, E.: Regular partitions of graphs. In: *Problèmes combinatoires et théorie des graphes (Colloq. Internat. CNRS, Univ. Orsay, Orsay, 1976)*, pp. 399–401. Éditions du Centre National de la Recherche Scientifique (CNRS), Paris (1978)
22. Trevisan, L.: Pseudorandomness in computer science and in additive combinatorics. In: *An Irregular Mind. Bolyai Society Mathematical Studies*, vol. 21, pp. 619–650. Springer, Heidelberg (2010)
23. Trevisan, L.: Lecture notes, <http://lucatrevisan.wordpress.com/>
24. Williams, R.: Private Communication (2009)

Dense Locally Testable Codes Cannot Have Constant Rate and Distance

Irit Dinur^{1,*} and Tali Kaufman^{2,**}

¹ Weizmann Institute, Rehovot, ISRAEL
irit.dinur@weizmann.ac.il

² Bar Ilan University, Ramat Gan and Weizmann Institute, Rehovot, ISRAEL
kaufmant@mit.edu

Abstract. A q -query locally testable code (LTC) is an error correcting code that can be tested by a randomized algorithm that reads at most q symbols from the given word. An important question is whether there exist LTCs that have the c^3 property: constant rate, constant relative distance, and that can be tested with a constant number of queries. Such LTCs are sometimes referred to as “asymptotically good”.

We show that *dense* LTCs cannot be c^3 . The *density* of a tester is roughly the average number of distinct local views in which a coordinate participates. An LTC is *dense* if it has a tester with density $\omega(1)$.

More precisely, we show that a 3-query locally testable code with a tester of density $\omega(1)$ cannot be c^3 . Furthermore, we show that a q -locally testable code ($q > 3$) with a tester of density $\omega(1)n^{q-2}$ cannot be c^3 . Our results hold when the tester has the following two properties:

- (no weights:) Every q -tuple of queries occurs with the same probability.
- (‘last-one-fixed’:) In every q -query ‘test’ of the tester, the value to any $q - 1$ of the symbols determines the value of the last symbol. (Linear codes have constraints of this type).

We also show that several natural ways to quantitatively improve our results would already resolve the general c^3 question, i.e. also for non-dense LTCs.

1 Introduction

An error correcting code is a set $\mathcal{C} \subset \Sigma^n$. The rate of the code is $\log |\mathcal{C}| / n$ and its (relative) distance is the minimal Hamming distance between two different codewords $x, y \in \mathcal{C}$, divided by n . We only consider codes with distance $\Omega(1)$.

A code is called *locally testable* with q queries if it has a *tester*, which is a randomized algorithm with oracle access to the received word x . The tester reads at most q symbols from x and based on this local view decides if $x \in \mathcal{C}$ or

* Work supported by ISF grant 1179/09, BSF grant 2008293, and ERC starting grant 239985.

** Work supported by the Alon fellowship.

not. It should accept codewords with probability one, and reject words that are far (in Hamming distance) from the code with noticeable probability. The tester has parameters (τ, ϵ) if

$$\forall x \in \Sigma^n, \text{dist}(x, \mathcal{C}) \geq \tau \implies \Pr[\text{Tester rejects } x] \geq \epsilon$$

Locally Testable Codes (henceforth, LTCs) have been studied extensively in recent years. A priori, even the existence of LTCs is not trivial. The Hadamard code is a celebrated example of an LTC, yet it is highly “inefficient” in the sense of having very low rate $(\log n/n)$. Starting with the work of Goldreich and Sudan [GS06], several other efficient constructions of LTCs have been given. The best known rate for LTCs is $1/\log^{O(1)} n$, and these codes have 3-query testers [BS05, Din07, Mei09]. The failure to construct constant rate, constant distance LTCs testable with constant number of queries leads to one of the main open questions in the area: are there LTCs that are c^3 , i.e. constant rate constant distance and testable with a constant number of queries (such LTCs are sometimes called in the literature “asymptotically good”). The case of two queries with LOF tests has been studied in [BSGS03]. However, the case of $q \geq 3$ is much more interesting and still quite open.

Dense Testers. In this work we make progress on a variant of the c^3 question. We show that LTCs with so-called dense testers, cannot be c^3 .

The density of a tester is roughly the average number, per-coordinate, of distinct local views that involve that coordinate. More formally, every tester gives rise to a constraint-hypergraph $H = ([n], E)$ whose vertices are the n coordinates of the codeword, and whose hyperedges correspond to all possible local views of the tester. Each hyperedge $h \in E$ is also associated with a constraint, i.e. with a Boolean function $f_h : \Sigma^q \rightarrow \{0, 1\}$ that determines whether the tester accepts or rejects on that local view. For a given string $x \in \Sigma^n$, we denote by x_h the substring obtained by restricting x to the coordinates in the hyperedge h . The value of $f_h(x_h)$ determines if the string x falsifies the constraint or not.

Definition 1 (The test-hypergraph, density). *Let $\mathcal{C} \subseteq \Sigma^n$ be a code, let $q \in \mathbb{N}$ and $\epsilon > 0$.*

Let H be a constraint hypergraph with hyperedges of size at most q . H is an (τ, ϵ) -test-hypergraph for \mathcal{C} if

- For every $x \in \mathcal{C}$ and every $h \in E$, $f_h(x_h) = 1$
- For every $x \in \Sigma^n$,

$$\text{dist}(x, \mathcal{C}) \geq \tau \implies \Pr_{h \in E} [f_h(x_h) = 0] \geq \epsilon$$

where $\text{dist}(x, y)$ denotes relative Hamming distance, i.e., the fraction of coordinates on which x differs from y .

Finally, the density of H is simply the average degree, $|E|/n$.

The hypergraph describes a tester that selects one of the hyperedges uniformly at random. Observe that we disallow weights on the hyperedges. This will be discussed further below.

Goldreich and Sudan [GS06] proved that every tester with density $\Omega(1)$ can be made into a “sparse” tester with density $O(1)$ by randomly eliminating each hyper-edge with suitable probability. This means that a code can have both dense and sparse testers at the same time. Hence, we define a code to have *density* $\geq d$ if it has a tester with density d . We stress, that *density* in this work is a property of the code rather than of the tester. In this work we show that the existence of certain dense testers restricts the rate of the code.

We say that an LTC is *sparse* if it has no tester whose density is $\omega(1)$. We do not know of any LTC that is sparse. Thus, our work here provides some explanation for the bounded rate that known LTCs achieve.

In fact, one wonders whether density is an inherent property of LTCs. The intuition for such a claim is that in order to be locally testable the code seems to require a certain redundancy among the local tests, a redundancy which might be translated into density. If one were to prove that every LTC is dense, then it would rule out, by combination with our work, the existence of c^3 -LTCs [1].

In support of this direction we point to the work of the second author with co-authors (Ben-Sasson et al [BSGK⁺10]) where it is shown that every linear LTC (even with bounded rate) must have some non-trivial density: They show that no linear LTC can be tested only with tests that form a basis to the dual code. Moreover, any tester must have a significantly larger number of tests. Namely some constant density is required in every tester of such an LTC.

1.1 Our Results

We bound the rate of LTCs with dense testers. We only consider testers whose constraints have the “last-one-fixed” (LOF) property, i.e. the value to any $q - 1$ symbols determine the value of the last symbol. Note for instance that any linear constraint has this property.

We present different bounds for the case $q = 3$ and the case $q > 3$ where q denotes the number of queries.

Theorem 1. *Let $C \subseteq \{0, 1\}^n$ be a 3-query LTC with distance δ , and let H be an $(\delta/3, \varepsilon)$ -test-hypergraph with density d and LOF constraints. Then, the rate of C is at most $O(1/d^{1/2})$ (where the O -notation hides dependencies on δ, ε).*

For the case of $q > 3$ queries we have the following result whose proof, which is similar to the $q = 3$ case, is omitted due to space limitations.

Theorem 2. *Let $C \subseteq \{0, 1\}^n$ be a q -query LTC with distance δ , and let H be an $(\delta/2, \varepsilon)$ -test-hypergraph with density Δ , where $\Delta = dn^{q-2}$, and LOF constraints. Then, the rate of C is at most $O(1/d)$.*

¹ This is slightly imprecise, since our results only hold for uniform and LOF testers. So for this approach to work one would need to remove these extra conditions from our results.

Extensions. In this preliminary version we assume that the alphabet is Boolean, but the results easily extend to any finite alphabet Σ . It may also be possible to get rid of the “last-one-fixed” restriction on the constraints, but this remains to be worked out.

Improvements. We show that several natural ways of improving our results will already resolve the ‘bigger’ question of ruling out c^3 -LTCs.

- In this work we only handle *non-weighted* testers, i.e., where the hypergraph has no weights. In general a tester can put different weights on different hyperedges. This is sometimes natural when combining two or more “types” of tests each with certain probability. This limitation cannot be eliminated altogether, but may possibly be addressed via a more refined definition of density. See further discussion Section 3.3.
- In Theorem 1 we prove that $\rho \leq O(1/d^{0.5})$. We show that any improvement of the 0.5 exponent (say to $0.5 + \varepsilon$) would again rule out the existence of c^3 -LTCs, see Lemma 4.
- In Theorem 2 we bound the rate only when the density is very high, namely, $\omega(n^{q-2})$. We show, in Lemma 5, that any bound for density $O(n^{q-3})$ would once more rule out the existence of c^3 -LTCs. It seems that our upper bound of $\omega(n^{q-2})$ can be improved to $\omega(n^{q-3})$, possibly by arguments similar to those in the proof of Theorem 1.

Related Work. In the course of writing our result we learned that Eli Ben-Sasson and Michael Viderman have also been studying the connection between density and rate and have obtained related results, through seemingly different methods.

2 Moderately Dense 3-query LTCs Cannot Be c^3

In this section we prove Theorem 1 which we now recall:

Theorem 1. *Let $C \subseteq \{0, 1\}^n$ be a 3-query LTC with distance δ , and let H be an $(\delta/3, \varepsilon)$ -test-hypergraph with density d and LOF constraints. Then, the rate of C is at most $O(1/d^{1/2})$.*

In order to prove the main theorem, we consider the hypergraph $H = (V, E(H))$ whose vertices are the coordinates of the code, and whose hyper-edges correspond to the different tests of the tester. By assumption, H has dn distinct hyper-edges. We describe an algorithm in Figure 1 for assigning values to coordinates of a codeword, and show that a codeword is determined using $k = O(\frac{n}{d^{1/2}})$ bits.

We need the following definition. For a partition (A, B) of the vertices V of H , we define the graph $G_B = (A, E)$ where

$$E = \{ \{a_1, a_2\} \subset A \mid \exists b \in B, \{a_1, a_2, b\} \in E(H) \}.$$

A single edge $\{a_1, a_2\} \in E(G_B)$ may have more than one “preimage”, i.e., there may be two (or more) distinct vertices $b, b' \in B$ such that both hyper-edges

$\{a_1, a_2, b\}, \{a_1, a_2, b'\}$ are in H . For simplicity we consider the case where the constraints are linear² which implies that for every codeword $w \in C$: $w_b = w_{b'}$. This is a source of some complication for our algorithm, which requires the following definition.

Definition 2. Two vertices v, v' are equivalent if

$$\forall w \in C, \quad w_v = w_{v'}.$$

Clearly this is an equivalence relation. A vertex has multiplicity m if there are exactly m vertices in its equivalence class. For the first read, the reader may assume that all multiplicities are 1.

Denote by V^* the set of vertices whose multiplicity is at most $\beta d^{1/2}$ for $\beta = \alpha/16$ where $\alpha = 3\varepsilon/\delta$.

0. **Init:** Let $\alpha = 3\varepsilon/\delta$ and fix $\beta = \alpha/16$. Let B contain all vertices with multiplicity at least $\beta d^{1/2}$. Let F contain a representative from each of these multiplicity classes. Let B also contain all vertices whose value is the same for all codewords.

1. **Clean:** Repeat the following until B remains fixed:

- (a) Add to B any vertex that occurs in a hyper-edge that has two endpoints in B .
- (b) Add to B all vertices in a connected component of G_B whose size is at least $\beta d^{1/2}$, and add an arbitrary element in that connected component into F .
- (c) Add to B any vertex that has an equivalent vertex in B .

2. **S-step:** Each vertex outside B tosses a biased coin and goes into S with probability $1/d^{1/2}$. Let $B \leftarrow B \cup S$ and set $F \leftarrow F \cup S$.

3. If there are at least two distinct $x, y \in C$ such that $x_B = y_B$ goto step **1**, otherwise halt.

Fig. 1. The Algorithm

The following lemma is easy.

Lemma 1. If the algorithm halted, the code has at most $2^{|F|}$ words.

Proof. This follows since at each step setting the values to vertices in F already fully determines the value of all vertices in B (in any valid codeword). Once the algorithm halts, the values of a codeword on B determines the entire codeword. Thus, there can be at most $2^{|F|}$ codewords.

Let B_t denote the set B at the end of the t -th Clean step (i.e. we refer to the main loop over steps **1a, 1b, 1c**). In order to analyze the expected size of F when the algorithm halts, we analyze the probability that vertices not yet in B will go into B on the next iteration. For a vertex v , this is determined by its neighborhood structure. Let

$$E_v = \{\{u, u'\} \mid u, u' \in V^*, \text{ and } \{u, u', v\} \in E(H)\}$$

² More generally, when the constraints are LOF the set of all such b 's can be partitioned into all those equal to w_b and all those equal to $1 - w_b$.

be a set of edges. Denote by A the vertices v with large $|E_v|$,

$$A = \{v \mid |E_v| \geq \alpha d\}.$$

The following lemma (whose proof appears in the next section) says that if v has sufficiently large E_v then it is likely to enter B in the next round:

Lemma 2. *For $t \geq 2$, if $v \in A$ then*

$$\Pr_S[v \in B_t] \geq \frac{1}{2}.$$

Next, consider a vertex $v \notin A$ that is adjacent, in the graph $G_{B_{t-1}}$, to a vertex $v' \in A$. This means that there is a hyper-edge $h = \{v, v', b\}$ where $b \in B_{t-1}$. If it so happens that $v' \in B_t$ (and the above lemma guarantees that this happens with probability $\geq \frac{1}{2}$), then the hyper-edge h would cause v to go into B_t as well. In fact, one can easily see that if v goes into B_t then all of the vertices in its connected component in $G_{B_{t-1}}$ will go into B_t as well (via step [1a](#)). Let A_t be the set of vertices outside B_t that are in A or are connected by a path in G_{B_t} to some vertex in A . We have proved

Corollary 1. *For $t \geq 2$, let $v \in A_{t-1}$ then*

$$\Pr_S[v \in B_t] \geq \frac{1}{2}.$$

□

Lemma 3. *If the algorithm hasn't halted before the t -th step and $|A_{t-1}| < \frac{\delta}{2}n$ then the algorithm will halt at the end of the t -th step.*

Before proving the two lemmas, let us see how they imply the theorem.

Proof. (of theorem [II](#)) For each $t \geq 2$, Corollary [1](#) implies that for each $v \in A_{t-1}$ half of the S 's put it in B_t . We can ignore the sets S whose size is above $2 \cdot n/d^{1/2}$, as their fraction is negligible. By linearity of expectation, we expect at least half of A_{t-1} to enter B_t . In particular, fix some S_{t-1} to be an S that attains (or exceeds) the expectation. As long as $|A_{t-1}| \geq \delta n/2$ we get

$$|B_t| \geq |B_{t-1}| + |A_{t-1}|/2 \geq |B_{t-1}| + \delta n/4.$$

Since $|B_t| \leq n$ after $\ell \leq 4/\delta$ iterations when the algorithm runs with S_1, \dots, S_ℓ we must have $|A_\ell| < \delta n/2$. This means that the conditions of Lemma [3](#) hold, and the algorithm halts.

How large is the set F ? In each S -step the set F grew by $|S| \leq 2n/d^{1/2}$ (recall we neglected S 's that were larger than that). The total number of vertices that were added to F in S -steps is thus $O(\ell \cdot n/d^{1/2})$.

Other vertices are added into F in the init step and in step [1b](#). In both of these steps one vertex is added to F for every $\beta d^{1/2}$ vertices outside B that are added into B . Since vertices never exit B , the total number of this type of F -vertices is $n/(\beta d^{1/2})$.

Altogether, with non-zero probability, the final set F has size $O(\frac{1}{d^{1/2}}) \cdot n$. Together with Lemma 1 this gives the desired bound on the number of codewords and we are done.

We now prove the two lemmas.

2.1 Proof of Lemma 2

We fix some $v \in A$. If $v \in B_{t-1}$ then we are done since $B_t \supseteq B_{t-1}$. So assume $v \notin B_{t-1}$ and let us analyze the probability of v entering B_t over the random choice of the set S at iteration $t - 1$. This is dictated by the graph structure induced by the edges of E_v . Let us call this graph $G = (U, E_v)$, where U contains only the vertices that touch at least one edge of E_v . We do not know how many vertices participate in U , but we know that $|E_v| \geq \alpha d$.

We begin by observing that all of the neighbors of $u \in U$ must be in the same equivalence class 3. Indeed each of the edges $\{v, u, u_i\}$ is a hyper-edge in H and the value of u_i is determined by the values of v and u . Therefore, the degree in G of any vertex $u \in U$ is at most $\beta d^{1/2}$, since vertices with higher multiplicity are not in V^* and therefore do not participate in edges of E_v .

For each $u \in U$ let I_u be an indicator variable that takes the value 1 iff there is a neighbor of u that goes into S . If this happens then either

- $u \in S$: this means that v has a hyperedge whose two other endpoints are in B_t and will itself go into B_t (in step 1a).
- $u \notin S$: this means that the graph G_{B_t} will have an edge $\{v, u\}$.

If the first case occurs for any $u \in U$ we are done, since v goes into B_t in step 1a. Otherwise, the random variable $\sum_{u \in U} I_u$ counts how many distinct edges $\{v, u\}$ will occur in G_{B_t} . If this number is above $\beta d^{1/2}$ then v will go into B_t (in step 1b) and we will again be done. It is easy to compute the expected value of I . First, observe that

$$\mathbb{E}[I_u] = 1 - (1 - 1/d^{1/2})^{deg(u)}$$

where $deg(u)$ denotes the degree of u in G and since the degree of u is at most $\beta d^{1/2}$, this value is between $deg(u)/2d^{1/2}$ and $deg(u)/d^{1/2}$. By linearity of expectation

$$\mathbb{E}[I] = \sum_u \mathbb{E}[I_u] \geq \sum_u deg(u)/2d^{1/2} = |E_v| d^{-1/2} \geq \alpha d^{1/2}.$$

We will show that I has good probability of attaining a value near the expectation (and in particular at least $\alpha d^{1/2}/2 \geq \beta d^{1/2}$), and this will put v in B_t at step 1b. The variables I_u are not mutually independent, but we will be able to show sufficient concentration by bounding the variance of I , and applying Chebychev's inequality.

³ Or, more generally for LOF constraints, in one of a constant number of equivalence classes.

The random variables I_u and $I_{u'}$ are dependent exactly when u, u' have a common neighbor (the value of I_u depends on whether the neighbors of u go into S). We already know that having a common neighbor implies that u, u' are in the same multiplicity class. Since $U \subset V^*$, this multiplicity class can have size at most $\beta d^{1/2}$. This means that we can partition the vertices in U according to their multiplicity class, such that I_u and $I_{u'}$ are fully independent when u, u' are from distinct multiplicity classes. Let u_1, \dots, u_t be representatives of the multiplicity classes, and let $d_i \leq \beta d^{1/2}$ denote the size of the i th multiplicity class. Also, write $u \sim u'$ if they are from the same multiplicity class.

$$\begin{aligned} \text{Var}[I] &= \mathbb{E}[I^2] - (\mathbb{E}[I])^2 = \mathbb{E} \sum_{u, u'} I_u I_{u'} - \sum_{u, u'} \mathbb{E} I_u \mathbb{E} I_{u'} \\ &= \sum_{u \sim u'} \mathbb{E}[I_u I_{u'}] + \sum_{u \not\sim u'} \mathbb{E} I_u \mathbb{E} I_{u'} - \sum_{u, u'} \mathbb{E} I_u \mathbb{E} I_{u'} \\ &\leq \sum_i \sum_{u \sim u_i} \sum_{u' \sim u_i} \mathbb{E} I_u I_{u'} \\ &\leq \sum_i \sum_{u \sim u_i} \sum_{u' \sim u_i} \mathbb{E} I_u \cdot 1 \\ &\leq \sum_i \sum_{u \sim u_i} \mathbb{E} I_u \cdot d_i \leq \sum_i \sum_{u \sim u_i} \frac{\text{deg}(u)}{d^{1/2}} \cdot \beta d^{1/2} \\ &= \beta \sum_u \text{deg}(u) = 2\beta |E_v| \end{aligned}$$

By Chebychev’s inequality,

$$\Pr[|I - \mathbb{E}[I]| \geq a] \leq \text{Var}[I]/a^2$$

Plugging in $a = \mathbb{E}[I]/2$ we get

$$\Pr \left[|I - \mathbb{E}[I]| \geq \frac{\mathbb{E}[I]}{2} \right] \leq \frac{\text{Var}[I]}{(\mathbb{E}[I]/2)^2} \leq (2\beta |E_v|) \cdot \left(\frac{1}{2} |E_v| d^{-1/2} \right)^{-2} \leq 8\beta d / |E_v| \leq 8\beta / \alpha.$$

and so by choosing $\beta = \alpha/16$ this probability is at most a half. Thus, the probability that $I \geq \mathbb{E}[I]/2 \geq \alpha d^{1/2}/2$ is at least a half. As we said before, whenever $I \geq \beta d^{1/2}$ we are guaranteed that v will enter B_t in the next Clean step [1b](#) and we are done. □

2.2 Proof of Lemma [3](#)

We shall prove that if the algorithm hasn’t halted before the t -th step and $|A_{t-1}| < \frac{\delta}{2}n$ then $|B_t| > (1 - \delta)n$. This immediately implies that the algorithm must halt because after fixing values to more than $1 - \delta$ fraction of the coordinates of a codeword, there is a unique way to complete it.

Recall that A is the set of all vertices v for which $|E_v| \geq \alpha d$. The set B_{t-1} is the set B in the algorithm after the $t - 1$ -th Clean step. The set A_{t-1} is the set

of vertices outside B_{t-1} that are connected by a path in $G_{B_{t-1}}$ to some vertex in A . Finally, denote $G = G_{B_{t-1}}$.

Assume for contradiction that $|B_t| \leq (1 - \delta)n$ and $|A_{t-1}| < \delta n/2$. This means that $Z = V \setminus (A_{t-1} \cup B_t)$ contains more than $\delta n/2$ vertices. Since $Z \cap A_{t-1} = \emptyset$, every vertex $v \in Z$ has $|E_v| < \alpha d$. Our contradiction will come by finding a vertex in Z with large E_v . If the algorithm doesn't yet halt, there must be two distinct codewords $x, y \in C$ that agree on B_t . Let $U_{x \neq y} = \{u \in V \mid x_u \neq y_u\}$. This is a set of size at least δn that is disjoint from B_t . Since $|A_{t-1}| \leq \delta n/2$ there must be at least $\delta n/2$ vertices in $Z \cap U_{x \neq y}$. Suppose $u \in Z \cap U_{x \neq y}$ and suppose u' is adjacent to u in G . First, by definition of Z , $u \in Z$ implies $u' \in Z$. Next, we claim that $u \in U_{x \neq y}$ implies $u' \in U_{x \neq y}$. Otherwise there would be an edge $\{u, u', b\} \in E(H)$ such that $b \in B_t$, and such that $x_u \neq y_u$ but both $x_{u'} = y_{u'}$ and $x_b = y_b$. This means that either x or y must violate this edge, contradicting the fact that all hyper-edges should accept a legal codeword. We conclude that the set $Z \cap U_{x \neq y}$ is a union of connected components of G . Since each connected component has size at most $\beta d^{1/2}$ (otherwise it would have gone into B in a previous Clean step) we can find a set $D \subset Z \cap U_{x \neq y}$ of size s , for

$$\frac{\delta}{3}n \leq \frac{\delta}{2}n - \beta d^{1/2} \leq s \leq \frac{\delta}{2}n,$$

that is a union of connected components, i.e. such that no G -edge crosses the cut between D and $V \setminus D$. Now define the hybrid word

$$w = x_D y_{V \setminus D}$$

that equals x on D and y outside D . We claim that $\text{dist}(w, C) = \text{dist}(w, y) = |D|/n \geq \delta/3$. Otherwise there would be a word $z \in C$ whose distance to w is strictly less than $|D|/n \leq \delta/2$ which, by the triangle inequality, would mean it is less than δn away from y thereby contradicting the minimal distance δn of the code.

Finally, we use the fact that C is an LTC,

$$\text{dist}(w, C) \geq \delta/3 \quad \implies \quad \text{Prob}_{h \sim E(H)}[h \text{ rejects } w] \geq \varepsilon.$$

Clearly to reject w a hyperedge must touch D . Furthermore, such a hyperedge cannot intersect B on 2 vertices because then the third non- B_t vertex also belongs to B_t . It cannot intersect B_t on 1 vertex because this means that either the two other endpoints are both in D , which is impossible since such a hyperedge would reject the legal codeword x as well; or this hyperedge induces an edge in G that crosses the cut between D and $V \setminus D$. Thus, rejecting hyper-edges must not intersect B_t at all.

Altogether we have εdn rejecting hyperedges spanned on $V \setminus B_t$ such that each one intersects D . This means that there must be some vertex $v \in D$ that touches at least $\varepsilon dn / (\delta n/3) = \alpha d$ rejecting hyperedges. Recall that $D \subset Z$ is disjoint from A , which means that $|E_v| < \alpha d$. On the other hand, each rejecting hyperedge touching v must add a distinct edge to E_v . Indeed recall that E_v

contains an edge $\{u, u'\}$ for each hyperedge $\{u, u', v\}$ such that $u, u' \in V^*$ and where V^* is the set of vertices with multiplicity at most $\beta d^{1/2}$. The claim follows since obviously all of the αd rejecting hyperedges are of this form (they do not contain a vertex of high multiplicity as these vertices are in B). \square

3 Exploring Possible Improvements

3.1 Tradeoff between Rate and Density

Any improvement over our bound of $\rho < 1/d^{1/2}$, say to a bound of the form $\rho < 1/d^{0.501}$ would already be strong enough to rule out c^3 -LTCs (with a non-weighted tester) regardless of their density. The reason for this is the following reduction by Oded Goldreich.

Lemma 4. *Suppose for some $q \geq 3$ and some $\epsilon, \delta > 0$ the following were true.*

For any family $\{C_n\}$ of q -query LTCs with rate $\leq \rho$ distance $\delta > 0$ and such that each C_n has a tester with density at least d , then $\rho \leq 1/d^{\frac{1}{q-1} + \epsilon}$.

Then, there is no family of q -query LTCs with constant rate, distance $\delta > 0$, and any density, such that the tester is non-weighted.

Proof. Let $\beta = \frac{1}{q-1} + \epsilon$, and let $t \in \mathbb{N}$. Let $\{C_i\}$ be an infinite family of q -query LTCs with density $d = O(1)$, relative rate $\rho = \Omega(1)$, and distance $\delta > 0$. Then there is another infinite family $\{\tilde{C}_i\}$ of q -query LTCs with density $d \cdot t^{q-1}$, same distance, and relative rate ρ/t . \tilde{C}_i is constructed from C_i by duplicating each coordinate t times and replacing each test hyper-edge by t^q hyperedges. Clearly the density and the rate are as claimed. The testability can also be shown. Plugging in the values $\tilde{\rho} = \rho/t$ and $\tilde{d} = dt^{q-1}$ into the assumption we get

$$\rho/t = \tilde{\rho} \leq 1/\tilde{d}^\beta = 1/(dt^{q-1})^\beta$$

In other words $\rho d^\beta \leq t^{1-(q-1)\beta}$. Since t is unbounded this can hold only if the exponent of t is positive, i.e., $\beta \leq 1/(q-1)$, a contradiction.

3.2 For $q > 3$ Density Must Be High

Lemma 5. *Let C be a q -query LTC with rate ρ , and density d . Then for every $q' > 0$ there is a $(q + q')$ -query LTC C' with density $d \cdot \binom{n}{q'}$ such that C' has rate $\rho/2$, distance $\delta/2$.*

Corollary 2. *If there is a 3-query LTC with constant rate and density, then there are LTCs with $q > 3$ -queries, constant rate, and density $\Omega(n^{q-3})$.*

The corollary shows that our upper bounds from Theorem 2 are roughly correct in their dependence on n , but there is still a gap in the exponent.

Proof. (of lemma 5) Imagine adding another n coordinates to the code C such that they are always all zero. Clearly the distance and the rate are as claimed. For the testability, we replace each q -hyper-edge e of the hypergraph of C with $\binom{n}{q'}$ new hyperedges that consist of the vertices of e plus any q' of the new vertices. The test associated with this hyperedge will accept iff the old test would have accepted, and the new vertices are assigned 0. It is easy to see that the new hypergraph has average degree $d \cdot \binom{n}{q'}$. Testability can be shown as well.

3.3 Allowing Weighted Hypergraph-Tests

In this section we claim that when considering hypergraph tests *with weights*, the density should not be defined as the ratio between the number of edges and the number of vertices. Perhaps a definition that takes the min-entropy of the graph into consideration would be better-suited, but this seems elusive, and we leave it for future work.

We next show that if one defines the density like before (ignoring the weights) then every LTC can be modified into one that has a maximally-dense tester. This implies that bounding the rate as a function of the density is the same as simply bounding the rate.

Lemma 6. *Let C be a q -query LTC with $q \geq 3$, rate ρ , distance δ , and any density. Then there is another q -query LTC C' with a weighted-tester of maximal density $\Omega(n^{q-1})$ such that C' has rate $\rho/2$, distance $\delta/2$.*

Corollary 3. *Let $f : \mathbb{N} \rightarrow \mathbb{N}$ be any non-decreasing non-constant function. Any bound of the form $\rho \leq 1/f(d)$ for weighted testers implies $\rho \leq 1/f(n^{q-1})$, and in particular $\rho \rightarrow 0$. \square*

Proof. (of lemma 6) One can artificially increase the density of an LTC tester hypergraph H by adding n new coordinates to the code that are always zero, and adding all possible q -hyperedges over those coordinates (checking that the values are all-zero). All of the new hyper-edges will be normalized to have total weight one half, and the old hyperedges will also be re-normalized to have total weight one half. Clearly the rate and distance have been halved, and the testability is maintained (with a different rejection ratio). However, the number of hyperedges has increased to n^q so the density is as claimed.

Acknowledgement. We would like to thank Oded Goldreich for very interesting discussions, and for pointing out the reduction in Section 3.1.

References

- [BS05] Ben-Sasson, E., Sudan, M.: Simple PCPs with poly-log rate and query complexity. In: Proc. 37th ACM Symp. on Theory of Computing, pp. 266–275 (2005)

- [BSGK⁺10] Ben-Sasson, E., Guruswami, V., Kaufman, T., Sudan, M., Viderman, M.: Locally testable codes require redundant testers. *SIAM J. Comput.* 39(7), 3230–3247 (2010)
- [BSGS03] Ben-Sasson, E., Goldreich, O., Sudan, M.: Bounds on 2-query codeword testing. In: Arora, S., Jansen, K., Rolim, J.D.P., Sahai, A. (eds.) *RANDOM 2003 and APPROX 2003*. LNCS, vol. 2764, pp. 216–227. Springer, Heidelberg (2003)
- [Din07] Dinur, I.: The PCP theorem by gap amplification. *Journal of the ACM* 54(3) (2007)
- [GS06] Goldreich, O., Sudan, M.: Locally testable codes and PCPs of almost-linear length. *J. ACM* 53(4), 558–655 (2006)
- [Mei09] Meir, O.: Combinatorial construction of locally testable codes. *SIAM J. Comput.* 39(2), 491–544 (2009)

Efficient Probabilistically Checkable Debates

Andrew Drucker*

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
Cambridge, MA 02139
adrucker@mit.edu

<http://www.springer.com/lncs>

Abstract. Probabilistically checkable debate systems (PCDSs) are debates between two competing provers, in which a polynomial-time verifier inspects a constant number of bits of the debate. It was shown by Condon, Feigenbaum, Lund, and Shor that every language in PSPACE has a PCDS in which the debate length is polynomially bounded. Using this result, they showed that the approximation versions of some natural PSPACE-complete problems are also PSPACE-complete.

We give an improved construction of these debates: for any language L that has an ordinary debate system definable by uniform circuits of size $s = s(n)$, we give a PCDS for L whose debate is of total bitlength $\tilde{O}(s)$, with a verifier that uses only $\log_2 s + \log_2(\text{polylog}(s))$ bits of randomness. This yields a much tighter connection between the time complexity of natural PSPACE-complete problems and the time complexity of their approximation versions.

Our key ingredient is a novel application of error-resilient communication protocols, as developed by Schulman; we use the more recent protocol of Braverman and Rao. We show that by requiring ordinary debates to be encoded in an error-resilient fashion, we can endow them with a useful “stability” property. Stable debates can then be transformed into PCDSs, by applying efficient PCPPs (as given by Dinur). Our main technical challenge in building stable debates is to enforce error-resilient encoding by the debaters. To achieve this, we show that there is a constant-round debate system, with a very efficient verifier, to debate whether a communication transcript follows the Braverman-Rao protocol.

Keywords: PCPPs, PCPs, Probabilistically Checkable Debates.

1 Introduction

1.1 Debate Systems

For many years, *debate systems* have played an important role in the study of complexity classes (e.g., in [7], [18], [21], [2], [9]). A debate system is an interaction between two competing, computationally-unbounded *debaters*, supervised

* Supported by a DARPA YFA grant.

by a computationally-bounded *verifier*. Debate systems are often equivalently described in terms of *alternation* or *alternating nondeterminism* [7].

In a debate system for a language L , an input $x \in \{0, 1\}^n$ is given; the first debater (Player 1) tries to convince the verifier that $x \in L$, while the second debater (Player 0) tries to convince the verifier that $x \notin L$. To this end, the debaters supply a sequence of strings $y = (y^1, \dots, y^{k(n)})$. Here, each y^i is of a prespecified length depending only on n ; y^i is supplied by a prespecified debater $type(i) \in \{0, 1\}$, and is allowed to depend on y^1, \dots, y^{i-1} . (We assume that $k(n)$, $type(i)$, and the lengths $|y_i|$ are all computable in time $\text{poly}(n)$.)

Finally, the verifier applies a deterministic predicate $V(x, y)$ to determine an output $b \in \{0, 1\}$. We say V defines a *debate system for L* if Player 1 can force $b = 1$ exactly when $x \in L$. If moreover $|y| \leq \text{poly}(n)$ and V is a polynomial-time algorithm, we say that V defines a *polynomial-time debate system for L* .

An important parameter is $k = k(n)$, the number of turns in the debate. If $k \geq 1$ is a constant, L has a k -round polynomial-time debate system if and only if L lies in the k^{th} level of the Polynomial Hierarchy (i.e., $\Sigma_k^P \cup \Pi_k^P$), as was essentially shown by Stockmeyer [24] and Wrathall [26]. Chandra and Stockmeyer [8] (see also [7]) showed that when k is allowed to grow polynomially in the input length, polynomial-time debate systems characterize PSPACE:

Theorem 1. [8], [7] *A language L has a polynomial-time debate system if and only if $L \in \text{PSPACE}$.*

Later Shamir [21], building on [18], showed that every language $L \in \text{PSPACE}$ has an *interactive proof*—a polynomial-time debate system in which Player 0 plays completely random strings. If $x \in L$, then some Player 1 strategy causes the verifier to accept with probability 1, while if $x \notin L$, any Player 1 strategy causes the verifier to accept with probability at most $1/2$.

These results, as well as debate characterizations of other complexity classes, have been instrumental in determining the complexity of many natural computational problems—particularly 2-player games, which are readily expressed as debates. See [14] for a fuller discussion of the importance of debate systems in complexity theory.

1.2 Probabilistically Checkable Debates

One of the most significant and surprising discoveries about debate systems is that they retain essentially all of their computational power under severe restrictions on the verifier V .

This discovery has its roots in the study of *probabilistically checkable proofs (PCPs)* for NP languages. We say that a randomized polynomial-time algorithm $V(x, y)$ is an $[r(n), q(n)]$ -*restricted probabilistically checkable proof system* for the language L (with input $x \in \{0, 1\}^n$ and proof string y of length $\text{poly}(n)$), if:

1. For all $x \in L$ there is a y with $\Pr[V(x, y) \text{ accepts}] = 1$;
2. For all $x \notin L$ and all y , $\Pr[V(x, y) \text{ accepts}] < 1 - \Omega(1)$;

3. V uses $r(n)$ bits of randomness and nonadaptively queries at most $q(n)$ bits of y .

The famous PCP Theorem of [1] states that we can provide an $[O(\log n), O(1)]$ -restricted probabilistically checkable proof system for any $L \in \text{NP}$. That is, there exists a special proof format that allows one to efficiently verify membership claims for L , while only looking at a constant number of bits of the proof!

An NP verifier for a language $L \in \text{NP}$ can be viewed in the debate framework, as a debate system consisting of a single turn by Player 1. So, single-turn debates can be made probabilistically checkable with $O(1)$ queries, and it is natural to wonder whether the same can be done for more general classes of debates. To formalize this, let $V(x, y)$ be a polynomial-time verifier, where as before $y = (y^1, \dots, y^k)$ are supplied by the competing debaters (we assume $|y| \leq \text{poly}(n)$). Now, however, V uses randomness. We call V an $[r(n), q(n)]$ -restricted probabilistically checkable debate system (PCDS) for the language L if:

1. For all $x \in L$, there is a Player 1 strategy that forces $\Pr[V(x, y^1, \dots, y^k) \text{ accepts}] = 1$ (note, we insist on *perfect completeness*);
2. For all $x \notin L$, there is a Player 0 strategy that forces $\Pr[V(x, y^1, \dots, y^k) \text{ accepts}] < 1 - \Omega(1)$;
3. V uses $r(n)$ bits of randomness and nonadaptively queries at most $q(n)$ bits of y .

How powerful are PCDSs when we allow k to grow polynomially in n , but require $q(n) = O(1)$? Intuitively, this restriction seems quite severe, since only a constant number of the debate strings y^1, \dots, y^k will receive any queries at all. However, this intuition is deceptive: in [9], Condon, Feigenbaum, Lund, and Shor proved that PCDSs are essentially as strong as arbitrary polynomial-time debate systems:

Theorem 2. [9] *Every $L \in \text{PSPACE}$ has an $[O(\log n), O(1)]$ -restricted PCDS.*

Their proof used the PCP Theorem as a building block, along with several interesting new ideas. This result was complemented by several other works that studied various classes of debates; in each case it was shown that restricting the verifier to make $O(1)$ queries to the debate string does not reduce the power of the debates in question. Ko and Lin [17] showed that for any $k \geq 1$, if L is in the k^{th} level of the Polynomial Hierarchy, then there is a k -round PCDS either for L or for \bar{L} . Condon, Feigenbaum, Lund, and Shor [10], in a follow-up to their original work, showed that the interactive proofs in Shamir's result can be made probabilistically checkable; in their PCDSs, the verifier looks at only a constant number of the random bits written by Player 0 as well as a constant number of Player 1's bits. A corresponding result for AM protocols was shown by Drucker [13].

1.3 The Inapproximability Connection

In addition to their inherent interest, probabilistically checkable debates also have a close connection to the complexity of approximation problems. The PCP Theorem (along with its many subsequent refinements) was the key to proving most of the known NP-hardness-of-approximation results for optimization problems whose decision versions lie in NP. Similarly, Theorem 2 implies that a number of reasonably natural computational problems lying in PSPACE are in fact PSPACE-hard to approximate, as was shown in 9. As one simple example, suppose we are given a 3-CNF formula $\psi(x_1, \dots, x_n)$, and we consider the game in which two players take turns in assigning values to the variables, with x_i assigned on the i^{th} round. Player 1 wants to maximize the fraction of satisfied clauses, while Player 0 wants to minimize this fraction. Let $\text{Val}(\mathcal{G}_\psi) \in [0, 1]$ denote the value of this game to Player 1. Using Theorem 1, one can show that it is PSPACE-complete to decide whether $\text{Val}(\mathcal{G}_\psi) = 1$. However, from Theorem 2, the authors of 9 showed that for some constant $\varepsilon > 0$, it is PSPACE-hard even to distinguish formulas with $\text{Val}(\mathcal{G}_\psi) = 1$ from formulas with $\text{Val}(\mathcal{G}_\psi) < 1 - \varepsilon$ 1.

While this is a remarkable result, it is not completely satisfactory because the reduction involved, though polynomial-time, causes a large polynomial blowup of the 3-CNF instance sizes 2. This blowup leads to somewhat weak conditional hardness results. Assume that any algorithm to correctly decide whether $\text{Val}(\mathcal{G}_\psi) = 1$ for a 3-CNF ψ of description length n , requires time $\geq T(n)$ infinitely often (“i.o.”), for some time bound $T(n) = n^{\omega(1)}$. Then, Theorem 2 implies that for small enough $\varepsilon > 0$, any algorithm achieving an ε -approximation to $\text{Val}(\mathcal{G}_\psi)$ requires runtime $T(n^a)$ i.o., for some explicit absolute constant $a < 1$.

We note that the reduction in the original PCP Theorem also incurred a similar blowup in parameters. However, in recent work Dinur [11, Thm. 8.1] gave a $[\log_2 n + \log_2(\text{polylog}(n)), O(1)]$ -probabilistically checkable proof system for length- n SAT instances. This yields much tighter conditional hardness statements for SAT: from an i.o.-lower bound $T(n) = n^{\omega(1)}$ on the runtime of algorithms for SAT, we get an i.o.-lower bound of $T(n/\log^c(n))$ on the runtime of algorithms to ε -approximate the maximum fraction of satisfiable clauses in a 3-CNF of description length n (for explicit constants $c, \varepsilon > 0$).

1.4 Our Result

Our main result is the following quantitative strengthening of Theorem 2, which shows that polynomial-time debates can be made probabilistically checkable in a randomness-efficient way, with a debate string whose size is nearly-linear in the circuit complexity of the original verifier:

¹ The technique is the same as that used to derive inapproximability results for Maximum Satisfiability from PCP constructions.

² 9 does not provide an explicit polynomial bound. The bound arising in their work can be brought down somewhat by substituting state-of-the-art PCPs from [11] into their applications of PCPs. However, it appears that following their basic approach leads to at least a cubic blowup.

Theorem 3 (Main). *Suppose L has a polynomial-time debate system with a verifier implementable by polynomial-time-constructible Boolean circuits of size $s = s(n) \leq \text{poly}(n)$. Then L has a $[\log_2 s + \log_2(\text{polylog}(s)), O(1)]$ -restricted PCDS, with a debate string of total bitlength $\tilde{O}(s)$.*

Many natural PSPACE-complete problems, like QBF-SAT (the set of true quantified Boolean formulas, under any standard encoding), have ordinary debate systems with circuits of size $s(n) = \tilde{O}(n)$, and for such problems Theorem 3 yields a PCDS with debate string bitlength $\tilde{O}(n)$ and randomness $\log_2(n) + \log_2(\text{polylog}(n))$. Then using Theorem 3, an i.o.-lower bound $T(n) = n^{\omega(1)}$ for determining whether $\text{Val}(\mathcal{G}_\psi) = 1$ for 3-CNFs ψ allows us to infer an i.o.-lower bound of $T(n/\log^c(n))$ on the runtime of algorithms to ε -approximate $\text{Val}(\mathcal{G}_\psi)$, for explicit constants $c, \varepsilon > 0$.

We also mention that any language solvable in space $S(n) \leq \text{poly}(n)$ has a polynomial-time debate system definable by uniform circuits of size $s(n) = \tilde{O}(S(n)^2)$ [23, 7]. Thus, for such languages we get a PCDS with debate bitlength $\tilde{O}(S(n)^2)$.

The PCDS construction in Theorem 3 has close to the best randomness-efficiency we can hope to achieve (for general L and s), barring a huge algorithmic breakthrough: if the randomness complexity could be brought down to $(1 - \Omega(1)) \log_2(s)$, we could use the resulting PCDS for QBF-SAT to solve length- n QBF-SAT instances in time $2^{n^{1-\Omega(1)}}$. Now, it is also natural to wonder whether the debate string in the PCDS of Theorem 3 could always be made to have bitlength bounded in terms of $\ell = \ell(n)$, the bitlength of the ordinary debate-string for L (which may be much smaller than s). Any bound of form $\text{poly}(\ell)$ would be very interesting. However, this seems unlikely. In the theory of probabilistically checkable proofs, the corresponding question is whether satisfiability of q -variable SAT instances can be proved with a PCP of proof-length $\text{poly}(q)$; Fortnow and Santhanam showed that this cannot be done unless $\text{NP} \subseteq \text{coNP/poly}$ [15].

The proof of Theorem 3 is fairly long and involved. In this extended abstract we only describe the main ideas at a high level; details can be found in the full version online [4].

1.5 Our Techniques

To prove Theorem 3, we give a new method for converting standard debate systems into probabilistically checkable ones. The method has two steps. In the first (and more novel) step, we transform a standard debate into one that has a useful “stability” property; in the second step, we transform a stable debate into a probabilistically checkable debate.

Stable Debates via Error-resilient Communication. We say a debate system $V(x, \cdot)$ for a language L is *stable* if for any $x \notin L$, Player 0 can not only

³ Actually, for QBF-SAT it is not hard to achieve $s(n) = O(n)$; see [25, p. 1].

⁴ <http://eccc.hpi-web.de/report/2011/073/>

force $V(x, y) = 0$, but can even force the debate string y to be $\Omega(1)$ -far in relative Hamming distance from any y' for which $V(x, y') = 1$. (Thus, our stability notion is asymmetric with respect to the players.) The notion of stability was used before, implicitly or explicitly, in several ways in [9], [10], [13]. However, in the previous works building many-round PCDSs [9], [10], debates are endowed with a stability-like property⁵ in a fairly inefficient way: roughly speaking, on each turn of the debate, the current player is asked to give a description of all previous moves along with their current move. This contributes a quadratic blowup in the overall debate-string length (in addition to blowups at other steps in the transformation).

In our transformation yielding a stable debate, we manage to avoid using such redundancy. We do so by drawing a new connection to *interactive coding*—the theory of error-resilient two-way communication. Interactive coding was pioneered by Schulman [20], who showed that any two-party communication protocol can be converted into one that succeeds even in the presence of a noticeable amount of adversarial noise (an adversary that may adaptively corrupt a $1/240$ fraction of the bits sent between the two parties). Moreover, this conversion increases the total communication by only a constant factor. Schulman’s powerful result seems not to be obtainable from standard tools for resilient one-way communication (i.e., error-correcting codes).

Recently, Braverman and Rao [6] gave a new encoding method to achieve the same goal. Their encoding corrects from a much larger fraction of adversarially corrupted symbols—nearly $1/8$ if bits are transmitted, or nearly $1/4$ if a larger but constant-sized message alphabet is used. More importantly for us, their encoding is somewhat simpler and easier to work with. (We do not know whether the protocol of [20] could also be used to prove our result.)

In our application of interactive coding, we begin with an ordinary debate system for a language L , defined by a verifier V implemented by uniform circuits of size $s(n)$. We then transform V into a second verifier V^{stab} , in which the two debaters are “forced” to encode their debate using the Braverman-Rao encoding. We show that the error-resilience property of the encoding can be used to ensure the stability property of V^{stab} .

But how can we force the debaters to follow the desired encoding? To do this, we construct an auxiliary “*encoding-checker*” debate system, that allows players to debate whether a communication transcript corresponds to a faithful, noise-free execution of the Braverman-Rao protocol.⁶ The encoding-checker debate we

⁵ These papers’ first step is to transform an ordinary debate system into one in which a probabilistic verifier inspects only a constant number of the individual player *moves* (y^i), which may be of polynomial length. Such debates play a role somewhat analogous to the role played by stable debates in our work; the analogy is loose, however, and stable debates turn out to be more easily and efficiently turned into PCDSs.

⁶ More precisely, they can debate whether a *particular* player is following the Braverman-Rao protocol. Checking one player’s behavior turns out to be sufficient; the other player can be indirectly “incentivized” to follow the protocol. This is important for achieving perfect completeness in our PCDSs.

construct has two important properties. First, it lasts for only $O(1)$ turns. This is important because an $O(1)$ -turn debate can fairly easily be made *stable*, by asking for the moves to be encoded in an error-correcting code—we use efficiently decodable codes given by Spielman [22]. With this stable encoding-checker debate in hand, we can make the entire debate stable. (Conceptually, this is the right picture; technically, we make the entire debate stable in one step, rather than first making the auxiliary debate stable.)

The second important property of our encoding-checker debate is that it has a very efficient verifier—one that is definable by a Boolean circuit of size $O(\ell)$, where ℓ is the bitlength of the communication transcript being checked. As a result, our stable verifier V^{stab} can be implemented by a circuit of size $\tilde{O}(s(n))$ for length- n inputs. This near-linear efficiency is important in the second step of our transformation, in which we make the debate probabilistically checkable.

Achieving these two strong properties in our encoding-checker debate is our main technical challenge. We will return to the issue of how this challenge can be met.

From Stable to Probabilistically Checkable Debates. In our second transformation step, we extend our stable debate $y = (y^1, \dots, y^k)$ by a single, final turn, in which Player 1 gives a “proof” string z purporting to show that $V_x^{stab}(y) := V^{stab}(x, y) = 1$. We then define a verifier V^* that, given x , probabilistically checks $O(1)$ bits of y and z . For this proof/verification task, we use a powerful variant of PCPs known as *probabilistically checkable proofs of proximity (PCPPs)* [3], [12], applied to the circuit for the stable verifier V_x^{stab} . We mention that PCPPs were put to a similar use in [13], and closely related techniques were used in [9], [10].

If $x \in L$, then Player 1 is able to win the stable debate (y^1, \dots, y^k) , so that $V_x^{stab}(y) = 1$, and then some proof z causes V^* to accept with certainty. On the other hand, if $x \notin L$, then by stability, Player 0 can guarantee that y is not even *close* to the set of debate strings for which $V_x^{stab} = 1$. This is precisely the condition in the definition of PCPPs that guarantees that V^* will reject with noticeable probability for any setting to z . Thus the probabilistic verifier V^* defines our desired PCDS for L .

How efficient is this verifier? The length of the final proof-string z and the randomness complexity of V^* are determined by two factors: the efficiency of the PCPP construction we use, and the size of the circuit for V_x^{stab} to which we apply the PCPP. We are fortunate in both respects. A very efficient construction of PCPPs is available, due to Dinur [11] (building on [4]); also, our efficient construction of stable debates ensures that the circuit for V_x^{stab} is of size $\tilde{O}(s(n))$. This yields a verifier and debate-string with the properties claimed in Theorem 3.

Building Encoding-Checker Debates. Recall that our approach requires us to build “encoding-checker” debates, to check that a given party’s (Alice’s)

behavior on a communication transcript corresponds to a correct execution of the Braverman-Rao protocol [6] for error-resilient communication.

A first complication is that the Braverman-Rao protocol is not even known to have an implementation in *polynomial* time, let alone *nearly-linear* time. Like Schulman’s earlier protocol [20], the protocol crucially relies on a special type of codes called *tree codes*, defined by Schulman. Tree codes have a “distance” parameter, that is loosely analogous to the minimum-distance parameter for error-correcting codes. No explicit construction is known of tree codes with distance large enough to run the protocols of [6], [20] (see [5] for a recent subexponential-time construction). However, in [20] an elegant probabilistic construction of tree codes was given. Importantly for us, this construction is very *randomness-efficient*, so that good tree codes exist with succinct representations. This is enough for our purposes: in our debate system, a computationally unbounded debater may succinctly propose a tree code, and establish *through debate* that it has the needed distance property.

The protocols of [20], [6] require the communicating parties to decode corrupted tree-code-encoded messages. In the model in which communication is corrupted adversarially (the relevant model for us), it is not known whether this decoding can be performed in polynomial time. However, we are again able to use the power of computationally unbounded debaters, this time to debate the correct values of tree-code encodings and decodings.

We mention in passing that tree codes are not known to be *necessary* for interactive coding. Indeed, in very recent papers by Moitra [19] and, independently, Gelles and Sahai [16], it was shown that the tree-code definition can be *relaxed* to a weaker but still-useful kind of object that is easier to construct, yielding interactive coding schemes that are computationally efficient in the *random channel-noise* model. The papers use different relaxations and achieve similar but incomparable results. Achieving efficiency in the adversarial-noise model remains an important open question.

Now in the Braverman-Rao protocol, correct behavior for each player is defined relative to some *input* to the communication task. In our application, the intended input is a player-strategy in the original debate V for L (Alice is allowed to choose her input/strategy). Such a strategy is an object of exponential size, and cannot even be written down in the encoding-checker debate. Fortunately, any particular execution of the Braverman-Rao protocol only depends on a much smaller portion of the strategy. This crucially helps us, although naively encoding the relevant portion is not succinct enough for our purposes.

The player Alice, while executing the Braverman-Rao protocol, maintains data, and this data needs to be represented in the encoding-checker debate. We are fortunate that the data takes the simple form of a sequence (a_1, a_2, \dots) of symbols over a constant-size alphabet, with a_i being defined on the i^{th} round of communication and never modified. This allows a debater to succinctly present a global description of Alice’s execution. However, a_i is defined in a complex way from previous values and from the messages received from Bob. To understand how to efficiently debate the proper settings to a_i , we make a detailed study of

a method used in the Braverman-Rao protocol to succinctly describe subsets of edges in a complete binary tree. Our encoding-checker debate system is built up from a sequence of simpler debates used to reason about this description method and its use in the protocol.

2 Questions for Future Work

1. Can our probabilistically checkable debate systems be made even more efficient? Currently, we make use of a family of error-correcting codes due to Spielman [22], which are decodable from a constant fraction of error by circuits of size $O(n \log n)$. Developing good error-correcting codes decodable by *linear*-sized circuits would shave a log factor from the length of our debate-strings. If we could do this, then our PCDSs could be made to essentially match the efficiency of the best PCPPs.
2. In our work, we do not attempt to minimize the increase in the number of *turns* in our debate transformation. Our approach gives no improvement over [9] in this respect: both approaches increase the number of turns by a constant factor, *if* the starting debate has strictly-alternating turns, each consisting of a single bit. If not, the number of turns may increase by a much larger amount.

To our knowledge, there also has been no in-depth study of the number of rounds required for error-resilient communication, when the communication protocol to be simulated lasts a bounded number of rounds (with several bits transmitted per round). Can we make communication error-resilient, while increasing each of the number of rounds and the total communication by only a constant (or at least slowly-growing) factor? The challenging case seems to be when rounds are of variable bitlength. Understanding this question would clarify the situation for PCDSs, and would be of interest in its own right.

3. Our PCDSs for L are of bitlength nearly-linear in $s(n)$, the *circuit size* of the verifier for an ordinary debate system for L . We left open whether the PCDS bitlength could be polynomial in the *bitlength* of the original debate. Can we derive unlikely consequences of this? To explore this, one might try to work by analogy with Fortnow and Santhanam's results on infeasibility of succinct PCPs for SAT [15] (see Section 1.4).
4. As mentioned earlier, the same authors who first built PCDSs in [9] also showed in [10] that *interactive proofs* (i.e., debates between a maximizing Player 1 and a completely random Player 0) can also be made probabilistically checkable. It would be very interesting to know whether this reduction can be carried out with efficiency comparable to our reduction for ordinary debate systems in this paper. This would yield improved conditional hardness statements for the complexity of approximating stochastic sequential optimization problems on CSPs. The difficulty in applying our methods is that we appear to have no effective way to make the random player "follow" the Braverman-Rao protocol.

References

1. Arora, S., Lund, C., Motwani, R., Sudan, M., Szegedy, M.: Proof verification and the hardness of approximation problems. *J. ACM* 45(3), 501–555 (1998)
2. Babai, L., Fortnow, L., Lund, C.: Non-deterministic exponential time has two-prover interactive protocols. In: 31st IEEE FOCS, pp. 16–25 (1990)
3. Ben-Sasson, E., Goldreich, O., Harsha, P., Sudan, M., Vadhan, S.P.: Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM J. Comput.* 36(4), 889–974 (2006)
4. Ben-Sasson, E., Sudan, M.: Robust locally testable codes and products of codes. *Random Struct. Algorithms* 28(4), 387–402 (2006)
5. Braverman, M.: Towards deterministic tree code constructions. *Electronic Colloquium on Computational Complexity (ECCC)* TR11-064 (2011)
6. Braverman, M., Rao, A.: Towards coding for maximum errors in interactive communication. In: 43rd ACM STOC, pp. 159–166 (1990)
7. Chandra, A.K., Kozen, D., Stockmeyer, L.J.: Alternation. *J. ACM* 28(1), 114–133 (1981)
8. Chandra, A.K., Stockmeyer, L.J.: Alternation. In: 17th IEEE FOCS, pp. 98–108 (1976)
9. Condon, A., Feigenbaum, J., Lund, C., Shor, P.W.: Probabilistically checkable debate systems and nonapproximability of PSPACE-hard functions. *Chicago J. Theor. Comput. Sci.* (1995)
10. Condon, A., Feigenbaum, J., Lund, C., Shor, P.W.: Random debaters and the hardness of approximating stochastic functions. *SIAM J. Comput.* 26(2), 369–400 (1997)
11. Dinur, I.: The PCP theorem by gap amplification. *J. ACM* 54(3), 12 (2007)
12. Dinur, I., Reingold, O.: Assignment testers: Towards a combinatorial proof of the PCP theorem. *SIAM J. Comput.* 36(4), 975–1024 (2006)
13. Drucker, A.: A PCP characterization of AM. *Electronic Colloquium on Computational Complexity (ECCC)* TR10-019 (2010); To appear in ICALP 2011
14. Fortnow, L.: Beyond NP: The work and legacy of Larry Stockmeyer. In: 37th ACM STOC, pp. 120–127 (2005)
15. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.* 77(1), 91–106 (2011)
16. Gelles, R., Sahai, A.: Potent tree codes and their applications: Coding for interactive communication, revisited. *ArXiv e-prints* (April 2011)
17. Ko, K.-I., Lin, C.-L.: Non-approximability in the polynomial-time hierarchy. TR 94-2, Dept. of Computer Science. SUNY at Stony Brook (1994)
18. Lund, C., Fortnow, L., Karloff, H.J., Nisan, N.: Algebraic methods for interactive proof systems. In: 31st IEEE FOCS, pp. 2–10 (1990)
19. Moitra, A.: Efficiently coding for interactive communication. *Electronic Colloquium on Computational Complexity (ECCC)* TR11-042 (2011)
20. Schulman, L.J.: Coding for interactive communication. *IEEE Trans. Inf. Theory* 42(6), 1745–1756 (1996)
21. Shamir, A.: $IP = PSPACE$. *J. ACM* 39(4), 869–877 (1992)
22. Spielman, D.A.: Linear-time encodable and decodable error-correcting codes. *IEEE Trans. Inf. Theory* 42(6), 1723–1731 (1996)

23. Stockmeyer, L.J., Meyer, A.R.: Word problems requiring exponential time (preliminary report). In: 5th ACM STOC, pp. 1–9 (1973)
24. Stockmeyer, L.J.: The polynomial-time hierarchy. *Theor. Comput. Sci.* 3(1), 1–22 (1976)
25. Williams, R.: Non-linear time lower bound for (succinct) quantified Boolean formulas. *Electronic Colloquium on Computational Complexity (ECCC) TR08-076* (2008)
26. Wrathall, C.: Complete sets and the polynomial-time hierarchy. *Theor. Comput. Sci.* 3(1), 23–33 (1976)

An Efficient Partitioning Oracle for Bounded-Treewidth Graphs

Alan Edelman¹, Avinatan Hassidim², Huy N. Nguyen¹,
and Krzysztof Onak^{3,*}

¹ Massachusetts Institute of Technology

{edelman,huy2n}@mit.edu

² Google

avinatanh@gmail.com

³ Carnegie Mellon University

konak@cs.cmu.edu

Abstract. Partitioning oracles were introduced by Hassidim *et al.* (FOCS 2009) as a generic tool for constant-time algorithms. For any $\varepsilon > 0$, a partitioning oracle provides query access to a fixed partition of the input bounded-degree minor-free graph, in which every component has size $\text{poly}(1/\varepsilon)$, and the number of edges removed is at most εn , where n is the number of vertices in the graph.

However, the oracle of Hassidim *et al.* makes an exponential number of queries to the input graph to answer every query about the partition. In this paper, we construct an efficient partitioning oracle for graphs with constant treewidth. The oracle makes only $O(\text{poly}(1/\varepsilon))$ queries to the input graph to answer each query about the partition.

Examples of bounded-treewidth graph classes include k -outerplanar graphs for fixed k , series-parallel graphs, cactus graphs, and pseudo-forests. Our oracle yields $\text{poly}(1/\varepsilon)$ -time property testing algorithms for membership in these classes of graphs. Another application of the oracle is a $\text{poly}(1/\varepsilon)$ -time algorithm that approximates the maximum matching size, the minimum vertex cover size, and the minimum dominating set size up to an additive εn in graphs with bounded treewidth. Finally, the oracle can be used to test in $\text{poly}(1/\varepsilon)$ time whether the input bounded-treewidth graph is k -colorable or perfect.

1 Introduction

Many NP-complete graph problems can be easily solved on graphs with bounded treewidth. For example, approximating vertex cover up to a multiplicative factor better than 1.36 is known to be NP hard [1]. In contrast, for graphs with bounded treewidth, one can in fact find an optimal vertex cover in time that is linear in the size of the graph, but depends exponentially on treewidth [2]. In this paper, we investigate yet another scenario in which bounded-treewidth graphs turn out to

* Supported in part by a Simons Postdoctoral Fellowship and NSF grants 0732334 and 0728645.

be much easier to deal with and allow for more efficient algorithms than general graphs.

Bounded Treewidth. The *tree decomposition* and *treewidth* were introduced by Robertson and Seymour [3, 4], and later found many applications in the design of algorithms and in machine learning (a nice, though outdated survey is [5]; see also [6, 7] for more recent applications). A *tree decomposition* of a graph $G = (V, E)$ is a pair $(\mathcal{X}, \mathcal{T})$, where $\mathcal{X} = (X_1, X_2, \dots, X_m)$ is a family of subsets of V , and \mathcal{T} is a tree (or forest) whose nodes are the subsets X_i , satisfying the following properties:

1. Every $v \in V$ belongs to at least one X_i , i.e., $\bigcup_{i=1}^m X_i = V$.
2. For every $(u, v) \in E$, there is an X_i such that both u and v belong to X_i .
3. For every vertex $v \in V$, the set of nodes in \mathcal{T} associated with v forms a connected subset of \mathcal{T} .

The *width of a tree decomposition* equals $\max_i |X_i| - 1$. The *treewidth* of G is defined as the minimum such width over all tree decompositions of G .

Graph families with bounded treewidth include k -outerplanar graphs, series-parallel graphs, cactus graphs, and pseudoforests.

The Bounded-degree Model. Before discussing our results, we describe the bounded-degree model introduced by Goldreich and Ron [8] and used in this paper. The degree of every vertex in the input graph is bounded by a constant d . An algorithm can make two kinds of queries to access the input graph $G = (V, E)$. First, for any vertex $v \in V$, it can obtain its degree $\deg(v)$ in constant time. Second, for any vertex $v \in V$ and any j such that $1 \leq j \leq \deg(v)$, the algorithm can obtain the label of the j -th neighbor of v in constant time.

The *query complexity* of an algorithm \mathcal{A} is the maximum number of queries made by \mathcal{A} to the input graph. Also, the conventional *time complexity* of \mathcal{A} refers to maximum running time of \mathcal{A} . \mathcal{A} is said to run in *constant time* if its time complexity is independent of the number n of vertices in the input graph.

Partitioning Oracles. The main tool in (polynomial-time) approximation algorithms for minor-free graphs is the separator theorem [9–11], which partitions a graph to two components with a small number of edges connecting them. It is used to partition the original problem into independent subproblems, which are tractable. Stitching together the optimal solutions for each of the independent subproblems results in an approximate solution for the original problem.

In [12], this intuition was used to design constant-time algorithms for various problems. Fix a partition P of the vertices in the input graph G such that the following properties hold:

1. Each connected component in the partition of the graph is small (say, has size $\text{poly}(1/\epsilon)$).
2. The number of edges connecting different connected components in the partition is less than $\epsilon|V|$.

Suppose now that we are given query access to such a partition P , i.e., we have a procedure \mathcal{O} which given a vertex v returns the connected component $P(v)$ that contains v . We call such a procedure a *partitioning oracle*. For a family of graphs \mathcal{F} , \mathcal{O} is a partitioning oracle for \mathcal{F} with parameter $\varepsilon > 0$ if it meets the following requirements:

- If $G \in \mathcal{F}$, then with probability $9/10$, the number of edges cut by the oracle is εn .
- The oracle provides a partition of G , even if $G \notin \mathcal{F}$.
- The partition P which \mathcal{O} provides access to is a function of only the input graph and random coin tosses of the oracle. In particular, P cannot be a function of the queries to the oracle¹.

We describe applications of partitioning oracles later, when we discuss the results that can be obtained using our oracle.

The main challenge here is to design efficient partitioning oracles that make few queries to the input graph and use little computation to answer every query about the partition. [12] shows how to design such an oracle for minor-free graphs (and also for some other hyperfinite graphs). However, their oracles have query complexity and running time of $2^{\text{poly}(1/\varepsilon)}$ (see [13] for a simple oracle with this property), and the main question left open by their paper is whether one can design a partitioning oracle that runs in time $\text{poly}(1/\varepsilon)$. In this paper we make partial progress by designing a partitioning oracle of complexity $\text{poly}(1/\varepsilon)$ for bounded-treewidth graphs. A similar open question is posed in [14], where they ask for a $\text{poly}(1/\varepsilon)$ -time tester for minor-closed properties. Constructing an efficient partitioning oracle for minor-free graphs would yield such a tester, but in general, such a tester need not be based on a partitioning oracle.

Our Main Result. The main result of the paper, an efficient partitioning oracle for bounded-treewidth graphs, is stated in the following theorem.

Theorem 1. *Let $G = (V, E)$ be a graph with maximum degree bounded by d . Let k be a positive integer. There is an oracle \mathcal{O} that given an $\varepsilon \in (0, 1/2)$, and query access to G , provides query access to a function $f : V \rightarrow 2^V$ of the following properties (where $k = O\left(\frac{d^3 \cdot h^{O(h)} \cdot \log(d/\varepsilon)}{\varepsilon^3}\right)$):*

1. For all $v \in V$, $v \in f(v)$.
2. For all $v \in V$, and all $w \in f(v)$, $f(v) = f(w)$.
3. If the treewidth of G is bounded by h , then with probability $9/10$, $|\{(v, w) \in E : f(v) \neq f(w)\}| \leq \varepsilon|V|$.
4. For all $v \in V$, $|f(v)| \leq k$.
5. For all v , the oracle makes $O(dk^{4h+7})$ queries to the input graph to answer a single query to the oracle, and the processing time is bounded by $\tilde{O}(k^{4h+O(1)} \cdot \log Q)$, where Q is the number of previous queries to the oracle.
6. The partition described by f is a function of G and random bits of the oracle, but does not depend on the queries to the oracle.

¹ This property allows algorithms to treat the partition P as fixed, even if it is not explicitly computed for the entire graph until sufficiently many queries are performed.

Applications. Partitioning oracles have numerous applications described in [12]. Let us describe some general applications of partitioning oracles, and the results yielded by our efficient partitioning oracle.

- **Testing minor-closed properties:** In *property testing* of graphs with maximum degree bounded by d , the goal is to distinguish graphs that have a specific property P from those that need to have at least εdn edges added and removed to obtain the property P , where $\varepsilon > 0$ is a parameter. Goldreich and Ron [8] show that the property of being a tree can be tested in $\tilde{O}(\varepsilon^{-3})$ time. Benjamini, Schramm, and Shapira [14] prove that any minor-closed property can be tested in $2^{2^{\text{poly}(1/\varepsilon)}}$ time. Hassidim *et al.* [12] introduce partitioning oracles and show how to use them to obtain a tester of running time $2^{\text{poly}(1/\varepsilon)}$ (see [13] for a simplified full proof). Yoshida and Ito [15] show that outerplanarity can be tested in $\text{poly}(1/\varepsilon)$ time. Via the reduction from [12], our new oracle yields a $\text{poly}(1/\varepsilon)$ -time tester for any minor closed family of graphs that has bounded treewidth. Sample minor-closed families of graphs with this property are k -outerplanar graphs, series-parallel graphs and pseudoforests. This also generalizes the result of Yoshida and Ito [15], since outerplanar graphs have treewidth at most 2.
- **Constant-time approximation algorithms:** Our oracle can also be used to obtain a $\text{poly}(1/\varepsilon)$ -time additive εn -approximation algorithm for the size of the maximum matching, minimum vertex cover, and minimum dominating set in (even unbounded) graphs with constant treewidth. See [12] for a general reduction. An important fact here is that for bounded-treewidth graphs, there are linear-time algorithms for computing the exact solutions to these problems [16]. This result adds to a long line of research on this kind of approximation algorithm [17–20, 12, 21, 22].
- **Testing properties of graphs with bounded treewidth:** Czumaj, Shapira, and Sohler [23] show that any hereditary property of bounded-degree bounded-treewidth graphs can be tested in constant time. This result is generalized by Newman and Sohler [22], who show that in fact any property of such graphs can be tested in constant time. Unfortunately, these two papers do not yield very efficient algorithms in general. Using our oracle and another reduction from [12], one can show that there are $\text{poly}(1/\varepsilon)$ -time algorithms for testing k -colorability and graph perfectness for these graphs. As before, one has to use efficient polynomial-time algorithms for these problems [16, 24] to solve the exact decision problems for sampled components in the reduction from [12].

1.1 Overview of Our Techniques

Let us briefly describe the main ideas behind the proof of our main result. Let G be a bounded-degree graph with treewidth h . We say that a vertex in G has a “good neighborhood” if a small set S of vertices including v can be disconnected from the graph by deleting at most $O(h)$ other vertices. Moreover, $O(h)/|S|$ is small.

First we show that most vertices have a good neighborhood. This follows by taking a tree decomposition of G , and showing a method that constructs a partition in which most vertices end up in connected components that can play the role of S for them in the original graph.

Then using the fact that a good neighborhood of small size t has a small border, i.e., of size $O(h)$, we show a procedure for enumerating all good neighborhoods for a given vertex. The procedure runs in $\text{poly}(dt)^{O(h)}$ time, where t is a bound on the size of the neighborhood. In particular, it can be used to check whether a given vertex has a good neighborhood and find it, if it exists.

Finally, we show a global partitioning algorithm that is likely to compute the desired partition of the graph. In this algorithm, each vertex v computes an arbitrary good neighborhood S_v containing it. If such a neighborhood does not exist, we set $S_v := \{v\}$ instead. Then we consider all vertices in V in a random order. In its turn, v removes all the remaining vertices in S_v from the graph. The set of vertices removed by v constitutes one (or a constant number) of the connected components in the partition of the input graph. This algorithm can easily be simulated locally and is the basis of our partitioning oracle.

Note: An anonymous reviewer suggested using known results on tree-partition-width [25, 26] to simplify our proofs. Those results can be used to give a simpler proof of a slightly worse decomposition than that in Lemma 1. For constant d , the worse decomposition still results in query complexity of $(1/\varepsilon)^{\text{poly}(h)}$. Unfortunately, in one application, namely the approximation of the minimum vertex cover size in graphs of arbitrary degree, the bound on d is a function of the parameter ε and eventually results in an algorithm with running time exponential in $\text{poly}(1/\varepsilon)$. Our construction results in a $\text{poly}(1/\varepsilon)$ -time algorithm.

Note 2: Due to space constraints, this version of the paper does not contain some of the proofs. The full paper is available on arXiv.

2 Definitions

Let $G = (V, E)$ be a graph and S be a subset of V . We write $N(S)$ to denote the set of vertices that are not in S and are adjacent to at least one vertex in S . We write $\eta(S)$ to denote the *cut-size* of S , which is defined as the size of $N(S)$, $\eta(S) = |N(S)|$. We write $\phi(S)$ to denote the *vertex conductance* of S , which is defined as $\phi(S) = \frac{\eta(S)}{|S|}$.

Definition 1. *Let $G = (V, E)$ be a graph. We say that $S \subseteq V$ is a neighborhood of v in G if $v \in S$ and the subgraph induced by S is connected. Given $k, c \geq 1$ and $\delta \in (0, 1)$, we say that S is a (k, δ, c) -isolated neighborhood of $v \in V$ if S is neighborhood of v in G , $|S| \leq k$, $\eta(S) \leq c$ and $\phi(S) \leq \delta$.*

Definition 2. Let $G = (V, E)$ be a graph and let A be a family of sets of vertices in G . A subfamily $B \subseteq A$ is a cover of A if for every set $T \in A$, $T \subseteq \bigcup_{S \in B} S$.

3 Local Isolated Neighborhoods in Constant Treewidth Graphs

The following lemma is at the heart of our proof. It shows that given a bounded-treewidth and bounded-degree graph, we can find an isolated neighborhood of v for almost every vertex v in the graph.

Lemma 1. Let $G = (V, E)$ be a graph with treewidth bounded by h and maximum degree bounded by d . For all $\varepsilon, \delta \in (0, 1/2)$, there exists a function $g : V \rightarrow 2^V$ with the following properties:

1. For all $v \in V$, $v \in g(v)$.
2. For all $v \in V$, $|g(v)| \leq k$, where $k = \frac{28860 d^3 (h+1)^5}{\delta \varepsilon^2}$.
3. For all $v \in V$, $g(v)$ is connected.
4. Let \mathcal{B} be the subset of V consisting of v such that $g(v)$ is a $(k, \delta, 2(h+1))$ -isolated neighborhood of v in G . The size of \mathcal{B} is at least $(1 - \varepsilon/20)|V|$.

Due to space constraints we do not present the proof of the lemma in this version of the paper. The starting point of our proof is a stronger version of the lemma for trees, which is also easier to prove. Later, to obtain a proof of Lemma 1, we apply the stronger lemma for trees to a canonical tree decomposition of a given bounded-treewidth graph.

4 Isolated Neighborhoods

In this section we show how to discover isolated neighborhoods efficiently. We also prove an upper bound on the number of incomparable isolated neighborhoods covering a specific vertex.

4.1 Finding an Isolated Neighborhood of a Vertex

The following lemma states that isolated neighborhoods with small cut-size in bounded-degree graphs can be found efficiently. To this end, we use the procedure **Find-Neighborhood** described as Algorithm 1. We omit the proof in this version of the paper.

Lemma 2. Let $G = (V, E)$ be a graph with maximum degree bounded by d . Given a vertex $v \in V$, integers $k, c \geq 1$ and $\delta \in (0, 1)$, procedure **Find-Neighborhood** finds a (k, δ, c) -isolated neighborhood of v in G , provided it exists. If no such isolated neighborhood exists, the algorithm returns $\{v\}$. The algorithm runs in $\text{poly}(dk) \cdot k^c$ time and makes $O(dk^{c+1})$ queries to the graph.

Algorithm 1. Procedure $\text{Find-Neighborhood}(v, k, \delta, c)$

```

1 Run BFS from  $v$  until it stops or exactly  $k$  vertices are visited
2 Let  $S$  be the set of vertices reached by the BFS
3 if  $S$  is a  $(k, \delta, c)$ -isolated neighborhood in the original graph then
4   return  $S$ 
5 if  $c > 0$  then
6   foreach  $w \in S \setminus \{v\}$  do
7     Remove  $w$  from the graph
8      $S' := \text{Find-Neighborhood}(v, k, \delta, c - 1)$ 
9     Insert  $w$  back into the graph
10    if  $S' \neq \{v\}$  then return  $S'$ 
11 else
12   return  $\{v\}$ 

```

4.2 Finding Isolated Neighborhoods Covering a Vertex

Let v and u be vertices in a graph $G = (V, E)$. When the values of k, δ, c are clear from context, we say that u covers v if the isolated neighborhood found by $\text{Find-Neighborhood}(u, k, \delta, c)$ contains v . The following lemma states that one can efficiently find all vertices that cover a given vertex. In this version of the paper, we omit the proof that uses a modified version of Find-Neighborhood .

Lemma 3. *Let $G = (V, E)$ be a graph with maximum degree bounded by d . There is an algorithm that given a vertex $v \in V$, integers $k, c \geq 1$ and $\delta \in (0, 1)$, finds all u that cover v in $\text{poly}(cdk) \cdot k^{2c}$ time and with $O(dk^{2(c+1)})$ queries.*

4.3 Small Cover of Isolated Neighborhoods

Recall that a cover for a family \mathcal{A} of subsets of vertices is any subset $\mathcal{B} \subseteq \mathcal{A}$ such that $\bigcup_{T \in \mathcal{A}} T = \bigcup_{T \in \mathcal{B}} T$. We now show that a family of isolated neighborhoods of a vertex has a relatively small cover.

Lemma 4. *Let \mathcal{A} be a family of isolated neighborhoods of a vertex v in a graph $G = (V, E)$. Let the cut-size of all the neighborhoods be bounded by an integer c . There is a cover $\mathcal{B} \subseteq \mathcal{A}$ of size at most $(c + 1)!$.*

Proof. We assume that \mathcal{A} is non-empty. Otherwise, the lemma holds trivially. We prove the lemma by induction on c . For $c = 0$, any isolated neighborhood of v is the connected component containing v . Therefore, \mathcal{A} consists of one set, $(c + 1)! = 1$, and the lemma holds.

For the inductive step, assume that $c > 0$ and that the lemma holds for cut-size bounds lower than c . Without loss of generality, there is a vertex $u \in V$ that belongs to exactly one set $S \in \mathcal{A}$. Otherwise, we can keep removing arbitrary sets from \mathcal{A} , one by one, until this becomes the case, because every cover for the pruned family of neighborhoods is still a cover for the original family.

For all $T \in \mathcal{A}$, let G_T be the subgraph of G induced by the vertices in $T \setminus S$. For each $w \in N(S)$, we construct a family \mathcal{A}_w of neighborhoods of w as follows. We consider each $T \in \mathcal{A}$. If $w \in T$, we add to \mathcal{A}_w the set X of vertices in the connected component of G_T that contains w . In this case, we say that T was the progenitor of X .

Let G' be the subgraph of G induced by $V \setminus S$. We claim that for each $w \in N(S)$ and each $Y \in \mathcal{A}_w$, the cut-size of Y in G' is bounded by $c - 1$. Let $Z \in \mathcal{A}$ be the progenitor of Y . Every vertex that belongs to the neighbor set of Y in G' also belongs to the neighbor set of Z in G . Moreover, since $w \in Z$, $Z \neq S$, and does not contain u . Therefore, there is a vertex in S that belongs to $N(Z)$ in G . This vertex does not appear in G' , which implies it does not belong to $N(Y)$ in G' . Due to our earlier observation that $N(Y)$ in G' is a subset of $N(Z)$ in G , $|N(Y)| \leq c - 1$.

We construct \mathcal{B} as follows. We start with an empty set and insert S into \mathcal{B} . By the inductive hypothesis, for every $w \in N(S)$, there is a cover \mathcal{B}_w of size at most $c!$ for \mathcal{A}_w . For each neighborhood Z in each of these covers, we add to \mathcal{B} , the progenitor of Z . This finishes the construction of \mathcal{B} . The size of \mathcal{B} is bounded by $|N(S)| \cdot c! + 1 \leq c \cdot c! + 1 \leq (c + 1)!$.

It remains to show that \mathcal{B} is a cover of \mathcal{A} . Consider any vertex $t \in \bigcup_{Z \in \mathcal{A}} Z$. If t belongs to S , we are done. Otherwise, let $Z_1 \in \mathcal{A}$ be such that $t \in Z_1$. There is a path in G that goes from v to some $w \in N(S)$ via vertices in S , and then using only vertices in $Z_1 \setminus S$ it goes from w to t . Let G_\star be the subgraph of G induced by $Z_1 \setminus S$. Let Y_1 be the set of vertices in the connected component of G_\star that contains w . Due to the path from w to t , $t \in Y_1$. By definition, $Y_1 \in \mathcal{A}_w$, so $t \in \bigcup_{Y \in \mathcal{A}_w} Y = \bigcup_{Y \in \mathcal{B}_w} Y$. Let $Y_2 \in \mathcal{B}_w$ be any set containing t . Its progenitor $Z_2 \supseteq Y_2$ belongs to \mathcal{B} , and therefore, $t \in \bigcup_{Z \in \mathcal{B}} Z$, which finishes the proof. \square

Corollary 1. *Let v be a vertex in a graph. Let $k, c \geq 1$ be integers and let $\delta \in (0, 1)$. The number of vertices u , for which $u \in \text{Find-Neighborhood}(u, k, \delta, c)$ is bounded by $k \cdot (c + 1)!$.*

5 The Partitioning Oracle (Proof of Theorem [1](#))

Here we prove the main claim of the paper. We show a global partitioning algorithm that can easily and efficiently be simulated locally. The global algorithm is likely to find the desired partition of the input graph.

Proof. We want to set our parameter δ and k , so that the following inequalities hold:

$$\delta \leq \frac{\varepsilon}{100 \cdot (2h + 3)! \cdot (1 + \log k + \log(2h + 3))}, \tag{1}$$

$$k \geq \frac{28860 d^3 (h + 1)^5}{\delta \varepsilon^3}. \tag{2}$$

By combining them, we obtain

$$k \geq \frac{2886000 \cdot d^3 \cdot (2h + 3)! \cdot (1 + \log k + \log(2h + 3)!)}{\varepsilon^3}.$$

Algorithm 2. Global-Partitioning(k, δ, h)

```

1 forall  $v \in V$  do set  $v$  as not marked
2 foreach  $v \in V$  do
3    $S_v := \text{Find-Neighborhood}(v, k, \delta, 2(h + 1))$ 
4    $r_v :=$  uniformly random value in  $(0, 1)$ 
5 foreach  $v \in V$  in increasing order of  $r_v$  do
6    $U := \{w \in S_v : w \text{ is not marked}\}$ 
7   forall  $w \in U$  do  $f[w] := U$ 
8   Mark all vertices in  $U$ 
9 Output  $f$ 

```

Algorithm 3. Local-Partitioning(q, k, δ, h) for a vertex q

```

1  $Q_q :=$  the set of vertices that cover  $q$  (see Lemma 3)
2 Let  $u$  be the vertex in  $Q_q$  with the lowest  $r_u$ 
3  $S_u := \text{Find-Neighborhood}(u, k, \delta, 2(h + 1))$ 
4  $P := \emptyset$ 
5 foreach  $w \in S_u$  do
6    $Q_w :=$  the set of vertices that cover  $w$  (see Lemma 3)
7   Let  $u_w$  be the vertex in  $Q_w$  with the lowest  $r_{u_w}$ 
8   if  $u = u_w$  then  $P := P \cup \{w\}$ 
9 return  $P$ 

```

This can be satisfied by values of k that do not grow faster than

$$O\left(\frac{d^3 \cdot h^{O(h)} \cdot \log(d/\varepsilon)}{\varepsilon^3}\right).$$

The we take the maximum δ that satisfies Equation 11.

Consider the global partitioning algorithm described as Algorithm 2 with parameters set as described above. The algorithm constructs a partition of the vertices in the graph. It starts with all vertices in the graph unmarked. For each vertex $v \in V$, the algorithm tries to find a $(k, \delta, 2(h + 1))$ -isolated neighborhood of v in G . If one such neighborhood is found, S_v is set to that neighborhood. Otherwise, if no such neighborhood exists, S_v is set to $\{v\}$. Next the algorithm starts partitioning the graph. It considers the vertices in random order. For each vertex v in that order, all the unmarked vertices in S_v constitute a component in the partition and get marked.

Clearly, at the end of the execution of Algorithm 2, all vertices must be marked. Therefore, $f(v)$ is well defined for all $v \in V$. Also, observe that when $f(v)$ is defined, we have $f(u) = f(v)$ for every $u \in f(v)$. Therefore, Claims 1 and 2 of the theorem hold for the function f computed by Algorithm 2.

Let us bound the probability that the number of edges between different parts is greater than $\varepsilon|V|$. By Lemma 11 with the lemma's ε set to ε/d , there exists a function $g : V \rightarrow \mathcal{P}(V)$ with the following properties:

1. For all $v \in V$, $v \in g(v)$.
2. For all $v \in V$, $|g(v)| \leq k$.
3. For all $v \in V$, $g(v)$ is connected.
4. Let \mathcal{B} be the subset of V such that $v \in \mathcal{B}$ if and only if $g(v)$ is a $(k, \delta, 2(h+1))$ -isolated neighborhood of v in G . The size of \mathcal{B} is at least $(1 - \varepsilon/20d)|V|$.

Let us group the edges between different components in the partition given by f . We distinguish two kinds of edges: the edges incident to at least one vertex in $V \setminus \mathcal{B}$ and the edges with both endpoints in \mathcal{B} . Observe that the total number of the former edges is at most $\frac{\varepsilon}{20}|V|$. It remains to bound the number of the latter edges that are cut. Consider a vertex $v \in \mathcal{B}$. Let Q_v be the set of vertices that cover v and let $m_v = |Q_v|$. Via Corollary [11](#), $m_v \leq k \cdot (2h + 3)!$. Let $q_1, q_2, \dots, q_{m_v} \in Q_v$ be the sequence of vertices that cover v in increasing order of r , i.e., $r_{q_1} \leq r_{q_2} \leq \dots \leq r_{q_{m_v}}$. For each $j \in \{1, 2, \dots, m_v\}$, let $S_v^{(j)} = \{S_{q_1}, S_{q_2}, \dots, S_{q_j}\}$, where S_{q_i} is the isolated neighborhood found by **Find-Neighborhood** starting from q_i . Note that, since r is random, $S_v^{(j)}$ and q_j are random variables for all $j \in \{1, 2, \dots, m_v\}$.

For vertices $u, v \in \mathcal{B}$, we say u *marks* v if $v \in S_u$ and v is not marked before u is considered. Also, we say a vertex $u \in \mathcal{B}$ is *marking* if u marks some vertex in the graph. It is clear from the definition that, for any $j \in \{1, 2, \dots, m_v\}$, if q_j is marking then $S_{q_j} \not\subseteq \bigcup_{i=1}^{j-1} S_{q_i}$. This implies that if q_j is marking, S_{q_j} must be a member of every cover of $S_v^{(j)}$. By Lemma [4](#), there exists a cover $B_j \subseteq S_v^{(j)}$ such that $|B_j| \leq (2h + 3)!$. Thus, whatever the first j sets $S_v^{(j)}$ are, the probability that j -th vertex q_j is marking is bounded by $(2h + 3)!/j$. Let the number of marking vertices in Q_v be a_v . We have

$$\begin{aligned}
 E[a_v] &= \sum_{j=1}^{m_v} \Pr[q_j \text{ is marking}] \leq \sum_{j=1}^{m_v} \frac{(2h + 3)!}{j} \leq (2h + 3)! \cdot (1 + \log m_v) \\
 &\leq (2h + 3)! \cdot (1 + \log k + \log(2h + 3)!).
 \end{aligned}$$

Let $M \subseteq \mathcal{B}$ be the set of marking vertices in the graph. It holds

$$\sum_{u \in M} |S_u| = \sum_{u \in \mathcal{B}} a_u.$$

Thus,

$$E \left[\sum_{u \in M} |S_u| \right] \leq (2h + 3)! \cdot (1 + \log k + \log(2h + 3)!) \cdot |\mathcal{B}|.$$

Note that, for each marking vertex u , the number of edges that have exactly one end in S_u is at most $d|S_u|\delta$. This is also an upper bound for the number of edges going out of u 's component in the partition. Therefore, the expected number of cut edges with both ends in \mathcal{B} is at most

$$\delta d \cdot (2h + 3)! \cdot (1 + \log k + \log(2h + 3)!) \cdot |\mathcal{B}| \leq \frac{\varepsilon}{100}|V|.$$

Thus, by Markov inequality, the probability that the number of edges in the second set is greater than $\frac{\varepsilon}{10}|V|$ is at most $1/10$. Therefore, the probability that the total number of edges in both sets is less than $\varepsilon|V|$ is at least $9/10$, as required by Claim 3.

Finally, observe that the size of each partition is trivially bounded by k , which is required by Claim 4.

We now show how Algorithm 2 can be simulated locally. Consider the local Algorithm 3 that given a query $q \in V$, computes $f[q]$. The local algorithm starts by computing the set of vertices that cover q . Then among the vertices in this set, it finds the vertex u with the smallest value r_u . It is clear that u is the vertex that marks q , and thus, $f[q]$ is a subset of S_u . Next the local algorithm considers each vertex in S_u and checks whether that vertex is also marked by u . If it is, the vertex should also be included in $f[q]$. Clearly local Algorithm 3 computes exactly the same set $f[q]$ as the global Algorithm 2, provided the selected random numbers are the same.

Let us bound the number of queries to the input graph that Algorithm 3 makes to answer query q :

- Lemma 3 shows that finding Q_q , the set of vertices that cover q , requires $O(dk^{2h+4})$ queries to the input graph.
- By Lemma 2, finding S_u takes at most $O(dk^{2h+3})$ queries to the input graph.
- Finally, for each $w \in S_u$, checking whether w is marked by u also takes $O(dk^{4h+6})$ queries to the input graph. Since $|S_u| \leq k$, it takes at most $O(dk^{4h+7})$ queries to find the subset of vertices in S_u that are marked by u .

Summarizing, the oracle has to make at most $O(dk^{4h+7})$ queries to the input graph to answer any query about f . Similarly, we assume that a dictionary operation requires $O(\log n)$ time for a collection of size n . We use a dictionary to keep the previously generated r_v in order to maintain consistency. The running time of the oracle to answer a query about f is then at most $\tilde{O}(k^{4h+O(1)} \cdot \log Q)$, as stated in Claim 5 of the theorem.

Finally, the partition is computed such that it is independent of queries to the oracle. It is only a function of coin tosses that correspond to Step 5 of Algorithm 2. \square

References

1. Dinur, I., Safra, S.: On the hardness of approximating minimum vertex cover. *Annals of Mathematics* 162(1), 439–485 (2005)
2. Bodlaender, H.: Dynamic programming on graphs with bounded treewidth. In: *Automata, Languages and Programming*, pp. 105–118 (1988)
3. Robertson, N., Seymour, P.: Graph minors. III. Planar tree-width. *Journal of Combinatorial Theory, Series B* 36(1), 49–64 (1984)

² As a technical requirement to make the argument valid, we also assume that the loop in Step 6 of Procedure **Find-Neighborhood** considers vertices of the input graph in order independent of queries to the oracle.

4. Robertson, N., Seymour, P.: Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms* 7(3), 309–322 (1986)
5. Bodlaender, H.: A tourist guide through treewidth. *Developments in Theoretical Computer Science* (1994) 1
6. Atserias, A.: On digraph coloring problems and treewidth duality. *European Journal of Combinatorics* 29(4), 796–820 (2008)
7. Bodlaender, H., Koster, A.: Combinatorial optimization on graphs of bounded treewidth. *The Computer Journal*, 1 (2007)
8. Goldreich, O., Ron, D.: Property testing in bounded degree graphs. *Algorithmica* 32(2), 302–343 (2002)
9. Lipton, R.J., Tarjan, R.E.: A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics* 36, 177–189 (1979)
10. Lipton, R.J., Tarjan, R.E.: Applications of a planar separator theorem. *SIAM J. Comput.* 9(3), 615–627 (1980)
11. Alon, N., Seymour, P.D., Thomas, R.: A separator theorem for graphs with an excluded minor and its applications. In: *STOC*, pp. 293–299 (1990)
12. Hassidim, A., Kelner, J.A., Nguyen, H.N., Onak, K.: Local graph partitions for approximation and testing. In: *FOCS*, pp. 22–31 (2009)
13. Onak, K.: *New Sublinear Methods in the Struggle Against Classical Problems*. PhD thesis. Massachusetts Institute of Technology (2010)
14. Benjamini, I., Schramm, O., Shapira, A.: Every minor-closed property of sparse graphs is testable. In: *STOC*, pp. 393–402 (2008)
15. Yoshida, Y., Ito, H.: Testing outerplanarity of bounded degree graphs. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) *APPROX 2010, LNCS*, vol. 6302, pp. 642–655. Springer, Heidelberg (2010)
16. Arnborg, S., Proskurowski, A.: Linear time algorithms for NP-hard problems restricted to partial k -trees. *Discrete Applied Mathematics* 23(1), 11–24 (1989)
17. Parnas, M., Ron, D.: Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theor. Comput. Sci.* 381(1–3), 183–196 (2007)
18. Marko, S., Ron, D.: Approximating the distance to properties in bounded-degree and general sparse graphs. *ACM Transactions on Algorithms* 5(2) (2009)
19. Nguyen, H.N., Onak, K.: Constant-time approximation algorithms via local improvements. In: *FOCS*, pp. 327–336 (2008)
20. Yoshida, Y., Yamamoto, M., Ito, H.: An improved constant-time approximation algorithm for maximum matchings. In: *STOC*, pp. 225–234 (2009)
21. Elek, G.: Parameter testing in bounded degree graphs of subexponential growth. *Random Struct. Algorithms* 37(2), 248–270 (2010)
22. Newman, I., Sohler, C.: Every property of hyperfinite graphs is testable. In: *STOC*, pp. 675–684 (2011)
23. Czumaj, A., Shapira, A., Sohler, C.: Testing hereditary properties of nonexpanding bounded-degree graphs. *SIAM J. Comput.* 38(6), 2499–2510 (2009)
24. Chudnovsky, M., Cornuéjols, G., Liu, X., Seymour, P.D., Vuskovic, K.: Recognizing Berge graphs. *Combinatorica* 25(2), 143–186 (2005)
25. Ding, G., Oporowski, B.: Some results on tree decomposition of graphs. *Journal of Graph Theory* 20(4), 481–499 (1995)
26. Wood, D.R.: On tree-partition-width. *European Journal of Combinatorics* 30(5), 1245–1253 (2009); Part Special Issue on Metric Graph Theory

Inflatable Graph Properties and Natural Property Tests*

Eldar Fischer and Eyal Rozenberg

Department of Computer Science, Technion, Haifa, Israel
{eldar, eyalroz}@cs.technion.ac.il

Abstract. We consider *natural* graph property tests, which act entirely independently of the size of the graph being tested. We introduce the notion of properties being *inflatable* — closed under taking (balanced) blowups — and show that the query complexity of natural tests for a property is related to the degree to which it is approximately hereditary and inflatable. Specifically, we show that for properties which are almost hereditary and almost inflatable, any test can be made natural, with a polynomial increase in the number of queries. The naturalization can be considered as an extension of the canonicalization due to [15], so that natural canonical tests can be described as *strongly canonical*.

Using the technique for naturalization, we restore in part the claim in [15] regarding testing hereditary properties by ensuring that a small random subgraph itself satisfies the property. This allows us to generalize the triangle-freeness lower bound result of [5]: Any lower bound, not only the currently established quasi-polynomial one, on one-sided testing for triangle-freeness holds essentially for two-sided testing as well. We also explore the relations of the notion of inflatability and other already-studied features of properties and property tests, such as one-sidedness, heredity, and proximity-oblivion.

1 Introduction

In Property Testing, one is allowed oracle access to some combinatorial object, and must distinguish with high probability between the case of this object satisfying a certain property, and the case of the object being far from satisfying it. The study of property testing in the context of graphs began with [12], which introduced the *dense model* for graph testing: graphs on n vertices are close if one needs to add and/or remove $\varepsilon \binom{n}{2}$ edges to convert one into the other.

Much of the work in this field involves characterizing which graph properties admit tests in this model in certain *query complexity* classes with respect to n and ε , and with certain features of the tests. While [12] and some later contributions (specifically, [13]; see also below) involve bounds in terms of n — most study of the dense model has focused on testing with query complexity depending only on the distance parameter ε (called simply “testable” properties). A

* Eldar Fischer’s research supported in part by ERC-2007-StG grant no. 202405 and by an ISF grant no. 1101/06.

large class of properties were established in [12] as testable, and the characterization of the class of testable properties was posed as an open problem. Over the following decade, a series of results gradually progressed towards this goal, with the characterization being finally achieved in [4], and independently in [7] (in terms of graph limits).

Notable on the way to characterizing the testable properties was [6], which proved that all *hereditary* graph properties have tests with one-sided error (i.e. which never reject graphs with the property). This kind of link between a feature of a class of properties and a feature of the test is a main focus of this paper.

Beyond the fundamental testability of properties, much study has focused on the query complexity's specific dependence on the distance parameter ε . Most general testability results in the dense model are based on the use of Szemerédi's regularity lemma, incurring a prohibitive dependence on ε , while the generalized partition properties proved testable in [12] have a mere polynomial dependence on ε in the number of queries.

Links between features of properties and features of tests have also been studied in this context, with several important results appearing in [15]. This paper introduced the notion of *canonical* testing: A canonical test uniformly samples a small induced subgraph, queries it entirely, and decides deterministically based on this subgraph. [15] proved that any test can be made canonical with at most about a squaring of its number of queries, immediately implying that the gap between adaptive and non-adaptive query complexity (see [14]) is at most quadratic. [15] also included a proposition regarding hereditary testable properties: These can be tested by merely ensuring that most small induced subgraphs themselves satisfy the property. Unfortunately, it later turned out that this result only holds for tests which are *natural*: Tests acting independently of the size of the input graph. This qualification appears in the errata [16].

How essential is this qualification? If we prevent properties from exhibiting blatant 'pathologies' precluding natural tests (e.g. the property of graphs having an odd number of vertices) — one would hope it may be possible to 'smooth out' any artificial dependence on n . Indeed, in this paper we show that tests can be made natural, with a polynomial penalty in the number of queries, for properties with appropriate features; we also justify their being appropriate by showing that properties admitting natural tests also exhibit these features albeit more weakly.

The first constraining feature is heredity, mentioned above. With a hereditary property, as n increases, the set of forbidden subgraphs gains more and more elements, so one expects the set of acceptable queried graphs to shrink gradually. The idea for the converse feature, implying a set of acceptable graphs shrinking as n decreases, is motivated by earlier works on lower bound results. The concept which springs to mind is graph blowups: The hierarchy theorems in [13] rely on the use of blowups; and the triangle testing lower bound in [2] involves a blowup (more on this below). The feature we define is being *inflatable* — being closed to blowups. This specifically prevents the pathology of going from satisfying a property at order n to being very far from it by merely adding a vertex.

With regards to the idea of ‘smoothing out’ non-naturality, a typical example would be a test which arbitrarily rejects some specific queried subgraph at even orders, and accepts it at odd ones. If this subgraph is very unlikely to appear in graphs in the property, a natural test could be ‘spoiled’ by adding this behavior to it, while still remaining a valid test. However, such behavior is not possible with enough potential queried subgraphs to have an overall high probability of being sampled. This leads one to recall the proof in [9] that testable properties are also estimable (a key result necessary for the characterization in [4]). The essential argument in [9] is, that with a good estimate of the subgraph distribution, one knows the probability of a test querying subgraphs of this order accepting.

An immediate noteworthy application of our naturalization technique regards lower bounds on the query complexity of testing triangle-freeness (or induced-subgraph-freeness in general). While triangle-freeness is known to be testable, there is a vast gap between the lower and upper bounds for it: The best upper bounds are based on Szemerédi’s regularity lemma ([1]; see [8] and [3]), or similar techniques (as in the recent improvement [11]); these yield a query complexity being a tower function whose height is a polynomial or a logarithmic in $1/\varepsilon$.

Lower bounds for a property’s query complexity can be established with Yao’s principle (see [17]), using two distributions which are hard to distinguish, one supported mostly on satisfying graphs and the other mostly on far graphs. But even the construction of just a single graph which is hard to distinguish as being far from a property can yield lower bound results. For one-sided testing of triangle-freeness, one notes that a one-sided test, querying a triangle-free subgraph, would have to accept. Indeed, such a construction in [2] established the best such bound known, slightly super-polynomial in $1/\varepsilon$.

A one-sided query complexity lower bound can be converted into a two-sided lower bound, provided one can show that any test for the property can be made one-sided. Indeed, this was to be possible using the proposition appearing in [15] — but as mentioned above, it relies on the test being natural. This problem is worked around in [5], which proves a quasi-polynomial lower bound for any triangle freeness test — directly, using Yao’s method. That proof, however, is custom-tailored, and may not be able to convert stronger one-sided lower bounds to general two-sided bounds. Using our naturalization technique, we restore in part the proposition regarding testing hereditary properties, proving that the query complexity of testing for triangle-freeness is entirely determined by the one-sided-error query complexity.

The rest of this paper is organized as follows. Following the necessary formal definitions and some technical preliminaries in [Section 2](#), our main result regarding naturalization, as well as its significant implications, are stated in [Section 3](#). Proof of our main [Theorem 1](#) constitutes [Section 4](#). For brevity, an additional section discussing inflatability and natural testability, as well as most proofs other than for our main result, have been omitted; they may be found in the full version of this paper, available online: [\[10\]](#).

2 Preliminaries

2.1 Graph Properties and the Dense Model for Property Testing

Definition 1. *The absolute distance between two graphs G, H of order n is the number of edges one has to add and/or remove in G to make it into an isomorphic copy of H ; in other words, it is the minimum over all bijections $\phi : V(G) \rightarrow V(H)$ of the number of edge discrepancies — the number of elements in the symmetric difference between $E(H)$ and $\{\{\phi(u'), \phi(v')\} \mid \{u', v'\} \in E(G)\}$. The distance $\text{dist}(G, H)$ between G and H is the absolute distance between them normalized by a factor of $\binom{n}{2}^{-1}$.*

Definition 2. *A property of graphs is a set $\Pi = \bigcup_{n=1}^{\infty} \Pi_n$ of graphs, closed under graph isomorphism, where Π_n is supported on graphs of order n .*

A graph is set to satisfy a property Π if it is an element of the set Π ; a graph G of order n is said to be ε -far from satisfying a property Π if G 's distance from every graph $H \in \Pi_n$ is at least ε .

Definition 3. *A property test for a graph property Π is a probabilistic oracle machine which, given the values (n, ε) , as well oracle access to a graph G of order n , makes a certain number of edge queries (“is there an edge between the vertices u and v ?”), and distinguishes with probability at least $2/3$ between the case of G being in Π and the case of G being ε -far from Π . The (possibly adaptive) number and choice of queries, as well as the rest of the algorithm, may in general depend on the value of n , as can the decision to accept or reject.*

This traditional definition of a property test in the dense model includes an artificial dependence of the query model on the value of n : Without utilizing this value it is not possible to make any samples; [15, Section 4] emphasizes the artifice of this particular dependence, and lead us to the following definition, equivalent in the context of graph properties:

Definition 4 (Alternative to Definition 3). *A property test for a graph property Π is a probabilistic oracle machine which is given the values (n, ε) , as well access to a graph G of order n , through an oracle which takes two types of requests: A request to uniformly sample an additional vertex out of the remaining vertices of G , and an edge query within the subgraph induced by the sampled vertices (“is there an edge between the i^{th} and j^{th} sampled vertices?”). The machine makes a sequence of requests to the oracle, and distinguishes with probability at least $2/3$ between the case of G being in Π and the case of G being ε -far from Π . If the test has sampled the entire input graph, additional requests to sample an additional vertex will indicate that there are none left.*

2.2 Features of Property Tests

Definition 5. *A property test is said to be one-sided (or said to have one-sided error) if it accepts all graphs in Π with probability 1.*

Definition 6. A test for a property Π is said to be canonical if, for some $s : \mathbb{N} \times (0, 1) \rightarrow \mathbb{N}$ and properties $(\Pi^{(i)})_{i=1}^{\infty}$, the test operates as follows: on input n and oracle access to an n -vertex graph G , the test samples uniformly a set of $s(n, \varepsilon)$ distinct vertices of G , queries the entire corresponding induced subgraph and accepts if and only if this subgraph is in $\Pi^{(n)}$. If there are fewer than $s(n, \varepsilon)$ vertices, the test queries the entire graph and accepts if it is in Π .

Theorem ([15, Theorem 2]). If a graph property has a test making queries involving at most $s(\varepsilon)$ vertices, independently of the size of the input graph, then it has a canonical test with queried subgraph order at most $9s(\varepsilon)$. If the original test is one-sided, this canonical test's queried subgraph order is $s(\varepsilon)$ and it is also one-sided.

Note. The statement in [15] does not mention the number of sampled vertices; in the proof, however, the original test is repeated 9 times and the majority-vote is used, to amplify the probability of success to $1/6$; see also [16, Page 2].

A canonical test, which accepts a graph G when the queried subgraph on its sampled vertices is G' , is said to accept G by sample G' .

Definition 7 (as appearing in [16]). A graph property test is said to be natural if its query complexity is independent of the size of the tested graphs, and on input (n, ε) and oracle access to a graph of order n , the test's output is based solely on the sequence of oracle answers it receives (while possibly using more random bits, provided that their number is also independent of n).

If our graph property tests are as defined traditionally (Definition 3), the above definition of a natural test is flawed, and no test which makes any queries can be natural: A test cannot make $q(\varepsilon)$ queries to an input graph with less than $\sqrt{q(\varepsilon)}$ vertices (this point is also mentioned in [6]). Instead of amending the definition of naturality to avoid this semantic issue, it seems more reasonable to use the alternative definition for the dense graph model, Definition 4, in which the artificial dependence on n is removed. In this case, Definition 7 is valid: If the test attempts to sample too many vertices, the oracle indicates its failure to do so and the test proceeds accordingly.

In this paper we will be dealing mostly with tests which combine both the above features, or rather, we will focus on making canonical tests natural as well. In the context of a canonical test, the naturalness means that the 'internal' property, the one for which the sampled subgraph is checked for, does not depend on the order of the input graph. This observation leads us to use naturality to define several 'levels' of canonicity for a property test:

Definition 8. Consider a canonical test for graph property Π , with $(\Pi^{(i)})_{i=1}^{\infty}$ being the sequence of properties the satisfaction of which the test checks for its sampled order- s subgraph. The test is said to be

perfectly canonical when $\Pi^{(i)} = \Pi$ — The test merely ensures that a small random subgraph satisfies the property the input graph is being tested for.

strongly canonical when $\Pi^{(i)} = \Pi'$ — The test ensures a small sampled sub-graph satisfies some fixed property, the same for any order of the input graph.
 weakly canonical for any $(\Pi^{(i)})_{i=1}^\infty$, possibly different at different orders i .

Indeed, a test is strongly canonical if and only if it is both canonical and natural.

2.3 Features of Graph Properties

In this subsection we define strict and approximate notions of graph properties being hereditary and inflatable.

Definition 9. A graph $G' = (V', E')$ is a (balanced) blowup of a graph $G = (V, E)$ if V' can be partitioned into $|V|$ clusters of vertices, all of the same size up to a difference of at most 1, each corresponding to a vertex in V , where the edges in E' between these clusters correspond to the edges of E . In other words, if $(u, v) \in E$ then the bipartite graph between the clusters corresponding to u and v is complete, and if $(u, v) \notin E$ then this bipartite graph is empty. There are no edges inside each partition set.

Definition 10. If $G' = (V', E')$ is a blowup of G , and the clusters in V' (corresponding to the vertices of G) all have exactly the same size (and, in particular, $|V|$ divides $|V'|$), then G' is said to be an exactly-balanced blowup.

We will sometimes refer to a “random” or a “uniformly sampled” blowup of a graph from order n to some order n' ; this will mean that the $n' \pmod n$ vertices which have the larger clusters in the blowup (clusters of size $\lceil n'/n \rceil$ rather than $\lfloor n'/n \rfloor$) are chosen at random.

Lemma 1. Let $G \neq H$ be graphs of order n , let $n' > n$ and let $\phi : V(G) \rightarrow V(H)$ be a bijection achieving $\text{dist}(G, H)$, i.e. exhibiting $\text{dist}(G, H) \cdot \binom{n}{2}$ discrepancies. If one uniformly samples a blowup G' of G to order n' , and applies the same blowup to H — in the sense that for every $v \in G$, the size of v 's cluster in G' is the same as the size of $\phi(v)$'s cluster in the blowup H' of H — then the expected distance between the two blowups is strictly lower than $\text{dist}(G, H)$.

A proof of the above lemma, as well as of all lemmata in this section, appears in the full version of this paper: [10].

Definition 11. A property Π is said to be inflatable if it is closed under blowups, i.e. if G satisfies Π , then so does any blowup of G .

Definition 12. A graph property Π is said to be (s, δ) -inflatable if for any graph G satisfying Π , of order at least s , all blowups of G are δ -close to satisfying Π . A property Π is said to be (s, δ) -inflatable on the average if for any graph G satisfying Π , of order at least s , the expected distance from Π of blowups of G to any fixed order (uniformly sampled out of all possible blowups) is less than δ .

Since blowups do not affect graph distances overmuch, not only graphs satisfying an inflatable property remain close to it, but rather, the distance of any graph from the property does not increase much by taking a blowup.

Definition 13. A graph property is said to be hereditary if it is closed under the taking of induced subgraphs. A property is said to be hereditary down to order n_0 if it is closed under the taking of induced subgraphs of order no less than n_0 .

Definition 14. A property Π is said to be (s, δ) -hereditary if, for every graph in Π , all of its induced subgraphs of order at least s are δ -close to Π . Π is said to be (s, δ) -hereditary on the average if, for every graph in Π , the expected distance from Π of a uniformly-sampled subgraph of any fixed order $s' \geq s$ is less than δ .

2.4 Fixed-Order Subgraph Distributions of Graphs

Definition 15. Given a graph G , consider the graph induced by a uniformly sampled subset of s vertices. We denote the distribution of this induced subgraph by D_G^s , the order- s subgraph distribution of G ; $D_G^s(G')$ is the relative frequency of a subgraph G' of order s in G .

Definition 16. Let \mathcal{G}^s denote all graphs of order s . The distance between two distributions D, D' over graphs of order s , denoted $\text{dist}(D, D')$, is the variation distance between them, i.e. $\text{dist}(D, D') = \frac{1}{2} \sum_{G \in \mathcal{G}^s} |D(G) - D'(G)|$.

Lemma 2. If two graphs G, H (of order $n \geq s$) are $\delta \binom{s}{2}^{-1}$ -close, then their order- s subgraph distributions are δ -close, i.e. $\text{dist}(D_G^s, D_H^s) \leq \delta$.

Lemma 3. Let $\delta > 0$, let G be a graph of order $n \geq \frac{2}{\delta} \binom{s}{2}$, let G' be a random blowup of G to order $n' > n$, and let $\mathcal{H} \subseteq \mathcal{G}^s$. Then

$$\left| \mathbf{E}_{G'} \left[\mathbf{Pr}_{H \sim D_{G'}^s} [H \in \mathcal{H}] \right] - \mathbf{Pr}_{H \sim D_G^s} [H \in \mathcal{H}] \right| < \delta$$

3 Our Results

We first state our our main result in a simplified manner, for motivation and clarity, followed by the version which we actually prove:

Theorem 1. If a hereditary, inflatable graph property has a test making $q(\varepsilon)$ queries, regardless of the size of the input graph, then it has a strongly canonical test — specifically, a natural test — making $O(q(\varepsilon)^4)$ queries.

Theorem 1 (exact version). Let Π be a graph property has a test with queries involving at most $s(\varepsilon)$ distinct vertices, regardless of the size of the input graph, and let $s_1 = 12 \binom{31s}{2}$. If Π is both $(s_1, \frac{1}{6} \binom{s_1}{2}^{-1})$ -hereditary on the average and (s_1, s_1^{-1}) -inflatable on the average, then it has a strongly canonical test whose queried subgraph order is $s_1 = O(s(\varepsilon)^2)$.

A weak converse of Theorem 1 also holds:

Theorem 2. *If a graph property Π has a natural (not necessarily canonical) test with queries involving $s(\varepsilon)$ distinct vertices, then for every $\varepsilon' > \varepsilon$, Π is (s_h, ε') -hereditary on the average and (s_i, ε') -inflatable on the average, for $s_h = O(s \cdot \log(\frac{1}{\varepsilon' - \varepsilon}))$ and $s_i = O(s^2 \cdot (\varepsilon' - \varepsilon)^{-1} \log^2(\frac{1}{\varepsilon' - \varepsilon}))$ respectively.*

Let us now recall [15, proposition D.2], discussed in the introduction:

Proposition (corrected as per [16]). *Let Π be a hereditary graph property, with a natural test making $q(\varepsilon)$ queries. Then Π has a perfectly canonical test with queried subgraph order $O(q(\varepsilon))$.*

Originally, this proposition was stated requiring only that $q(\cdot)$ not depend on n , without requiring that the test be natural. When we combine the corrected, qualified version above with [Theorem 1], we obtain:

Corollary 1. *If a hereditary inflatable property has a test making $q(\varepsilon)$ queries, then it has a perfectly canonical test with queried subgraph order $\text{poly}(q(\varepsilon))$.*

The converse of this corollary, useful for proving lower bounds, is that if a hereditary inflatable property has no perfectly canonical test with queried subgraph order $\text{poly}(q(\varepsilon))$, then it has no test whatsoever whose number of queries is $q(\varepsilon)$ (natural or otherwise, with one-sided or two-sided error). For triangle-freeness specifically, combining [Corollary 1] with [2, Lemma 3.1], and the fact that the property of triangle-freeness is inflatable, one obtains an alternative, ‘generic’ proof of the following:

Corollary 2 (first proven as [5, Theorem 1]). *The query complexity of any ε -test — natural or otherwise, with one- or two-sided error — for the property of being triangle-free is at least $(c/\varepsilon)^{c \cdot \log(c/\varepsilon)}$, for some global constant c .*

As mentioned in the introduction, the proof in [5] uses a construction specific to the details of the $(c/\varepsilon)^{c \cdot \log(c/\varepsilon)}$ one-sided lower bound. The construction we will be utilizing in the proof of [Theorem 1] applies to any test for triangle-freeness, so a proposition similar to the above corollary would hold for any possible lower bound on the query complexity of testing triangle-freeness. It will similarly hold for the property of being free of any single non-bipartite graph which is not a blowup of a smaller graph.

Returning to [15, proposition D.2], while for hereditary inflatable properties we have established it with a power-of-four penalty on the number of queries, for properties with one-sided tests it can be shown to hold as stated:

Proposition 1. *If a hereditary inflatable property Π has a one-sided (not necessarily natural) test making $q(\varepsilon)$ queries, then Π has a perfectly canonical test with queried subgraph order at most $2q$.*

Finally, putting inflatability in the context of proximity-oblivious testing, the following partial characterization follows:

Proposition 2. *Let Π be an inflatable hereditary property. Π has a constant-query proximity-oblivious test if and only if there exists a constant s such that, for $n \geq s$, Π_n consists exactly of those graphs of order n , which are free of order- s graphs outside of Π_s .*

Proofs of [Theorem 2](#), [Proposition 1](#) and [Proposition 2](#) appear in the full version of this paper: [\[10\]](#).

4 Naturalizing Tests — Proof of Theorem 1

Let Π be a property meeting the conditions of [Theorem 1](#). As Π has a test with queries involving at most $s(\varepsilon)$ vertices (independently of n), by [\[15, Theorem 2\]](#) it has a canonical test, querying a uniformly sampled subgraph of order at most $9s$, in its entirety. As discussed in [Subsection 2.2](#), we may assume that the canonical test’s probability of error is at most $\frac{1}{36}$ rather than $\frac{1}{3}$, at the cost of increasing the queried subgraph order to $s_0 = 31s$.

One may think of the existence of such a canonical test as meaning that the membership of a graph in Π is essentially determined by its distribution of (induced) subgraphs of order s_0 . This being the case, let us consider a (canonical) ‘meta-test’ for Π , which (receiving n) estimates whether the subgraph distribution leads to acceptance: This meta-test is listed as [Algorithm 1](#).

Algorithm 1. A Meta-Test for Π

- 1: Uniformly query a subgraph G_{sample} of order $s_1 = 12 \binom{s_0}{2} = 12 \binom{31s(\varepsilon)}{2}$.
 - 2: If at least a $\frac{1}{6}$ -fraction of the order- s_0 subgraphs G' of G_{sample} are such that the (canonical) s_0 -test accepts G by sample G' , **accept**. Otherwise **reject**.
-

Lemma 4. [Algorithm 1](#) is a valid test for Π , with failure probability at most $\frac{1}{6}$.

Proof. Suppose the input graph G either satisfies Π or is ε -far from satisfying Π . Let G' be one of the $\binom{s_1}{s_0}$ order- s_0 subgraphs of G_{sample} . Let $X_{G'}$ be the indicator for the s_0 -test erring (that is, rejecting G in case G satisfies Π , or accepting G in case G is far from Π) by sample G' . Every order- s_0 subgraph of G_{sample} is in fact uniformly sampled from the input graph, thus $\mathbf{E}X_{G'}$ is the probability of the s_0 -test erring — at most $\frac{1}{36}$. The expected fraction of order- s subgraphs of G_{sample} by which the s_0 -test errs is therefore also at most $\frac{1}{36}$. Considering the meta-test’s behavior again, it only errs if at least a $\frac{1}{6}$ -fraction of the subgraphs of G_{sample} cause the s_0 -test to err. by Markov’s inequality the probability of this occurring is at most $\frac{1/36}{1/6} = \frac{1}{6}$. \square

Let us now modify [Algorithm 1](#) to reject samples which are themselves not in the property at order s_1 ; the result is listed as [Algorithm 2](#).

Lemma 5. [Algorithm 2](#) is a valid test for Π .

Algorithm 2. Modified Meta-Test for Π

- 1: Uniformly query a subgraph G_{sample} of order $s_1 = 12\binom{s_0}{2} = 12\binom{31s(\epsilon)}{2}$.
 - 2: If G_{sample} is not in Π , reject.
 - 3: If at least a $\frac{1}{6}$ -fraction of the order- s_0 subgraphs G' of G_{sample} are such that the s_0 -test accepts G by sample G' , then accept. Otherwise reject.
-

Proof. The additional check only increases the probability of rejection of any input graph, so it does not adversely affect the soundness of the modified test (that is, a graph ϵ -far from Π is still rejected by [Algorithm 2](#) with probability at least $\frac{5}{6} \geq \frac{2}{3}$).

Regarding completeness, we recall that Π is $(s_1, \frac{1}{6}\binom{s_1}{2})^{-1}$ -hereditary on the average. This implies that, for an input graph in Π , the average distance of subgraphs of order s_1 from Π is $\frac{1}{6}\binom{s_1}{2}^{-1}$; as each order- s_1 subgraph not in Π is at least $\binom{s_1}{2}^{-1}$ -far from Π , the fraction of order- s_1 subgraphs of G which aren't in Π is at most $\frac{1}{6}$. Regardless of these, at most a $\frac{1}{6}$ -fraction of the order- s_1 subgraphs of a satisfying graph cause [Algorithm 1](#) to reject. Union bounding over these two sets of subgraphs causing rejection we find that the probability of the modified meta-test rejecting a graph in Π is less than $2 \cdot \frac{1}{6} = \frac{1}{3}$. \square

Now, if [Algorithm 2](#) were somehow also natural, this would complete the proof of [Theorem 1](#), as the test otherwise meets the requirements. Since [Algorithm 2](#) is canonical, its naturality means being strongly canonical: Accepting the same set of sampled subgraphs for any input graph order, despite being given n as input. Interestingly enough, our modification has indeed made this the case:

Lemma 6. *Let H be a graph of order s_1 by which sample [Algorithm 2](#) accepts for at least some input graph order n . [Algorithm 2](#) cannot reject for any input graph order $n' \geq s_1$ by sample H .*

Proof. Assume on the contrary that [Algorithm 2](#) rejects by sample H for some $n' \geq s_1$. We first note that [Algorithm 2](#) does not reject by H at order n' on account of H not being in Π (as samples which aren't in Π are rejected at all input orders). We will show this to imply that the original test is unsound.

Let $\Pi'_{n'}$ denote the set of order- s_0 subgraphs by which sample the s_0 -test accepts an input graph G at order n' . Our assumption is that the probability of the s_0 -test accepting a subgraph of H is less than $\frac{1}{6}$, or in terms of the subgraph distribution, $\Pr_{H_s \sim \mathcal{D}_H^{s_0}}[\Pi'_{n'}] < \frac{1}{6}$.

Now, consider a random blowup H' of H to order n' . Π is $(s_1, \frac{1}{12}\binom{s_0}{2})^{-1}$ -inflatable on the average, and H is in Π , so $\mathbf{Ex}_{G'}[\text{dist}(H', \Pi)] < \frac{1}{12}\binom{s_0}{2}^{-1}$ and by Markov's inequality, $\Pr_{H'}[\text{dist}(H', \Pi) \geq \frac{1}{6}\binom{s_0}{2}^{-1}] < \frac{1}{2}$.

Also, let $\delta = \frac{1}{6}$. Since $s_1 \geq \frac{2}{\delta}\binom{s_0}{2}$, we may apply [Lemma 3](#) (substituting H and H' for G and G' , s_0 for s , s_1 for n) for the event of the s_0 -test accepting at order n' :

$$\begin{aligned} \mathbf{E} \mathbf{x}_{H'} \left[\mathbf{Pr}_{H_s \sim D_{H'}^{s_0}} [H_s \in \Pi'_{n'}] \right] &\leq \mathbf{Pr}_{H_s \sim D_H^{s_0}} [H_s \in \Pi'_{n'}] \\ &+ \left| \mathbf{E} \mathbf{x}_{H'} \left[\mathbf{Pr}_{H_s \sim D_{H'}^{s_0}} [H_s \in \Pi'_{n'}] \right] - \mathbf{Pr}_{H_s \sim D_H^{s_0}} [H_s \in \Pi'_{n'}] \right| \\ &< \mathbf{Pr}_{H_s \sim D_H^{s_0}} [H_s \in \Pi'_{n'}] + \delta < \frac{1}{6} + \frac{1}{6} = \frac{1}{3} \end{aligned}$$

and again by Markov’s inequality $\mathbf{Pr}_{H'} [\mathbf{Pr}_{H_s \sim D_{H'}^{s_0}} [H_s \in \Pi'_{n'}] \geq \frac{2}{3}] < \frac{1}{2}$. Combining these two facts, we conclude that with positive probability, H' is a graph which is both very close to Π and is accepted by the s_0 -test with probability at most $\frac{2}{3}$.

Now, let \widetilde{H}' be a graph in Π at distance at most $\frac{1}{6} \binom{s_0}{2}^{-1}$ from H' . By [Lemma 2](#), these two graphs’ order- s_0 subgraph distributions are $\frac{1}{6}$ -close, implying that $|\mathbf{Pr}_{H_s \sim D_{H'}^{s_0}} [H_s \in \Pi'_{n'}] - \mathbf{Pr}_{H_s \sim D_{\widetilde{H}'}^{s_0}} [H_s \in \Pi'_{n'}]| < \frac{1}{6}$.

We now use the triangle inequality to bound the probability of the s_0 -test accepting \widetilde{H}' :

$$\begin{aligned} \mathbf{Pr}_{H_s \sim D_{\widetilde{H}'}^{s_0}} [H_s \in \Pi'_{n'}] &\leq \mathbf{Pr}_{H_s \sim D_{H'}^{s_0}} [H_s \in \Pi'_{n'}] \\ &+ \left| \mathbf{Pr}_{H_s \sim D_{H'}^{s_0}} [H_s \in \Pi'_{n'}] - \mathbf{Pr}_{H_s \sim D_{\widetilde{H}'}^{s_0}} [H_s \in \Pi'_{n'}] \right| < \frac{2}{3} + \frac{1}{6} = \frac{5}{6} \end{aligned}$$

This contradicts the original test’s probability of error — it must accept \widetilde{H}' , a graph in Π , with probability at least $1 - \frac{1}{36} > \frac{5}{6}$. It cannot therefore be the case that [Algorithm 2](#) rejects H at order n' . □

Proof (of [Theorem 1](#)). Given a property Π satisfying the conditions, we have devised [Algorithm 2](#). This is a canonical test for Π , with queried subgraph order $s_1 = 12 \binom{31s}{2}$; by [Lemma 6](#), it accepts and rejects the same set of queried subgraphs for all graph orders $n \geq s_1$ — that is, it is a natural test. □

5 Open Questions

Naturalization without Canonization. Can non-canonical tests be made natural, without incurring the penalty for making them non-adaptive and canonical?

Testing a Large Graph by Testing Small Subgraphs. [\[16\]](#) poses the question of whether any test for a hereditary property can be replaced with merely ensuring that a random *small* induced subgraph has the property. We’ve shown that being hereditary and inflatable, or one-sided-testable, is a sufficient condition for this to hold. Are these conditions, or similar ones, also necessary?

The Benefit of Non-natural Testing. Some testable properties have a non-constant-factor gap in query complexity between their adaptive and non-adaptive tests. Is this also the case for natural testing? Can one find specific properties exhibiting a gap, or ‘non-contrived’ properties for which there is no gap (as in [14])?

A More Appropriate Notion of Inflatability. The definition of a blowup and of (perfect) inflatability is somewhat arbitrary; thus, being the empty graph is inflatable, but being the complete graph is not — since the clusters in a blowup are empty. Also, the property of being H -free, when H is a blowup of a smaller graph, is not inflatable. However, these properties are inflatable on the average (at least with an exceedingly high threshold s). Can one devise a more appropriate notion of inflatability, which covers such properties as well, while still allowing for a ‘naturalization’ similar to that in [Theorem 1]?

Acknowledgement. We wish to thank Oded Goldreich for insightful suggestions regarding the presentation of these results, and specifically for suggesting the concept of ‘strong canonicity’.

References

1. Alon, N.: Private communication (1999)
2. Alon, N.: Testing subgraphs in large graphs. *Rnd. Str. & Alg.* 21(3-4), 359–370 (2002)
3. Alon, N., Fischer, E., Krivelevich, M., Szegedy, M.: Efficient testing of large graphs. *Combinatorica* 20, 451–476 (2000)
4. Alon, N., Fischer, E., Newman, I., Shapira, A.: A combinatorial characterization of the testable graph properties. *SIAM J. Comp.* 39(1), 143–167 (2009)
5. Alon, N., Shapira, A.: A characterization of easily testable induced subgraphs. *Combinatorics, Probability and Computing* 15(6), 791–805 (2006)
6. Alon, N., Shapira, A.: A characterization of the (natural) graph properties testable with one-sided error. *SIAM J. Comp.* 37(6), 1703–1727 (2008)
7. Borgs, C., Chayes, J., Lovász, L., Sós, V.T., Szegedy, B., Vesztergombi, K.: Graph limits and parameter testing. In: 38th STOC, pp. 261–270. ACM Press, New York (2006)
8. Fischer, E.: The art of uninformed decisions: A primer to property testing. In: Paun, G., Rozenberg, G., Salomaa, A. (eds.) *Current Trends in Theoretical Computer Science*, vol. 1, pp. 229–264. World Scientific Publishing, Singapore (2004)
9. Fischer, E., Newman, I.: Testing versus estimation of graph properties. *SIAM J. Comp.* 37(2), 482–501 (2007)
10. Fischer, E., Rozenberg, E.: Inflatable graph properties and natural property tests, <http://www.cs.technion.ac.il/~eyalroz/publications/FR2011.pdf>
11. Fox, J.: A new proof of the graph removal lemma. *Ann. Math.* (to appear)
12. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *Journal of the ACM* 45(4), 653–750 (1998)
13. Goldreich, O., Krivelevich, M., Newman, I., Rozenberg, E.: Hierarchy theorems for property testing. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) *APPROX 2009*. LNCS, vol. 5687, pp. 504–519. Springer, Heidelberg (2009)

14. Goldreich, O., Ron, D.: Algorithmic aspects of property testing in the dense graphs model. In: Goldreich, O. (ed.) Property Testing. LNCS, vol. 6390, pp. 295–305. Springer, Heidelberg (2010)
15. Goldreich, O., Trevisan, L.: Three theorems regarding testing graph properties. *Rnd. Str. & Alg.* 23(1), 23–57 (2003)
16. Goldreich, O., Trevisan, L.: Errata for [15] (2005), <http://www.wisdom.weizmann.ac.il/~oded/PS/tt-rr.ps>
17. Yao, A.C.C.: Probabilistic computations: Toward a unified measure of complexity (extended abstract). In: 18th FOCS, pp. 222–227 (1977)

Fast Simulation of Large-Scale Growth Models

Tobias Friedrich¹ and Lionel Levine²

¹ Max-Planck-Institut für Informatik, 66123 Saarbrücken, Germany

² Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Abstract. We give an algorithm that computes the final state of certain growth models without computing all intermediate states. Our technique is based on a “least action principle” which characterizes the odometer function of the growth process. Starting from an approximation for the odometer, we successively correct under- and overestimates and provably arrive at the correct final state. The degree of speedup depends on the accuracy of the initial guess.

Determining the size of the boundary fluctuations in growth models like internal diffusion-limited aggregation (IDLA) is a long-standing open problem in statistical physics. As an application of our method, we calculate the size of fluctuations over two orders of magnitude beyond previous simulations.

1 Introduction

In this paper we study the *abelian stack model*, a type of growth process on graphs. Special cases include *internal diffusion limited aggregation* (IDLA) and *rotor-router aggregation*. We describe a method for computing the final state of the process, given an initial approximation. The more accurate the approximation, the faster the computation.

IDLA

Starting with N chips at the origin of the two-dimensional square grid \mathbb{Z}^2 , each chip in turn performs a simple random walk until reaching an unoccupied site. Introduced by Meakin and Deutch [25] and independently by Diaconis and Fulton [10], IDLA models physical phenomena such as solid melting around a heat source, electrochemical polishing, and fluid flow in a Hele-Shaw cell. Lawler, Bramson, and Griffeath [22] showed that as $N \rightarrow \infty$, the asymptotic shape of the resulting cluster of N occupied sites is a disk (and in higher dimensions, a Euclidean ball).

The boundary of an IDLA cluster is a natural model of a random propagating front (Figure 1, left). From this perspective, the most basic question one could ask is, what is the scale of the fluctuations around the limiting circular shape? Until recently this was a long-standing open problem in statistical physics. It is now known that the fluctuations in dimension 2 are of order at most $\log N$ [3, 19]; however, it is still open to show that the fluctuations are at least this large. We give numerical evidence that $\log N$ is in fact the correct order, and estimate the constant in front of the log.

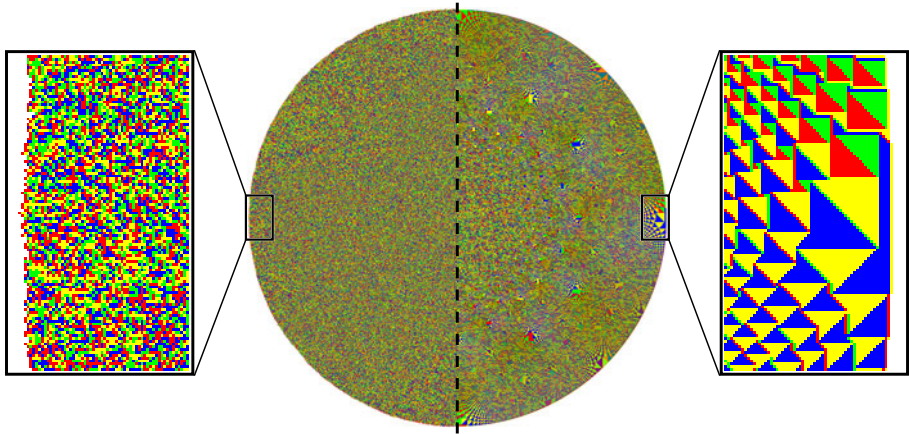


Fig. 1. IDLA cluster (left) and rotor-router cluster with counterclockwise rotor sequence (right) of $N = 10^6$ chips. Half of each circular cluster is shown. Each site is colored according to the final direction of the rotor on top of its stack (yellow=W, red=S, blue=E, green=N). Note that the boundary of the rotor-router cluster is much smoother than the boundary of the IDLA cluster. Larger rotor-router clusters of size up to $N = 10^{10}$ can be found at [1].

Rotor-router Aggregation

James Propp [21] proposed the following way of derandomizing IDLA. At each lattice site in \mathbb{Z}^2 is a *rotor* that can point North, East, South or West. Instead of stepping in a random direction, a chip rotates the rotor at its current location counterclockwise, and then steps in the direction of this rotor. Each of N chips starting at the origin walks in this manner until reaching an unoccupied site. Given the initial configuration of the rotors (which can be taken, for example, all North), the resulting growth process is entirely deterministic. Regardless of the initial rotors, the asymptotic shape is a disk (and in higher dimensions, a Euclidean ball) and the inner fluctuations are proved to be $O(\log N)$ [24]. The true fluctuations appear to grow even more slowly, and may even be bounded independent of N .

Rotor-router aggregation is remarkable in that it generates a nearly perfect disk in the square lattice without any reference to the Euclidean norm $(x^2 + y^2)^{1/2}$. Perhaps even more remarkable are the patterns formed by the final directions of the rotors (Figure 1, right).

Computing the Odometer Function

The central tool in our analysis of both models is the *odometer function*, which measures the number of chips emitted from each site. The odometer function determines the shape of the final occupied cluster via a nonlinear operator that we call the *stack Laplacian*. Our main technical contribution is that even for highly

non-deterministic models such as IDLA, one can achieve *fast exact calculation via intermediate approximation*. Approximating the two growth processes by an idealized model called the divisible sandpile, we can use the known asymptotic expansion of the potential kernel of random walk on \mathbb{Z}^2 to obtain an initial approximation of the odometer function. We present a method for carrying out subsequent local corrections to provably transform this approximation into the exact odometer function, and hence compute the shape of the occupied cluster. Our runtime strongly depends on the accuracy of the initial approximation.

Applications

Traditional step-by-step simulation of both aforementioned models in \mathbb{Z}^2 requires a runtime of order N^2 to compute the occupied cluster. Using our new algorithm, we are able to generate large clusters faster: Our observed runtimes are about $N \log N$ for the rotor-router model and about $N^{1.5}$ for IDLA. By generating many independent IDLA clusters, we estimate the order of fluctuations from circularity over two orders of magnitude beyond previous simulations. Our data strongly support the findings of [26] that the order of the maximum fluctuation for IDLA in \mathbb{Z}^2 is logarithmic in N . Two proofs of an upper bound $C \log N$ on the maximum fluctuation for IDLA in \mathbb{Z}^2 have recently been announced: see [2, 3, 15, 19]. While the implied constant C in these bounds is large, our simulations suggest that the maximum fluctuation is only about $0.528 \ln N$.

For rotor-router aggregation we achieve four orders of magnitude beyond previous simulations, which has enabled us to generate fine-scaled examples of the intricate patterns that form in the rotors on the tops of the stacks at the end of the aggregation process (Figure 1, right). These patterns remain poorly understood even on a heuristic level. We have used our algorithm to generate a four-color 10-gigapixel image [1] of the final rotors for $N = 10^{10}$ chips. This file is so large that we had to use a Google maps overlay to allow the user to zoom and scroll through the image. Indeed, the degree of speedup in our method was so dramatic that memory, rather than time, became the limiting factor.

Related Work

Unlike random walk, in a rotor-router walk each vertex serves its neighbors in a fixed order. The resulting walk, which is completely deterministic, nevertheless closely resembles a random walk in several respects [6–8, 11, 16, 18]. The rotor-router mechanism also leads to improvements in algorithmic applications. Examples include autonomous agents patrolling a territory [28], external mergesort [5], broadcasting information in networks [12, 13], and iterative load-balancing [17].

Abelian stacks (defined in the next section) are a way of indexing the steps of a walk by location and time rather than time alone. This fruitful idea goes back at least to Diaconis and Fulton [10], §4. Wilson [29] (see also [27]) used this stack-based view of random walk in his algorithm for sampling a random spanning tree of a directed graph. The final cycle-popping phase of our algorithm is directly inspired by Wilson's algorithm. Our serial algorithm for IDLA also draws on ideas from the parallel algorithm of Moore and Machta [26].

Abelian stacks are a special case of *abelian networks* [9], also called “abelian distributed processors.” In this viewpoint, each vertex is a finite automaton, or “processor.” The chips are called “messages.” When a processor receives a message, it can change internal state and also send one or more messages to neighboring processors according to its current internal state. We believe that it might be possible to extend our method to other types of abelian networks, such as the Bak-Tang-Wiesenfeld sandpile model [4]. Indeed, the initial inspiration for our work was the “least action principle” for sandpiles described in [14].

2 Formal Model

The underlying graph for the abelian stack model can be any finite or infinite directed graph $G = (V, E)$. Each edge $e \in E$ is oriented from its source vertex $\mathbf{s}(e)$ to its target vertex $\mathbf{t}(e)$. Self-loops (edges e such that $\mathbf{s}(e) = \mathbf{t}(e)$) and multiple edges (distinct edges e, e' such that $\mathbf{s}(e) = \mathbf{s}(e')$ and $\mathbf{t}(e) = \mathbf{t}(e')$) are permitted. We assume that G is *locally finite* – each vertex is incident to finitely many edges – and *strongly connected*: for any two vertices $x, y \in V$ there are directed paths from x to y and from y to x . At each vertex $x \in V$ is an infinite stack of *rotors* $(\rho_n(x))_{n \geq 0}$. Each rotor $\rho_n(x)$ is an edge of G emanating from x , that is, $\mathbf{s}(\rho_n(x)) = x$. We say that rotor $\rho_0(x)$ is “on top” of the stack.

A finite number of indistinguishable chips are dispersed on the vertices of G according to some prescribed initial configuration. For each vertex x , the first chip to visit x is absorbed there and never moves again. Each subsequent chip arriving at x first shifts the stack at x so that the new stack is $(\rho_{n+1}(x))_{n \geq 0}$. After shifting the stack, the chip moves from x to the other endpoint $y = \mathbf{t}(\rho_1(x))$ of the rotor now on top. We call this two-step procedure (shifting the stack and moving a chip) *firing* the site x . The effect of this rule is that the n -th time a chip is emitted from x , it travels along the edge $\rho_n(x)$.

We will generally assume that the stacks are *infinite*: for each edge e , infinitely many rotors $\rho_n(\mathbf{s}(e))$ are equal to e . If G is infinite, or if the total number of chips is at most the number of vertices, then this condition ensures that firing eventually stops, and all chips are absorbed.

We are interested in the set of *occupied sites*, that is, sites that absorb a chip. The *abelian property* [10, Theorem 4.1] asserts that this set does not depend on the order in which vertices are fired. This property plays a key role in our method; we discuss it further in §3.

If the rotors $\rho_n(x)$ are independent and identically distributed random edges e such that $\mathbf{s}(e) = x$, then we obtain IDLA. For instance, in the case $G = \mathbb{Z}^2$, we can take the rotors $\rho_n(x)$ to be independent with the uniform distribution on the set of 4 edges joining x to its nearest neighbors $x \pm \mathbf{e}_1, x \pm \mathbf{e}_2$. The special case of IDLA in which all chips start at a fixed vertex o is more commonly described as follows. Let $A_1 = \{o\}$, and for $N \geq 2$ define a random set A_N of N vertices of G according to the recursive rule

$$A_{N+1} = A_N \cup \{x_N\} \tag{1}$$

where x_N is the endpoint of a random walk started at o and stopped when it first visits a site not in A_N . These random walks describe one particular sequence in which the vertices can be fired, for the initial configuration of N chips at o . The first chip is absorbed at o , and subsequent chips are absorbed in turn at sites x_1, \dots, x_{N-1} . When firing stops, the set of occupied sites is A_N .

A second interesting case is deterministic: the sequence $\rho_n(x)$ is periodic in n , for every vertex x . For example, on \mathbb{Z}^2 , we could take the top rotor in each stack to point to the northward neighbor, the next to the eastward neighbor, and so on. This choice yields the model of rotor-router aggregation defined by Propp [21] and analyzed in [23, 24]. It is described by the growth rule (II), where x_N is the endpoint of a rotor-router walk started at the origin and stopped on first exiting A_N .

3 Least Action Principle

A rotor configuration on G is a function $r: V \rightarrow E$ such that $\mathbf{s}(r(v)) = v$ for all $v \in V$. A chip configuration on G is a function $\sigma: V \rightarrow \mathbb{Z}$ with finite support. Note we do not require $\sigma \geq 0$. If $\sigma(x) = m > 0$, we say there are m chips at vertex x ; if $\sigma(x) = -m < 0$, we say there is a hole of depth m at vertex x .

For an edge e and a nonnegative integer n , let

$$R_\rho(e, n) = \#\{1 \leq k \leq n \mid \rho_k(\mathbf{s}(e)) = e\}$$

be the number of times e occurs among the first n rotors in the stack at the vertex $\mathbf{s}(e)$ (excluding the top rotor $\rho_0(\mathbf{s}(e))$). When no ambiguity would result, we drop the subscript ρ .

Write \mathbb{N} for the set of nonnegative integers. Given a function $u: V \rightarrow \mathbb{N}$, we would like to describe the net effect on chips resulting from firing each vertex $x \in V$ a total of $u(x)$ times. In the course of these firings, each vertex x emits $u(x)$ chips, and receives $R_\rho(e, u(\mathbf{s}(e)))$ chips along each incoming edge e with $\mathbf{t}(e) = x$. This motivates the following definition.

Definition 1. The stack Laplacian of a function $u: V \rightarrow \mathbb{N}$ is the function $\Delta_\rho u: V \rightarrow \mathbb{Z}$ given by $\Delta_\rho u(x) = \sum_{\mathbf{t}(e)=x} R_\rho(e, u(\mathbf{s}(e))) - u(x)$, where the sum is over all edges e with target vertex $\mathbf{t}(e) = x$.

Given an initial chip configuration σ_0 , the configuration σ resulting from performing $u(x)$ firings at each site $x \in V$ is given by

$$\sigma = \sigma_0 + \Delta_\rho u. \tag{2}$$

The rotor configuration on the tops of the stacks after these firings is also easy to describe. We denote this configuration by $\text{Top}_\rho(u)$, and it is given by $\text{Top}_\rho(u)(x) = \rho_{u(x)}(x)$. Vertices x_1, \dots, x_m form a legal firing sequence for σ_0 if $\sigma_j(x_{j+1}) > 1$, $j = 0, \dots, m - 1$ where $\sigma_j = \sigma_0 + \Delta_\rho u_j$ and $u_j(x) = \#\{i \leq j : x_i = x\}$.

In words, the condition $\sigma_j(x_{j+1}) > 1$ says that after firing x_1, \dots, x_j , the vertex x_{j+1} has at least two chips. We require at least two because in our growth model, the first chip to visit each vertex gets absorbed.

The firing sequence is *complete* if no further legal firings are possible; that is, $\sigma_m(x) \leq 1$ for all $x \in V$. If x_1, \dots, x_m is a complete legal firing sequence for the chip configuration σ_0 , then we call the function $u := u_m$ the *odometer* of σ_0 . The odometer tells us how many times each site fires.

Abelian Property. [10, Theorem 4.1] Given an initial configuration σ_0 and stacks ρ , every complete legal firing sequence for σ_0 has the same odometer function u .

It follows that the final chip configuration $\sigma_m = \sigma_0 + \Delta_\rho u$ and the final rotor configuration $\text{Top}_\rho(u)$ do not depend on the choice of complete legal firing sequence.

Given a chip configuration σ_0 and rotor stacks $(\rho_k(x))_{k \geq 0}$, our goal is to compute the final chip configuration σ_m without performing individual firings one at a time. A fundamental observation is that by equation (2), it suffices to compute the odometer function u of σ_0 . Indeed, once we know that each site x fires $u(x)$ times, we can add up the number of chips x receives from each of its neighbors and subtract the $u(x)$ chips it emits to figure out the final number of chips at x . This arithmetic is accomplished by the term $\Delta_\rho u$ in equation (2). In practice, it is usually easy to compute $\Delta_\rho u$ given u , an issue we address in §4.

Our approach will be to start from an approximation of u and correct errors. In order to know when our algorithm is finished, the key mathematical point is to find a list of properties of u that characterize it uniquely. Our main result in this section, Theorem 1, gives such a list. As we now explain, the hypotheses of this theorem can all be guessed from certain necessary features of the final chip configuration σ_m and the final rotor configuration $\text{Top}_\rho(u)$. What is perhaps surprising is that these few properties suffice to characterize u .

Let x_1, \dots, x_m be a complete legal firing sequence for the chip configuration σ_0 . We start with the observation that since no further legal firings are possible,

- $\sigma_m(x) \leq 1$ for all $x \in V$.

Next, consider the set $A := \{x \in V : u(x) > 0\}$ of sites that fire. Since each site that fires must first absorb a chip, we have

- $\sigma_m(x) = 1$ for all $x \in A$.

Finally, observe that for any vertex $x \in A$, the rotor $r(x) = \text{Top}_\rho(u)(x)$ at the top of the stack at x is the edge traversed by the last chip fired from x . The last chip fired from a given finite subset A' of A must be to a vertex outside of A' , so A' must have a vertex whose top rotor points outside of A' .

- For any finite set $A' \subset A$, there exists $x \in A'$ with $\mathfrak{t}(r(x)) \notin A'$.

We can state this last condition more succinctly by saying that the rotor configuration $r = \text{Top}_\rho(u)$ is *acyclic* on A ; that is, the spanning subgraph $(V, r(A))$ has no directed cycles. Here $r(A) = \{r(x) \mid x \in A\}$.

Theorem 1. *Let G be a finite or infinite directed graph, ρ a collection of rotor stacks on G , and σ_0 a chip configuration on G . Fix $u_*: V \rightarrow \mathbb{N}$, and let $A_* = \text{supp}(u_*)$. Let $\sigma_* = \sigma_0 + \Delta_\rho u_*$, and suppose that*

- $\sigma_* \leq 1$;
- A_* is finite;
- $\sigma_*(x) = 1$ for all $x \in A_*$; and
- $\text{Top}_\rho(u_*)$ is acyclic on A_* .

Then the odometer function u is well-defined, and $u_ = u$.*

Note that to ensure that u is well-defined (i.e., that there exists a finite complete legal firing sequence) it is common to place some minimal assumptions on ρ and σ_0 . For example, if G is infinite and strongly connected, then it suffices to assume that the stacks ρ are infinitive. Theorem 1 does not explicitly make any assumptions of this kind; rather, if a function u_* exists satisfying the conditions listed, then u must be finite (and equal to u_*).

4 Algorithm: From Approximation to Exact Calculation

In this section we describe how to compute the odometer function u exactly, given as input an approximation u_1 . The running time depends on the accuracy of the approximation, but the correctness of the output does not. How to find a good approximation u_1 for N chips started at the origin in \mathbb{Z}^2 , can be found in the full version of the paper [15].

Recall that G may be finite or infinite, and we assume that G is strongly connected. We assume that the initial configuration σ_0 satisfies $\sigma_0(x) \geq 0$ for all x , and $\sum_x \sigma_0(x) < \infty$. If G is finite, we assume that $\sum_x \sigma_0(x)$ is at most the number of vertices of G (otherwise, some chips would never get absorbed). The only assumption on the approximation u_1 is that it is nonnegative with finite support. Finally, we assume that the rotor stacks are infinitive, which ensures that the growth process terminates after finitely many firings: that is, $\sum_{x \in V} u(x) < \infty$.

For $x \in V$, write $d_{out}(x) = \#\{e \in E \mid \mathfrak{s}(e) = x\}$ and $d_{in}(x) = \#\{e \in E \mid \mathfrak{t}(e) = x\}$ for the out-degree and in-degree of x , respectively. The odometer function u depends on the initial chip configuration σ_0 and on the rotor stacks $(\rho_k(x))_{k \geq 0}$. The latter are completely specified by the function $R(e, n)$ defined in §3. Note that for rotor-router aggregation, since the stacks are periodic, $R(e, n)$ has the simple explicit form $R(e, n) = \left\lfloor \frac{n + d_{out}(x) - j}{d_{out}(x)} \right\rfloor$ where j is the least positive integer such that $\rho_j(x) = e$. For IDLA, $R(e, n)$ is a random variable with the Binomial(n, p) distribution, where p is the transition probability associated to the edge e . In this section we take $R(e, n)$ as known. From

a computational standpoint, if the stacks are random, then determining $R(e, n)$ involves calls to a pseudorandom number generator.

Our algorithm consists of an approximation step followed by two error-correction steps: an annihilation step that corrects the chip locations, and a reverse cycle-popping step that corrects the rotors.

- (1) **Approximation.** Perform firings according to the approximate odometer, by computing the chip configuration $\sigma_1 = \sigma_0 + \Delta_\rho u_1$. Using Definition 1, this takes time $O(d_{in}(x) + 1)$ for each vertex x , for a total time of $O(\#E + \#V)$. This step is where the speedup occurs, because we are performing many firings at once: $\sum_x u_1(x)$ is typically much larger than $\#E + \#V$. Return σ_1 .
- (2) **Annihilation.** Start with $u_2 = u_1$ and $\sigma_2 = \sigma_1$. If $x \in V$ satisfies $\sigma_2(x) > 1$, then we call x a *hill*. If $\sigma_2(x) < 0$, or if $\sigma_2(x) = 0$ and $u_2(x) > 0$, then we call x a *hole*. For each $x \in \mathbb{Z}^2$,
 - (a) If x is a hill, fire it by incrementing $u_2(x)$ by 1 and then moving one chip from x to $\mathfrak{t}(\text{Top}(u_2)(x))$.
 - (b) If x is a hole, unfire it by moving one chip from $\mathfrak{t}(\text{Top}(u_2)(x))$ to x and then decrementing $u_2(x)$ by one.

A hill can disappear in one of two ways: by reaching an unoccupied site on the boundary, or by reaching a hole and canceling it out. When there are no more hills and holes, return u_2 .
- (3) **Reverse cycle-popping.** Start with $u_3 = u_2$ and $A_3 = \{x \in V : u_3(x) > 0\}$. If $\text{Top}(u_3)$ is not acyclic on A_3 , then pick a cycle and unfire each of its vertices once. This may create additional cycles. Update A_3 and repeat until $\text{Top}(u_3)$ is acyclic on A_3 . Output u_3 .

5 Experimental Results

We implemented our algorithm for both growth models in \mathbb{Z}^2 . The source code is available from [1]. In this section we discuss some of our results. More experimental findings can be found in the full version [15].

Traditional step-by-step simulation requires quadratic time to compute the occupied cluster [22, 26]. We found experimentally that our algorithm ran in significantly shorter time: about $N \log N$ for the rotor-router model, and about $N^{1.5}$ for IDLA.

5.1 Rotor-Router Aggregation

In the classic rotor-router model, the rotor stack is the cyclic sequence of the four cardinal directions in counterclockwise order. The absolute error in our odometer approximation

$$\|u_1 - u\|_1 = \sum_x |u_1(x) - u(x)|$$

appears to scale linearly with N . This quantity is certainly a lower bound for the running time of our algorithm. The measured runtimes indicate close-to-linear runtime behavior, which suggests that our multiscale approach to canceling out hills and holes is relatively efficient.

The asymptotic shape of rotor-router aggregation is a disk [23, 24]. To measure how close A_N is to a disk, we define the *inradius* and *outradius* of a set $A \subset \mathbb{Z}^2$ by $r_{in}(A) = \min\{|x|: x \notin A\}$ and $r_{out}(A) = \max\{|x|: x \in A\}$, respectively. We then define

$$\text{diff}(N) = r_{out}(A_N) - r_{in}(A_N).$$

A natural question is whether this difference is bounded independent of N . We certainly expect it to increase much more slowly than the order $\log N$ observed for IDLA.

Kleber [21] calculated that $\text{diff}(3 \cdot 10^6) \approx 1.6106$. We can now extend the measurement of $\text{diff}(N)$ up to $N = 2^{36} \approx 6.8 \cdot 10^{10}$ and observe in this case for example $\text{diff}(2^{36}) \approx 1.587$. Our algorithm runs in less than four hours for this value of N ; by comparison, a step-by-step simulation of this size would take about 23000 years on a computer with one billion operations per second. In our implementation, the limiting factor is memory rather than time.

Up to dihedral symmetry, there are three different balanced period-4 rotor sequences for \mathbb{Z}^2 : WENS, WNSE and WNES. The notation WENS means that the first four rotors in each stack point respectively west, east, north and south.

Figure 2a shows the radius difference $\text{diff}(N)$ for various N for the three different rotor sequences. As these values are rather noisy, we have also calculated and plotted the averages

$$\overline{\text{diff}}(N) := \frac{1}{|I(N)|} \sum_{N' \in I(N)} \text{diff}(N') \quad (3)$$

with $I(N) = [\frac{N}{2}, \frac{3N}{2}]$ for $N \leq 10^6$, and $I(N) = [N - 5 \cdot 10^5, N + 5 \cdot 10^5]$ for $N > 10^6$.

Note that in Figure 2a, the radius difference $\overline{\text{diff}}(N)$ grows extremely slowly in N . In particular, it appears to be sublogarithmic.

We observe a systematic difference in behavior for the three different rotor sequences. The observed radius differences are lowest for WNSE, intermediate for WNES, and highest for WENS. For example,

$$\overline{\text{diff}}(10^8) \approx \begin{cases} 1.034 & \text{for WNSE,} \\ 1.623 & \text{for WNES,} \\ 1.837 & \text{for WENS.} \end{cases}$$

5.2 Internal Diffusion Limited Aggregation (IDLA)

In IDLA, the rotor directions $\rho_k(x)$ for $x \in \mathbb{Z}^2$ and $k \in \mathbb{Z}$ are chosen independently and uniformly at random from among the four cardinal directions. In the course of firing and unfiring during steps 2 and 3 of our algorithm, the same

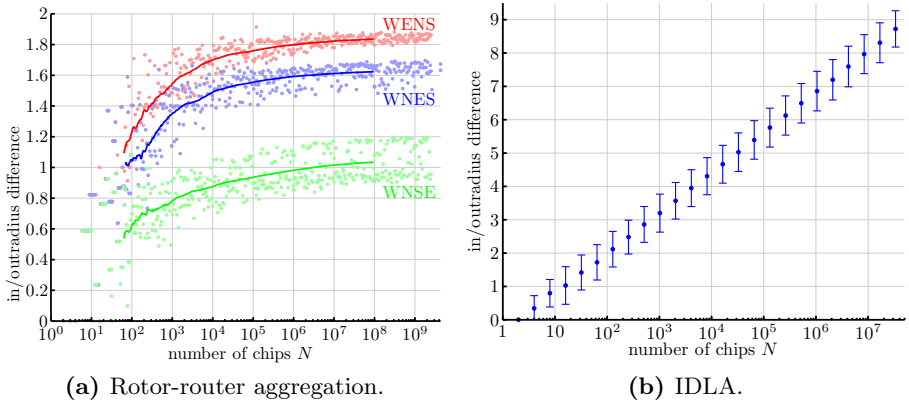


Fig. 2. Difference between the inradius and outradius for the rotor-router aggregate (left) and IDLA (right). In the left figure, the single dots are individual values of $\text{diff}(N)$ while the darker curves show the averages $\overline{\text{diff}(N)}$ as defined in equation (3). The right figure shows the average values and standard deviations.

rotor $\rho_k(x)$ may be requested several times. Therefore, we need to be able to generate the same pseudorandom value for $\rho_k(x)$ each time it is used. Generating and storing all rotors $\rho_k(x)$ for all x and all $1 \leq k \leq u_1(x)$ is out of the question, however, since it would cost $\Omega(N^2)$ time and space.

Moore and Machta [26] encountered the same issue in developing a fast parallel algorithm for IDLA. Rather than store all of the random choices, they chose to store only certain seed values for the random number generator and generate random walk steps online as needed. In the full version of the paper [15] we describe how to adapt this idea to our setting for fast serial computation of IDLA.

The results of our large-scale simulations of IDLA are summarized in Figure 2b, extending the experiments of Moore and Machta [26] ($N \leq 10^{5.25}$ with 100 trials) to over 10^6 trials for $N \leq 2^{16}$ and over 300 trials for $N \leq 2^{25} \approx 10^{7.5}$. The observed runtime of our algorithm for IDLA is about $N^{1.5}$; in contrast, building an IDLA cluster of size N by serial simulation of N random walks takes expected time order N^2 (cf. [26, Fig. 3]).

The expected value of the difference $\text{diff}(N)$ between outradius and inradius grows logarithmically in N : The data fits to $\mathbb{E} \text{diff}(N) = 0.528 \ln(N) - 0.457$ with a coefficient of determination of $R^2 = 0.99994$. Error bars in figure Figure 2b show standard deviations of the random variable $\text{diff}(N)$. Note that the size of the standard deviation is approximately constant: it does not grow with N . This finding is consistent with the connection with Gaussian free field revealed in [20]: indeed, the maximum of the discrete two-dimensional Gaussian free field over the boundary of a disk of area N has mean of order $\log N$ and variance of order 1.

Acknowledgments. We thank James Propp for many enlightening conversations and for comments on several drafts of this article. We thank David Wilson suggesting the AES random number generator and for tips about how to use it effectively.

References

- [1] <http://rotor-router.mpi-inf.mpg.de>
- [2] Asselah, A., Gaudillère, A.: From logarithmic to subdiffusive polynomial fluctuations for internal DLA and related growth models, arXiv:1009.2838 (2010)
- [3] Asselah, A., Gaudillère, A.: Sub-logarithmic fluctuations for internal DLA, arXiv:1011.4592 (2010)
- [4] Bak, P., Tang, C., Wiesenfeld, K.: Self-organized criticality: an explanation of the $1/f$ noise. *Phys. Rev. Lett.* 59(4), 381–384 (1987)
- [5] Barve, R.D., Grove, E.F., Vitter, J.S.: Simple randomized mergesort on parallel disks. *Parallel Computing* 23(4-5), 601–631 (1997)
- [6] Cooper, J., Spencer, J.: Simulating a random walk with constant error. *Combinatorics, Probability & Computing* 15, 815–822 (2006)
- [7] Cooper, J., Doerr, B., Spencer, J., Tardos, G.: Deterministic random walks on the integers. *European Journal of Combinatorics* 28, 2072–2090 (2007)
- [8] Cooper, J., Doerr, B., Friedrich, T., Spencer, J.: Deterministic random walks on regular trees. *Random Structures and Algorithms* 37(3), 353–366 (2010)
- [9] Dhar, D.: Theoretical studies of self-organized criticality. *Physica A* 369, 29–70 (2006)
- [10] Diaconis, P., Fulton, W.: A growth model, a game, an algebra, Lagrange inversion, and characteristic classes. *Rend. Sem. Mat. Univ. Politec. Torino* 49(1), 95–119 (1991)
- [11] Doerr, B., Friedrich, T.: Deterministic random walks on the two-dimensional grid. *Combinatorics, Probability & Computing* 18(1-2), 123–144 (2009)
- [12] Doerr, B., Friedrich, T., Sauerwald, T.: Quasirandom rumor spreading. In: 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2008), pp. 773–781 (2008)
- [13] Doerr, B., Friedrich, T., Sauerwald, T.: Quasirandom rumor spreading: Expanders, push vs. pull, and robustness. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikolettseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555, pp. 366–377. Springer, Heidelberg (2009)
- [14] Fey, A., Levine, L., Peres, Y.: Growth rates and explosions in sandpiles. *J. Stat. Phys.* 138, 143–159 (2010)
- [15] Friedrich, T., Levine, L.: Fast simulation of large-scale growth models, arXiv:1006.1003 (2011)
- [16] Friedrich, T., Sauerwald, T.: The cover time of deterministic random walks. *Electr. J. Comb.* 17(1), 1–7 (2010)
- [17] Friedrich, T., Gairing, M., Sauerwald, T.: Quasirandom load balancing. In: 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2010), pp. 1620–1629 (2010)
- [18] Holroyd, A.E., Propp, J.G.: Rotor walks and Markov chains. *Algorithmic Probability and Combinatorics* 520, 105–126 (2010)
- [19] Jerison, D., Levine, L., Sheffield, S.: Logarithmic fluctuations for internal DLA, arXiv:1010.2483 (2010)

- [20] Jerison, D., Levine, L., Sheffield, S.: Internal DLA and the Gaussian free field, arXiv:1101.0596 (2011)
- [21] Kleber, M.: Goldbug variations. *Math. Intelligencer* 27(1), 55–63 (2005)
- [22] Lawler, G.F., Bramson, M., Griffeath, D.: Internal diffusion limited aggregation. *Ann. Probab.* 20(4), 2117–2140 (1992)
- [23] Levine, L., Peres, Y.: Spherical asymptotics for the rotor-router model in \mathbb{Z}^d . *Indiana Univ. Math. J.* 57(1), 431–450 (2008)
- [24] Levine, L., Peres, Y.: Strong spherical asymptotics for rotor-router aggregation and the divisible sandpile. *Potential Anal.* 30, 1–27 (2009)
- [25] Meakin, P., Deutch, J.M.: The formation of surfaces by diffusion-limited annihilation. *J. Chem. Phys.* 85 (1986)
- [26] Moore, C., Machta, J.: Internal diffusion-limited aggregation: parallel algorithms and complexity. *J. Stat. Phys.* 99(3-4), 661–690 (2000)
- [27] Propp, J.G., Wilson, D.B.: How to get a perfectly random sample from a generic Markov chain and generate a random spanning tree of a directed graph. *J. Algorithms* 27, 170–217 (1998)
- [28] Wagner, I.A., Lindenbaum, M., Bruckstein, A.M.: Smell as a computational resource – a lesson we can learn from the ant. In: 4th Israel Symposium on Theory of Computing and Systems (ISTCS 1996), pp. 219–230 (1996)
- [29] Wilson, D.B.: Generating random spanning trees more quickly than the cover time. In: 28th Annual ACM Symposium on the Theory of Computing (STOC 1996), pp. 296–303 (1996)

Improved Inapproximability Results for Counting Independent Sets in the Hard-Core Model

Andreas Galanis¹, Qi Ge², Daniel Štefankovič², Eric Vigoda¹, and Linji Yang¹

¹ School of Computer Science, Georgia Institute of Technology, Atlanta GA 30332
{agalanis,vigoda,ljyang}@cc.gatech.edu

² Department of Computer Science, University of Rochester, Rochester, NY 14627
{qge,stefanko}@cs.rochester.edu

Abstract. We study the computational complexity of approximately counting the number of independent sets of a graph with maximum degree Δ . More generally, for an input graph $G = (V, E)$ and an activity $\lambda > 0$, we are interested in the quantity $Z_G(\lambda)$ defined as the sum over independent sets I weighted as $w(I) = \lambda^{|I|}$. In statistical physics, $Z_G(\lambda)$ is the partition function for the hard-core model, which is an idealized model of a gas where the particles have non-negligible size. Recently, an interesting phase transition was shown to occur for the complexity of approximating the partition function. Weitz showed an FPAS for the partition function for any graph of maximum degree Δ when Δ is constant and $\lambda < \lambda_c(\mathbb{T}_\Delta) := (\Delta - 1)^{\Delta-1}/(\Delta - 2)^\Delta$. The quantity $\lambda_c(\mathbb{T}_\Delta)$ is the critical point for the so-called uniqueness threshold on the infinite, regular tree of degree Δ . On the other side, Sly proved that there does not exist efficient (randomized) approximation algorithms for $\lambda_c(\mathbb{T}_\Delta) < \lambda < \lambda_c(\mathbb{T}_\Delta) + \varepsilon(\Delta)$, unless $\text{NP}=\text{RP}$, for some function $\varepsilon(\Delta) > 0$. We remove the upper bound in the assumptions of Sly's result for $\Delta \neq 4, 5$, that is, we show that there does not exist efficient randomized approximation algorithms for all $\lambda > \lambda_c(\mathbb{T}_\Delta)$ for $\Delta = 3$ and $\Delta \geq 6$. Sly's inapproximability result uses a clever reduction, combined with a second-moment analysis of Mossel, Weitz and Wormald which prove torpid mixing of the Glauber dynamics for sampling from the associated Gibbs distribution on almost every regular graph of degree Δ for the same range of λ as in Sly's result. We extend Sly's result by improving upon the technical work of Mossel et al., via a more detailed analysis of independent sets in random regular graphs.

1 Introduction

For a graph $G = (V, E)$ and activity $\lambda > 0$, the hard-core model is defined on the set $\mathcal{I}(G)$ of independent sets of G where set $I \in \mathcal{I}(G)$ has weight $w(I) := \lambda^{|I|}$. The so-called partition function for the model is defined as:

¹ Research supported in part by NSF grants CCF-0830298 and CCF-0910584.

² Research supported in part by NSF grant CCF-0910415.

$$Z_G(\lambda) := \sum_{I \in \mathcal{I}(G)} w(I) = \sum_{I \in \mathcal{I}(G)} \lambda^{|I|}.$$

The Gibbs distribution μ is over the set $\mathcal{I}(G)$ where $\mu(I) = w(I)/Z_G(\lambda)$. The case $\lambda = 1$ is especially interesting from a combinatorial perspective, since the partition function is the number of independent sets in G and the Gibbs distribution is uniformly distributed over the set of independent sets.

The hard-core model has received considerable attention in several fields. In statistical physics, it is studied as an idealized model of a gas where the gas particles have non-negligible size so neighboring sites cannot simultaneously be occupied [41]. The activity λ corresponds to the fugacity of the gas. The model also arose in operations research in the study of communication networks [7].

We study the computational complexity of approximating the partition function. Valiant [15] proved that exactly computing the number of independent sets of an input graph $G = (V, E)$ is #P-complete. Greenhill [5] proved that even when the input is restricted to graphs with maximum degree 3, it is still #P-complete. Hence, our focus is on approximating the partition function.

Weitz [16] gave an FPAS (fully polynomial-time approximation scheme) for the partition function of graphs with maximum degree Δ when Δ is constant and $\lambda < \lambda_c(\mathbb{T}_\Delta) := (\Delta - 1)^{\Delta-1}/(\Delta - 2)^\Delta$. The activity $\lambda_c(\mathbb{T}_\Delta)$ is the critical activity for the threshold of uniqueness/non-uniqueness of the infinite-volume Gibbs measures on the infinite Δ -regular tree [7]. Recently, Sly [11] proved that, unless $NP = RP$, for every $\Delta \geq 3$, there exists a function $\varepsilon(\Delta) > 0$ such that for graphs with maximum degree Δ there does not exist an FPRAS (fully-polynomial time randomized approximation scheme) for the partition function at activity λ satisfying:

$$\lambda_c(\mathbb{T}_\Delta) < \lambda < \lambda_c(\mathbb{T}_\Delta) + \varepsilon(\Delta). \tag{*}$$

It was conjectured in Sly [11] and Mossel et al. [10] that the inapproximability result holds for all $\lambda > \lambda_c(\mathbb{T}_\Delta)$. We almost resolve this conjecture, that is we prove the conjecture for all Δ with the exception of $\Delta \in \{4, 5\}$.

Theorem 1. *Unless $NP=RP$, there does not exist an FPRAS for the partition function of the hard-core model for graphs of maximum degree at most Δ at activity λ when:*

- $\Delta = 3$ and $\lambda > \lambda_c(\mathbb{T}_3) = 4$; or
- $\Delta \geq 6$ and $\lambda > \lambda_c(\mathbb{T}_\Delta)$; or
- $\Delta = 4$ and $\lambda \in (\lambda_c(\mathbb{T}_4) = 1.6875, 2.01538] \cup (4, +\infty)$; or
- $\Delta = 5$ and $\lambda \in (\lambda_c(\mathbb{T}_5) = 256/243, 1.45641] \cup (1.6875, 2.01538] \cup (4, +\infty)$.

Sly’s work utilizes earlier work of Mossel et al. [10] which studied the Glauber dynamics. The Glauber dynamics is a simple Markov chain (X_t) that is used to sample from the Gibbs distribution (and hence to approximate the partition function via now standard techniques, see [6,12]). For an input graph $G = (V, E)$ and activity $\lambda > 0$, the state space of the chain is $\mathcal{I}(G)$. From a state $X_t \in \mathcal{I}(G)$, the transitions $X_t \rightarrow X_{t+1}$ are defined by the following stochastic process:

- Choose a vertex v uniformly at random from V .
- Let

$$X' = \begin{cases} X_t \cup \{v\} & \text{with probability } \lambda/(1 + \lambda) \\ X_t \setminus \{v\} & \text{with probability } 1/(1 + \lambda). \end{cases}$$

- If $X' \in \mathcal{I}(G)$, then set $X_{t+1} = X'$, otherwise set $X_{t+1} = X_t$.

It is straightforward to verify that the Glauber dynamics is ergodic, and the unique stationary distribution is the Gibbs distribution. The mixing time T_{mix} is the minimum number of steps T from the worst initial state X_0 , so that the distribution of X_T is within (total) variation distance $\leq 1/4$ of the stationary distribution. The chain is said to be *rapidly mixing* if the mixing time is polynomial in $n = |V|$, and it is said to be *torpidly mixing* if the mixing time is exponential in n (for the purposes of this paper, that means $T_{\text{mix}} = \exp(\Omega(n))$). We refer the reader to Levin et al. [8] for a more thorough introduction to the Glauber dynamics.

Mossel et al. [10] proved that the Glauber dynamics is torpidly mixing, for all $\Delta \geq 3$, for graphs with maximum degree Δ when λ satisfies (2). This improved upon earlier work of Dyer et al. [2] which held for larger λ , but not down to the critical activity $\lambda_c(\mathbb{T}_\Delta)$. The torpid mixing result of Mossel et al. [10] follows immediately (via a conductance argument) from their main result that for a random Δ -regular bipartite graph, for λ satisfying (2), an independent set drawn from the Gibbs distribution is “unbalanced” with high probability.

The proof of Mossel et al. [10] is a technically involved second moment calculation that Sly [11] calls a “technical tour de force”. Our main contribution is to improve upon Mossel et al.’s result, most notably, extending it to all $\lambda > \lambda_c(\mathbb{T}_\Delta)$ for $\Delta = 3$. Our improved analysis comes from using a slightly different parameterization of the second moment of the partition function, which brings in symmetry, and allows for simpler proofs.

To formally state our results for independent sets of random regular graphs, we need to partition the set of independent sets as follows. For a bipartite graph $G = (V_1 \cup V_2, E)$ where $|V_1| = |V_2| = n$, for $\delta > 0$, for $i \in \{1, 2\}$, let $\mathcal{I}_i^\delta = \{I \in \mathcal{I}(G) : |I \cap V_i| > |I \cap V_{3-i}| + \delta n\}$ denote the independent sets that are unbalanced and “biased” towards V_i . Let $\mathcal{I}_B^\delta = \{I \in \mathcal{I}(G) : |I \cap V_i| \leq |I \cap V_{3-i}| + \delta n\}$ denote the set of nearly balanced independent sets.

Let $\mathcal{G}(n, \Delta)$ denote the probability distribution over bipartite graphs with $n + n$ vertices formed by taking the union of Δ random perfect matchings. Strictly speaking, this distribution is over multi-graphs. However, for independent sets the multiplicity of an edge does not matter so we can view $\mathcal{G}(n, \Delta)$ as a distribution over simple graphs with maximum degree Δ . Moreover, since our results hold asymptotically almost surely (a.a.s.) over $\mathcal{G}(n, \Delta)$, as noted in [10, Section 2.1], by standard arguments (see the note after the proof of [9, Theorem 4]), our results also hold a.a.s. for the uniform distribution over bipartite Δ -regular graphs.

Theorem 2. *For all $\Delta \geq 3$ there exists $\varepsilon(\Delta) > 0$, for any λ where $\lambda_c(\mathbb{T}_\Delta) < \lambda < \lambda_c(\mathbb{T}_\Delta) + \varepsilon(\Delta)$, there exist $a > 1$ and $\delta > 0$ such that, asymptotically almost surely, for a graph G chosen from $\mathcal{G}(n, \Delta)$, the Gibbs distribution μ satisfies:*

$$\mu(\mathcal{I}_B^\delta) \leq a^{-n} \min\{\mu(\mathcal{I}_1^\delta), \mu(\mathcal{I}_2^\delta)\}. \tag{1}$$

Therefore, the Glauber dynamics is torpidly mixing. The function $\varepsilon(\Delta)$ satisfies: for $\Delta = 4$, $\varepsilon(4) \geq .327887$; for $\Delta = 5$, $\varepsilon(5) \geq .402912$; for $\Delta \geq 6$, $\varepsilon(\Delta) \geq \lambda_c(\mathbb{T}_\Delta) - \lambda_c(\mathbb{T}_{\Delta+1})$; and for $\Delta = 3$, $\varepsilon(3) = \infty$.

This proves Conjecture 2.4 of [10] for the case $\Delta = 3$ and extends the results of Mossel et al. for $\Delta \geq 4$.

In the following section we look at the first and second moments of the partition function. We then state two technical lemmas (Lemma 3 and Lemma 4) from which Theorems 1, 2 easily follow using work of Sly [11] and Mossel et al. [10]. In Section 3 we prove the technical lemmas. Some of our proofs use Mathematica to prove inequalities involving rational functions, this is discussed in Section 2.4

2 Technical Overview

2.1 Phase Transition Revisited

Recall, for the infinite Δ -regular tree \mathbb{T}_Δ , Kelly [7] showed that there is a phase transition at $\lambda_c(\mathbb{T}_\Delta) = (\Delta - 1)^{\Delta-1}/(\Delta - 2)^\Delta$. Formally, this phase transition can be defined in the following manner. Let T_ℓ denote the complete tree of degree Δ and containing ℓ levels. Let p_ℓ denote the marginal probability that the root is occupied in the Gibbs distribution on T_ℓ . Note, for even ℓ the root is in the maximum independent set, whereas for odd ℓ the root is not in the maximum independent set. Our interest is in comparing the marginal probability for the root in even versus odd sized trees. Hence, let $p^+ = \lim_{\ell \rightarrow \infty} p_{2\ell}$ and $p^- = \lim_{\ell \rightarrow \infty} p_{2\ell+1}$. One can prove these limits exist by analyzing appropriate recurrences. The phase transition on the tree \mathbb{T}_Δ captures whether p^+ equals (or not) p^- . For all $\lambda \leq \lambda_c(\mathbb{T}_\Delta)$, we have $p^+ = p^-$, and let $p^* := p^+ = p^-$. On the other side, for all $\lambda > \lambda_c(\mathbb{T}_\Delta)$, we have $p^+ > p^-$. Mossel et al. [10] exhibited the critical role these quantities p^+ and p^- play in the analysis of the Gibbs distribution on random graphs $\mathcal{G}(n, \Delta)$.

2.2 First Moment of the Partition Function

Proceeding as in [10], roughly speaking, to prove Theorem 2 we need to prove that there exist graphs G whose partition function is close to the expected value (where the expectation is over a random G). To do that we use the second moment method. We first investigate the first moment of the partition function.

For $\alpha, \beta > 0$, let $\mathcal{I}_G^{\alpha, \beta} = \{I \in \mathcal{I}_G \mid |I \cap V_1| = \alpha n, |I \cap V_2| = \beta n\}$, that is, α is the fraction of the vertices in V_1 that are in the independent set and β is the fraction of the vertices in V_2 that are in the independent set.

Let $Z_G^{\alpha,\beta} = \sum_{I \in \mathcal{I}_G^{\alpha,\beta}} \lambda^{(\alpha+\beta)n}$. The first moment of $Z_G^{\alpha,\beta}$ is

$$\mathbb{E}_G[Z_G^{\alpha,\beta}] = \lambda^{(\alpha+\beta)n} \binom{n}{\alpha n} \binom{n}{\beta n} \left(\frac{\binom{(1-\beta)n}{\alpha n}}{\binom{n}{\alpha n}} \right)^\Delta \approx \exp(n\Phi_1(\alpha, \beta)),$$

where $\Phi_1(\alpha, \beta) = (\alpha + \beta) \ln(\lambda) + H(\alpha) + H(\beta) + \Delta(1 - \beta)H(\frac{\alpha}{1-\beta}) - \Delta H(\alpha)$, and $H(x) = -x \ln x - (1 - x) \ln(1 - x)$ is the entropy function.

For every $\Delta \geq 2$, we define the region $\mathcal{T}_\Delta = \{(\alpha, \beta) \mid \alpha, \beta > 0 \text{ and } \alpha + \beta + \Delta(\Delta - 2)\alpha\beta \leq 1\}$. The following lemma shows that the local maxima of Φ_1 lie in \mathcal{T}_Δ . Note that for $\Delta \geq 2$, we have $\mathcal{T}_{\Delta+1} \subset \mathcal{T}_\Delta$. Hence, the local maxima for all $\Delta \geq 2$ lie in \mathcal{T}_2 . The first moment was analyzed in the work of Dyer et al. [2]. We use the following lemma from Mossel et al. [10] that relates the properties of the first moment to p^*, p^+ and p^- .

Lemma 1 (Lemma 3.2 and Proposition 4.1 of Mossel et al. [10]). *The following holds:*

1. the stationary point (α, β) of Φ_1 over \mathcal{T}_2 is the solution to $\beta = \phi(\alpha)$ and $\alpha = \phi(\beta)$, where

$$\phi(x) = (1 - x) \left(1 - \left(\frac{x}{\lambda(1 - x)} \right)^{1/\Delta} \right), \tag{2}$$

and the solutions are exactly (p^+, p^-) , (p^-, p^+) , and (p^*, p^*) when $\lambda > \lambda_c(\mathbb{T}_\Delta)$, and the unique solution is (p^*, p^*) when $\lambda \leq \lambda_c(\mathbb{T}_\Delta)$;

2. when $\lambda \leq \lambda_c(\mathbb{T}_\Delta)$, (p^*, p^*) is the unique maximum of Φ_1 over \mathcal{T}_2 , and when $\lambda > \lambda_c(\mathbb{T}_\Delta)$, (p^+, p^-) and (p^-, p^+) are the maxima of Φ_1 over \mathcal{T}_2 , and (p^*, p^*) is not a local maximum;
3. all local maxima of Φ_1 satisfy $\alpha + \beta + \Delta(\Delta - 2)\alpha\beta \leq 1$;
4. p^+, p^-, p^* satisfy $p^- < p^* < p^+$ and when $\lambda \rightarrow \lambda_c(\mathbb{T}_\Delta)$ from above, we have $p^*, p^-, p^+ \rightarrow 1/\Delta$.

2.3 Second Moment of the Partition Function

The second moment of $Z_G^{\alpha,\beta}$ is $\mathbb{E}_G[(Z_G^{\alpha,\beta})^2] \approx \exp(n \cdot \max_{\gamma, \delta, \varepsilon} \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon))$, where

$$\begin{aligned} \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon) = & 2(\alpha + \beta) \ln(\lambda) + H(\alpha) + H_1(\gamma, \alpha) + H_1(\alpha - \gamma, 1 - \alpha) \\ & + H(\beta) + H_1(\delta, \beta) + H_1(\beta - \delta, 1 - \beta) + \Delta \left(H_1(\gamma, 1 - 2\beta + \delta) - H(\gamma) \right. \\ & + H_1(\varepsilon, 1 - 2\beta + \delta - \gamma) + H_1(\alpha - \gamma - \varepsilon, \beta - \delta) - H_1(\alpha - \gamma, 1 - \gamma) \\ & \left. + H_1(\alpha - \gamma, 1 - \beta - \gamma - \varepsilon) - H_1(\alpha - \gamma, 1 - \alpha) \right), \end{aligned} \tag{3}$$

where $H(x) = -x \ln(x) - (1 - x) \ln(1 - x)$ and $H_1(x, y) = -x(\ln(x) - \ln(y)) + (x - y)(\ln(y - x) - \ln(y))$.

To make Φ_2 well defined, the variables have to satisfy $(\alpha, \beta) \in \mathcal{T}_2$ and

$$\begin{aligned} \gamma, \delta, \varepsilon \geq 0, \quad \alpha - \gamma - \varepsilon \geq 0, \quad \beta - \delta \geq 0, \quad 1 - 2\beta + \delta - \gamma - \varepsilon \geq 0, \\ 1 - \alpha - \beta - \varepsilon \geq 0, \quad \beta - \delta + \varepsilon + \gamma - \alpha \geq 0. \end{aligned} \tag{4}$$

Define $\gamma^* = \alpha^2$, $\delta^* = \beta^2$, $\varepsilon^* = \alpha(1 - \alpha - \beta)$. The following technical condition about the second moment was proposed in [11]. If we prove the condition holds then due to work of Mossel et al. [10] Theorem 2 will follow, and further, due to work of Sly [11] then Theorem 1 will follow.

Condition 1 (Condition 1.2¹ of Sly [11]) Fix $\lambda > 0$ and let p^+ and p^- be the corresponding probabilities. There exists a constant $\chi > 0$ such that when $|p^+ - \alpha| < \chi$ and $|p^- - \beta| < \chi$ then $g_{\alpha, \beta}(\gamma, \delta, \varepsilon) := \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon)$ achieves its unique maximum in the region (4) at the point $(\gamma^*, \delta^*, \varepsilon^*)$.

Two known regions of λ for which Condition 1 holds are the following: 1) $\Delta \geq 3$, and $\lambda_c(\mathbb{T}_\Delta) < \lambda < \lambda_c(\mathbb{T}_\Delta) + \varepsilon(\Delta)$ for some $\varepsilon(\Delta) > 0$, ([10, Lemma 6.10, Lemma 5.1]); 2) $\Delta = 6$ and $\lambda = 1$, ([11, Section 5]).

In this paper, we show that Condition 1 holds for a broad range of λ , and this is how we prove Theorems 1, and 2. Before stating the range when Condition 1 holds we need to define the following quantity. Let $\lambda_{1/2}(\mathbb{T}_\Delta)$ be the smallest value of λ such that $\phi(\phi(1/2)) = 1/2$. Equivalently $\lambda_{1/2}(\mathbb{T}_\Delta)$ is the minimum solution of

$$\left(1 + (1/\lambda)^{1/\Delta}\right)^{1-1/\Delta} \left(1 - (1/\lambda)^{1/\Delta}\right)^{1/\Delta} = 1. \tag{5}$$

Lemma 2. Condition 1 holds for 1) $\Delta = 3$ and $\lambda > \lambda_c(\mathbb{T}_\Delta)$; and 2) $\Delta > 3$ and $\lambda \in (\lambda_c(\mathbb{T}_\Delta), \lambda_{1/2}(\mathbb{T}_\Delta)]$.

Proof. When Δ and λ are such that $p^+ \leq 1/2$ then the following lemma implies Condition 1 (the precise condition on λ to have $p^+ \leq 1/2$ is $\lambda \in (\lambda_c(\mathbb{T}_\Delta), \lambda_{1/2}(\mathbb{T}_\Delta)]$ where $\lambda_{1/2}(\mathbb{T}_\Delta)$ is defined by (5)).

Lemma 3. Let $\Delta \geq 3$ and let $(\alpha, \beta) \in \mathcal{T}_\Delta$, $\alpha, \beta > 0$, and $\alpha, \beta \leq 1/2$. Then $g_{\alpha, \beta}(\gamma, \delta, \varepsilon) := \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon)$ achieves its unique maximum in the region (4) at the point $(\gamma^*, \delta^*, \varepsilon^*)$.

For all $\lambda \in (\lambda_c(\mathbb{T}_\Delta), \lambda_{1/2}(\mathbb{T}_\Delta)]$ we have $p^+ \leq 1/2$, this follows from the fact that p^+ is continuous in λ (it also follows in a more obvious way from the fact that p^+ increasing, but that fact requires a short proof), and $p^+ = 1/\Delta$ for $\lambda = \lambda_c(\mathbb{T}_\Delta)$. Thus for $\lambda \in (\lambda_c(\mathbb{T}_\Delta), \lambda_{1/2}(\mathbb{T}_\Delta)]$ we have that Condition 1 is satisfied; this follows from Lemma 3 and the fact that $(\alpha, \beta) = (p^+, p^-)$ is contained in the interior of \mathcal{T}_Δ (this follows from Lemma 1).

For $\Delta = 3$ the following lemma establishes Condition 1 in the case complementary to Lemma 3, more precisely, it establishes the condition for $\Delta = 3$ and λ such that $p^+ \geq 1/2$.

³ The numbering in this paper for results from Sly’s work [11] refer to the arXiv version of his paper.

Lemma 4. Fix $\Delta = 3$ and $\lambda > \lambda_c(\mathbb{T}_\Delta)$. Let p^+ and p^- be the corresponding probabilities. Assume that $1/2 \leq p^+ < 1$. There exists a constant $\chi > 0$ such that when $|p^+ - \alpha| < \chi$ and $|p^- - \beta| < \chi$ then $g_{\alpha,\beta}(\gamma, \delta, \varepsilon) := \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon)$ achieves its unique maximum in the region (4) at the point $(\gamma^*, \delta^*, \varepsilon^*)$.

Assume $\Delta = 3$ and $\lambda > \lambda_c(\mathbb{T}_\Delta)$. Let p^+ and p^- be the corresponding probabilities. If $p^+ \leq 1/2$ then Condition 1 follows from Lemma 3 and the fact that $(\alpha, \beta) = (p^+, p^-)$ is contained in the interior of \mathcal{T}_Δ (this follows from Lemma 1). If $p^+ \geq 1/2$ then Condition 1 follows from Lemma 4 and the fact that $(\alpha, \beta) = (p^+, p^-)$ is contained in the interior of \mathcal{T}_Δ . \square

We defer the proofs of Lemmas 3 and 4 to Section 3.

As a corollary of Lemma 2 we get that Condition 1 holds for the range of λ specified in Theorem 2.

Corollary 1. Condition 1 holds for:

1. For $\Delta = 3$ and $\lambda > \lambda_c(\mathbb{T}_3)$.
2. For $\Delta \geq 6$ and $\lambda_c(\mathbb{T}_\Delta) < \lambda \leq \lambda_{1/2}(\mathbb{T}_\Delta)$ and $\lambda_{1/2}(\mathbb{T}_\Delta) > \lambda_c(\mathbb{T}_{\Delta-1})$.
3. For $\Delta = 5$ and $\lambda_c(\mathbb{T}_5) < \lambda \leq \lambda_c(\mathbb{T}_5) + .402912$.
4. For $\Delta = 4$, $\lambda_c(\mathbb{T}_4) < \lambda \leq \lambda_c(\mathbb{T}_4) + .327887$.

Theorem 2 follows from Corollary 1 via the second-moment method. That proof closely follows the work of Mossel et al. [10], and hence we omit the details here and refer the interested reader to the full version of this paper [3]. For Theorem 1 for the case of $\Delta = 3$, we use Theorem 2 (or closely related results) combined with the reduction of Sly [11]. That proof closely follows the work of Sly [11], so once again we refer the interested reader to the full version [3]. To extend the inapproximability result for $\Delta = 3$ to $\Delta \geq 6$ we use the following simple combinatorial result.

Lemma 5. Let G be a graph of maximum degree Δ and let $k > 1$ be an integer. Consider the graph H obtained from G by replacing each vertex by k copies of that vertex and each edge by the complete bipartite graph between the corresponding copies. Then, $Z_G((1 + \lambda)^k - 1) = Z_H(\lambda)$.

2.4 On the Use of Computational Assistance

We are going to use Mathematica to prove inequalities involving rational functions in regions bounded by rational functions. Such inequalities are known to be decidable by Tarski’s quantifier elimination [14], the particular version of Collins algebraic decomposition (CAD) used by Mathematica’s `Resolve` command is described in [13].

3 Analysis of the Second Moment Function

In this section, we will prove Lemmas 3 and 4. The proofs of the lemmas introduced in this section are deferred to the full version of the paper [3].

The derivatives of Φ_2 with respect to $\gamma, \delta, \varepsilon$ are

$$\exp\left(\frac{\partial\Phi_2}{\partial\gamma}\right) = \frac{(1-2\beta+\delta-\gamma-\varepsilon)^\Delta(\alpha-\gamma-\varepsilon)^\Delta(1-2\alpha+\gamma)^{\Delta-1}}{(1-\beta-\gamma-\varepsilon)^\Delta(\beta-\alpha+\gamma-\delta+\varepsilon)^\Delta(\alpha-\gamma)^{\Delta-2}\gamma}, \tag{6}$$

$$\exp\left(\frac{\partial\Phi_2}{\partial\delta}\right) = \frac{(\beta-\alpha-\delta+\gamma+\varepsilon)^\Delta(1-2\beta+\delta)^{\Delta-1}}{(1-2\beta+\delta-\gamma-\varepsilon)^\Delta(\beta-\delta)^{\Delta-2}\delta}, \tag{7}$$

$$\exp\left(\frac{\partial\Phi_2}{\partial\varepsilon}\right) = \frac{(1-2\beta+\delta-\gamma-\varepsilon)^\Delta(\alpha-\gamma-\varepsilon)^\Delta(1-\alpha-\beta-\varepsilon)^\Delta}{\varepsilon^\Delta(\beta-\alpha-\delta+\gamma+\varepsilon)^\Delta(1-\beta-\gamma-\varepsilon)^\Delta}. \tag{8}$$

We first argue that the maximum of $g_{\alpha,\beta}$ cannot occur on the boundary of the region defined by (4).

Lemma 6. *For every $\Delta \geq 3$ and $(\alpha, \beta) \in \mathcal{T}_\Delta$, $g_{\alpha,\beta}(\gamma, \delta, \varepsilon) := \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon)$ attains its maximum in the interior of (4).*

Fix $\Delta, \alpha, \beta, \gamma, \delta$ and view Φ_2 as a function of ε . We follow [10] who solved $\frac{\partial\Phi_2}{\partial\varepsilon} = 0$ (which is equivalent to (8) = 1) for ε , and showed that there is a unique solution in the interior of (4):

$$\hat{\varepsilon} := \hat{\varepsilon}(\alpha, \beta, \gamma, \delta) = \frac{1}{2}(1 + \alpha - \beta - 2\gamma - \sqrt{D}), \tag{9}$$

where $D = (1 + \alpha - \beta - 2\gamma)^2 - 4(\alpha - \gamma)(1 - 2\beta - \gamma + \delta) = (\alpha + \beta - 1)^2 + 4(\alpha - \gamma)(\beta - \delta)$. Note that $\hat{\varepsilon}$ is a maximum of Φ_2 , since

$$\begin{aligned} \frac{\partial\Phi_2}{\partial^2\varepsilon} &= -\frac{\Delta}{1-2\beta+\delta-\gamma-\varepsilon} - \frac{\Delta}{\alpha-\gamma-\varepsilon} - \frac{\Delta}{\varepsilon} - \frac{\Delta}{\beta-\delta-\alpha+\gamma+\varepsilon} \\ &\quad - \frac{\Delta}{1-\alpha-\beta-\varepsilon} + \frac{\Delta}{1-\beta-\gamma-\varepsilon} < 0. \end{aligned}$$

Define

$$\hat{\eta} := \hat{\eta}(\alpha, \beta, \gamma, \delta) = \frac{1}{2}(1 - \alpha + \beta - 2\delta - \sqrt{D}), \tag{10}$$

and note that

$$\begin{aligned} \alpha - \gamma - \hat{\varepsilon} &= \beta - \delta - \hat{\eta} = \frac{1}{2}(-(1 - \alpha - \beta) + \sqrt{D}), \\ (\alpha - \gamma - \hat{\varepsilon})(1 - \alpha - \beta - \hat{\varepsilon} - \hat{\eta}) &= \hat{\varepsilon}\hat{\eta}. \end{aligned} \tag{11}$$

The new parameter $\hat{\eta}$ (not used in [10]) is symmetric with $\hat{\varepsilon}$ (the constrains and formulas we have are invariant under a symmetry that swaps $\alpha, \gamma, \hat{\varepsilon}$ with $\beta, \delta, \hat{\eta}$) and allows for simpler arguments (using the symmetry).

Equation (9) allows us to eliminate variable ε . Let

$$f(\gamma, \delta) := g_{\alpha,\beta}(\gamma, \delta, \hat{\varepsilon}) = \Phi_2(\alpha, \beta, \gamma, \delta, \hat{\varepsilon}).$$

To prove that $(\gamma^*, \delta^*, \varepsilon^*)$ is the unique global maximum of $g_{\alpha,\beta}$ in the interior of the region defined by (4), it suffices to prove that (γ^*, δ^*) is the unique global

maximum of f for (γ, δ) in the interior of the following region (which contains the (γ, δ) -projection of the region defined by (4)):

$$\begin{aligned} 0 \leq \gamma \leq \alpha, \quad 0 \leq \delta \leq \beta, \\ 0 \leq 1 - 2\beta + \delta - \gamma, \quad 0 \leq 1 - 2\alpha + \gamma - \delta. \end{aligned} \tag{12}$$

Each inequality in (12) is implied by the inequalities in (4) (the only non-trivial case is the last inequality which is the sum of $1 - \alpha - \beta - \varepsilon \geq 0$ and $\beta - \delta + \varepsilon + \gamma - \alpha \geq 0$).

The first derivatives of f with respect to γ, δ are

$$\frac{\partial f}{\partial \gamma}(\gamma, \delta) = \Delta \ln W_{11} + \ln W_{12}, \tag{13}$$

$$\frac{\partial f}{\partial \delta}(\gamma, \delta) = \Delta \ln W_{21} + \ln W_{22}, \tag{14}$$

where

$$\begin{aligned} W_{11} &= \frac{(\alpha - \gamma - \hat{\varepsilon})\hat{\varepsilon}(1 - 2\alpha + \gamma)}{\hat{\eta}(\alpha - \gamma)^2} = \frac{\hat{\varepsilon}(1 - 2\alpha + \gamma)}{(1 - \alpha - \beta - \hat{\varepsilon})(\alpha - \gamma)}, & W_{12} &= \frac{(\alpha - \gamma)^2}{(1 - 2\alpha + \gamma)\gamma}, \\ W_{21} &= \frac{(\beta - \delta - \hat{\eta})\hat{\eta}(1 - 2\beta + \delta)}{\hat{\varepsilon}(\beta - \delta)^2} = \frac{\hat{\eta}(1 - 2\beta + \delta)}{(1 - \alpha - \beta - \hat{\eta})(\beta - \delta)}, & W_{22} &= \frac{(\beta - \delta)^2}{(1 - 2\beta + \delta)\delta}. \end{aligned}$$

(The equalities in the definition of W_{11} and the definition of W_{21} follow from (10) and (11).)

To determine whether (13) and (14) are zero it will be useful to understand conditions that make $W_{11}, W_{12}, W_{21}, W_{22}$ greater or equal to 1. The following lemma gives such conditions.

Lemma 7. *For every $(\alpha, \beta) \in \mathcal{T}_2$, and (γ, δ) in the interior of (12),*

$$\begin{aligned} W_{11} \geq 1 &\iff (1 - \alpha)^2\delta + \beta^2(2\alpha - 1 - \gamma) \geq 0, & W_{12} \geq 1 &\iff \gamma \leq \alpha^2, \\ W_{21} \geq 1 &\iff (1 - \beta)^2\gamma + \alpha^2(2\beta - 1 - \delta) \geq 0, & W_{22} \geq 1 &\iff \delta \leq \beta^2. \end{aligned}$$

For every $\Delta \geq 3$, and $(\alpha, \beta) \in \mathcal{T}_\Delta$, we have (γ^*, δ^*) is a stationary point of f (this follows from the fact that for $\gamma = \alpha^2$ and $\delta = \beta^2$ the inequalities on the right-hand sides in Lemma 7 become equalities, and from (13), (14) we have that the derivatives of f vanish). By considering the sign of (13) and (14) and Lemma 7, we have the stationary points of f (and hence of $g_{\alpha, \beta}$) can only be in

$$0 < \gamma \leq \alpha^2, \quad 0 < \delta \leq \beta^2, \tag{15}$$

$$(1 - \alpha)^2\delta + \beta^2(2\alpha - 1 - \gamma) \leq 0, \quad (1 - \beta)^2\gamma + \alpha^2(2\beta - 1 - \delta) \leq 0, \tag{16}$$

or

$$\alpha^2 \leq \gamma < \alpha, \quad \beta^2 \leq \delta < \beta, \tag{17}$$

$$(1 - \alpha)^2\delta + \beta^2(2\alpha - 1 - \gamma) \geq 0, \quad (1 - \beta)^2\gamma + \alpha^2(2\beta - 1 - \delta) \geq 0. \tag{18}$$

We are going to prove that (γ^*, δ^*) is the unique maximum of f for every $\Delta \geq 3$, $(\alpha, \beta) \in \mathcal{T}_\Delta$, $\alpha, \beta > 0$ and $\alpha, \beta \leq 1/2$, by showing that the Hessian matrix of f is always negative definite in the region defined by the union of (15) and (17) (and hence the function f is strictly concave).

Let

$$\begin{aligned} R_1 &= \frac{1-\alpha-\beta}{1-\alpha-\beta-\varepsilon-\eta}, & R_2 &= \frac{\sqrt{D}}{1-2\alpha+\gamma}, & R_3 &= \frac{2(\alpha-\gamma-\varepsilon)}{\alpha-\gamma}, \\ R_4 &= \frac{\sqrt{D}}{\gamma}, & R_5 &= \frac{2(1-\beta-\gamma-\varepsilon)}{\alpha-\gamma}, & R_6 &= \frac{\sqrt{D}}{1-2\beta+\delta}, \\ R_7 &= \frac{2(\alpha-\gamma-\varepsilon)}{\beta-\delta}, & R_8 &= \frac{\sqrt{D}}{\delta}, & R_9 &= \frac{2(1-\beta-\gamma-\varepsilon)}{\beta-\delta}. \end{aligned}$$

Inspecting the R_i we obtain the following observation.

Observation 3 R_1, \dots, R_9 are positive when $(\alpha, \beta) \in \mathcal{T}_2$ and (γ, δ) in the interior of (12).

Let

$$M = \begin{pmatrix} \frac{\partial f}{\partial^2 \gamma}(\gamma, \delta) & \frac{\partial f}{\partial \gamma \partial \delta}(\gamma, \delta) \\ \frac{\partial f}{\partial \delta \partial \gamma}(\gamma, \delta) & \frac{\partial f}{\partial^2 \delta}(\gamma, \delta) \end{pmatrix}.$$

In terms of R_1, \dots, R_9 we have $\frac{\partial f}{\partial \gamma \partial \delta}(\gamma, \delta) = \frac{\partial f}{\partial \delta \partial \gamma}(\gamma, \delta) = \frac{\Delta R_1}{\sqrt{D}}$,

$$\frac{\partial f}{\partial^2 \gamma}(\gamma, \delta) = \frac{1}{\sqrt{D}} \left[(-R_1 + R_2 + R_3)\Delta - R_2 - R_3 - R_4 - R_5 \right], \tag{19}$$

$$\frac{\partial f}{\partial^2 \delta}(\gamma, \delta) = \frac{1}{\sqrt{D}} \left[(-R_1 + R_6 + R_7)\Delta - R_6 - R_7 - R_8 - R_9 \right]. \tag{20}$$

From the positivity of R_1 (when $(\alpha, \beta) \in \mathcal{T}_2$ and (γ, δ) in the interior of (12)) we immediately obtain the following observation.

Observation 4 For every $(\alpha, \beta) \in \mathcal{T}_2$, and (γ, δ) in the interior of (12) we have $\frac{\partial f}{\partial \gamma \partial \delta}(\gamma, \delta) = \frac{\partial f}{\partial \delta \partial \gamma}(\gamma, \delta) > 0$.

We prove the following technical inequality on the R_i .

Lemma 8. For every $(\alpha, \beta) \in \mathcal{T}_2$, and (γ, δ) in the interior of (12) we have $R_1 > R_2 + R_3$ and $R_1 > R_6 + R_7$.

Plugging Lemma 8 into (19) and (20) we obtain:

Corollary 2. For every $(\alpha, \beta) \in \mathcal{T}_2$, and (γ, δ) in the interior of (12) we have $\frac{\partial f}{\partial^2 \gamma}(\gamma, \delta) < 0$ and $\frac{\partial f}{\partial^2 \delta}(\gamma, \delta) < 0$.

To show that M is negative definite it is enough to show that the determinant of M is positive (Corollary 2 implies that the sum of the eigenvalues of M is negative and if the determinant is positive we can conclude that both eigenvalues are negative). We have

$$\begin{aligned}
 \det(M) &= \frac{\partial f}{\partial^2 \gamma}(\gamma, \delta) \cdot \frac{\partial f}{\partial^2 \delta}(\gamma, \delta) - \frac{\partial f}{\partial \gamma \partial \delta}(\gamma, \delta) \cdot \frac{\partial f}{\partial \delta \partial \gamma}(\gamma, \delta) \\
 &= \frac{1}{D} \left\{ (\Delta - 1)^2 \left[(-R_1 + R_2 + R_3)(-R_1 + R_6 + R_7) - R_1^2 \right] \right. \\
 &\quad + (\Delta - 1) \left[(-R_1 + R_2 + R_3)(-R_1 - R_8 - R_9) + \right. \\
 &\quad \quad \left. + (-R_1 + R_6 + R_7)(-R_1 - R_4 - R_5) - 2R_1^2 \right] \\
 &\quad \left. + \left[(-R_1 - R_8 - R_9)(-R_1 - R_4 - R_5) - R_1^2 \right] \right\}. \tag{21}
 \end{aligned}$$

By proving technical inequalities on R_1, \dots, R_9 we will establish the positivity of $\det M$ in the following two cases.

Lemma 9. $\det(M) > 0$ for every $\Delta \geq 3$, $(\alpha, \beta) \in \mathcal{T}_\Delta$, $\alpha, \beta > 0$, (γ, δ) in the interior of (I2) and (γ, δ) in (I5).

Lemma 10. $\det(M) > 0$ for every $\Delta \geq 3$, $(\alpha, \beta) \in \mathcal{T}_\Delta$, $\alpha, \beta \leq 1/2$, $\alpha, \beta > 0$, (γ, δ) in the interior of (I2) and (γ, δ) in (I7).

Assuming the lemmas, the proof of Lemma 3 is immediate.

Proof of Lemma 3. Lemma 9 and Lemma 10 imply that f has a unique maximum at (γ^*, δ^*) for every $\Delta \geq 3$, $(\alpha, \beta) \in \mathcal{T}_\Delta$, $\alpha, \beta > 0$, $\alpha, \beta \leq 1/2$, (γ, δ) in the interior of (I2). This follows from the fact that $\det(M) > 0$ implies that the Hessian of f is negative definite in the region of interest, i.e., where $g_{\alpha, \beta}$ could possibly have stationary points, which in turn implies that f is strictly concave in the region and hence has a unique maximum. By the definition of f , it follows that $g_{\alpha, \beta}$ has a unique maximum at $(\gamma^*, \delta^*, \varepsilon^*)$. \square

When $\Delta = 3$, by (2), we have the solution of $\beta = \phi(\alpha)$ and $\alpha = \phi(\beta)$ satisfies $\alpha^2 - 2\alpha + \alpha\beta + 1 - 2\beta + \beta^2 = 0$. We define \mathcal{T}'_3 to be the set of pairs (α, β) such that $\alpha^2 - 2\alpha + \alpha\beta + 1 - 2\beta + \beta^2 = 0$, $1/2 \leq \alpha < 1$ and $0 < \beta \leq (3 - \sqrt{5})/4$. Our goal is to show that $\det(M) > 0$ for every $(\alpha, \beta) \in \mathcal{T}'_3$, (γ, δ) in the interior of (I2) and (γ, δ) in (I7).

We can rewrite $\det(M)$ using the formula in (21) as

$$\det(M) = \frac{1}{D} \left[3R_1(U_1 + U_2) + U_1U_2 \right] = \frac{U_1}{D} \left[3R_1(1 + U_2/U_1) + U_2 \right], \tag{22}$$

where $U_1 = R_8 + R_9 - 2R_6 - 2R_7$ and $U_2 = R_4 + R_5 - 2R_2 - 2R_3$.

By proving technical inequalities on $U_1, U_2, R_1, \dots, R_9$ we will establish the positivity of $\det M$ in the interior of (I2).

Lemma 11. $\det(M) > 0$ for every $(\alpha, \beta) \in \mathcal{T}'_3$, (γ, δ) in the interior of (I2) and (γ, δ) in (I7).

Proof of Lemma 4. By Lemma 11, $g_{\alpha, \beta}(\gamma, \delta, \varepsilon) := \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon)$ achieves its unique maximum in the region (4) at $(\gamma^*, \delta^*, \varepsilon^*)$, for every $(\alpha, \beta) \in \mathcal{T}'_3$. We next

show that $g_{\alpha,\beta}(\gamma, \delta, \varepsilon) := \Phi_2(\alpha, \beta, \gamma, \delta, \varepsilon)$ also achieves its unique maximum in the region (4) at $(\gamma^*, \delta^*, \varepsilon^*)$ in a small neighborhood of \mathcal{T}'_3 . First note that Φ_2 is continuous. By Lemma 6, we have for sufficiently small $\chi > 0$, the maximum of $g_{\alpha,\beta}$ cannot be obtained on the boundary of the region (4). Note that the derivatives of Φ_2 are continuous. It follows that for sufficiently small $\chi > 0$, all stationary points of $g_{\alpha,\beta}$ have to be close to the point $(\gamma^*, \delta^*, \varepsilon^*)$. We can choose χ such that $\det(M) > 0$ in the neighborhood of the point $(\gamma^*, \delta^*, \varepsilon^*)$, which implies that $g_{\alpha,\beta}$ has a unique stationary point and it is a maximum. \square

References

1. van den Berg, J., Steif, J.E.: Percolation and the hard-core lattice gas model. *Stochastic Processes and their Applications* 49(2), 179–197 (1994)
2. Dyer, M., Frieze, A.M., Jerrum, M.: On counting independent sets in sparse graphs. *SIAM J. Comput.* 31(5), 1527–1541 (2002)
3. Galanis, A., Ge, Q., Štefankovič, D., Vigoda, E., Yang, L.: Improved Inapproximability Results for Counting Independent Sets in the Hard-Core Model (2011) Preprint available from the arXiv at, <http://arxiv.org/abs/1105.5131>
4. Gaunt, D.S., Fisher, M.E.: Hard-Sphere Lattice Gases. I. Plane-Square Lattice. *Journal of Chemical Physics* 43(8), 2840–2863 (1965)
5. Greenhill, C.: The complexity of counting colourings and independent sets in sparse graphs and hypergraphs. *Computational Complexity* 9(1), 52–72 (2000)
6. Jerrum, M.R., Valiant, L.G., Vazirani, V.V.: Random generation of combinatorial structures from a uniform distribution distribution. *Theoretical Computer Science* 43(2-3), 169–186 (1986)
7. Kelly, F.P.: Loss Networks. *Annals of Applied Probability* 1(3), 319–378 (1991)
8. Levin, D.A., Peres, Y., Wilmer, E.L.: *Markov Chains and Mixing Times*. American Mathematical Society, Providence (2009)
9. Molloy, M., Robalewska, H., Robinson, R.W., Wormald, N.C.: 1-factorizations of random regular graphs. *Random Structures and Algorithms* 10(3), 305–321 (1997)
10. Mossel, E., Weitz, D., Wormald, N.: On the hardness of sampling independent sets beyond the tree threshold. *Probability Theory and Related Fields* 143(3-4), 401–439 (2009)
11. Sly, A.: Computational Transition at the Uniqueness Threshold. In: *Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pp. 287–296 (2010), Full version available from the arXiv at, <http://arxiv.org/abs/1005.5584>
12. Štefankovič, D., Vempala, S., Vigoda, E.: Adaptive Simulated Annealing: A Near-optimal Connection between Sampling and Counting. *Journal of the ACM* 56(3), 1–36 (2009)
13. Strzebonski, A.W.: Cylindrical algebraic decomposition using validated numerics. *J. Symb. Comput.* 41(9), 1021–1038 (2006)
14. Tarski, A.: RAND Corporation, Santa Monica, Calif. RAND Corporation, Santa Monica (1948)
15. Valiant, L.G.: The Complexity of Enumeration and Reliability Problems. *SIAM J. Computing* 8(3), 410–421 (1979)
16. Weitz, D.: Counting independent sets up to the tree threshold. In: *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*, pp. 140–149 (2006)

Proximity Oblivious Testing and the Role of Invariances

Oded Goldreich and Tali Kaufman

¹ Department of Computer Science, Weizmann Institute of Science, Rehovot, ISRAEL
oded.goldreich@weizmann.ac.il

² Bar-Ilan University and Weizmann Institute of Science, ISRAEL
kaufmant@mit.edu

Abstract. We present a general notion of properties that are characterized by local conditions that are invariant under a sufficiently rich class of symmetries. Our framework generalizes two popular models of testing graph properties as well as the algebraic invariances studied by Kaufman and Sudan (STOC'08). Our focus is on the case that the property is characterized by a constant number of local conditions and a rich set of invariances.

We show that, in the aforementioned models of testing graph properties, characterization by such invariant local conditions is closely related to proximity oblivious testing (as defined by Goldreich and Ron, STOC'09). In contrast to this relation, we show that, in general, characterization by invariant local conditions is neither necessary nor sufficient for proximity oblivious testing. Furthermore, we show that easy testability is *not* guaranteed even when the property is characterized by local conditions that are invariant under a 1-transitive group of permutations.

Keywords: Property Testing, Graph Properties, Locally Testable Codes, Sparse Linear Codes, The Long-Code.

1 Introduction

In the last couple of decades, the area of property testing has attracted much attention (see, e.g., a couple of recent surveys [16,17]). Loosely speaking, property testing typically refers to sub-linear time probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by making adequate queries; that is, the object is seen as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the size of the object).

While a host of fascinating results and techniques has emerged, the desire for a comprehensive understanding of what makes some properties easy to test (while others are hard to test) is far from being satisfied [1]. Two general approaches

¹ This assertion is not meant to undermine significant successes of several characterization projects, most notably the result of [1].

that seem to have a potential of addressing the question (of “what makes testing possible”) were suggested recently.

1. Restricting attention to the class of *proximity oblivious testers*, which are constant-query testers that reject any object with probability proportional (but not necessarily linearly proportional) to its distance from the predetermined property. Indeed, the characterization of proximity oblivious testers, in two central models of graph properties, obtained in [10], addresses the foregoing question: *graph properties have proximity oblivious testers if and only if they can be characterized in terms of adequate local conditions*²
2. But even before [10], an approach based on *adequately invariant local conditions* was put forward in [14]. It was shown that in the context of testing algebraic properties, *a sufficient condition for testability* (which in fact yields proximity oblivious testers) *is that the property can be characterized in terms of local conditions that are invariant in an adequate sense*. (Here and throughout this paper, a **local condition** means a condition that refers to the value of the function at a constant number of points.)

Thus, these two approaches have a very similar flavor, but they are very different at the actual details. On the one hand, the definition of *proximity oblivious testers* does not refer to any structure of the underlying domain of functions, and the local conditions in the two graph models do not refer explicitly to any invariance. However, invariance under relabeling of the graph’s vertices is implicit in the entire study of graph properties (since the latter are defined in terms of such invariance). On the other hand, the *linear invariances* considered in [14] presume that the functions’ domain can be associated with some vector space and that the properties are invariant under linear transformations of this vector space.

Thus, the first task that we undertake is providing a definition of a general notion of “characterization by invariant local conditions”, where at the very minimum this general definition should unify the notions underlying [10,14]. Such a definition is presented in Section 2. Loosely speaking, a property P is characterized by **invariant local conditions** if P is characterized by a set C of local conditions (i.e., $f \in P$ iff f satisfies all conditions in C) and C is generated by a *constant* number of local conditions coupled with a set of actions that preserves P (i.e., the invariances).

Given such a definition, a natural conjecture that arises, hereafter referred to as the **invariance conjecture**, is that a property has a constant-query proximity-oblivious tester if and only if it can be characterized by invariant local conditions. This conjecture is rigorously formulated within our definitional framework (see Section 2.2) and the current work is devoted to its study. The main results of our study may be stated informally as follows:

1. The invariance conjecture holds in the context of testing graph properties in the dense graph model (see Theorem 3.1).

² We warn that the picture is actually not that clean, because in the case of the bounded-degree model the notion of adequacy includes some technical condition, termed non-propagation.

2. The invariance conjecture holds in the context of testing graph properties in the bounded-degree graph model if and only if all local properties are non-propagating (see Theorem 4.1 and [10, Open Problem 5.8]).
3. In general, the invariance conjecture fails in both directions.
 - (a) Characterization by invariant local conditions is not necessary for proximity oblivious testing. This is demonstrated both by linear properties (see Theorem 5.1) and by the dictatorship property (see Theorem 5.2).
 - (b) Characterization by invariant local conditions is not sufficient for proximity oblivious testing (see Theorem 5.3). This is demonstrated by the property called Eulerian orientation (which refers to the orientation of the edges of a cyclic grid, cf. [6]).

Thus, there are natural settings in which the invariance conjecture holds, but there are also natural settings in which it fails (in each of the possible directions).

The Technical Angle. Items 1 and 2 are established by relying on corresponding results of [10], while our contribution is in observing that the local conditions stated in [10] (in terms of subgraph freeness) coincide with local conditions that are invariant under graph isomorphisms. Actually, to rule out characterizations by other possible invariances (i.e., invariances other than graph isomorphism), we also use the canonization technique of [11, Thm. 2]. In the two examples of Item 3a we rely on the fact that these properties were shown to have (proximity oblivious) testers in [13] and [3], respectively. Thus, in both these cases, our contribution is showing that these properties *cannot* be characterized by invariant local conditions. In Item 3b we rely on a lower bound established in [6] (regarding testing Eulerian orientations of cyclic grids), and our contribution is in observing that this property *can* be characterized by invariant local conditions.

We mention that the property used towards establishing Item 3b is invariant under a 1-transitive³ permutation group. Thus, even such an invariance feature does not guarantee easy testability (i.e., a standard tester of query complexity that only depends on the proximity parameter). Furthermore, this holds even when all local conditions are generated by a single local condition (closed under the said invariance).

Terminology. Throughout the text, when we say proximity oblivious testing we actually mean proximity oblivious testing *in a constant number of queries*. The definition of proximity oblivious testing appears in the appendix.

Organization. In Section 2 we provide a definitional framework that captures the foregoing discussion. In particular, this framework includes a general definition of the notion of characterizations by invariant local conditions and a formal statement of the invariance conjecture. In Section 3 we show that the invariance conjecture holds in the context of testing graph properties in the dense graph model, and in Section 4 we present an analogous conditional (or partial) result

³ A permutation group G over D is called 1-transitive if for every $e, e' \in D$ there exists a $\pi \in G$ such that $\pi(e) = e'$.

for the bounded-degree graph model. The failure of the invariance conjecture is demonstrated in Section 5, and possible conclusions are discussed in Section 6.

2 General Framework

For simplicity, we consider properties of finite functions defined over a finite domain D and having a finite range R , whereas an asymptotic treatment requires considering properties that are infinite sequences of such properties (i.e., a sequence of the type $(P_n)_{n \in \mathbb{N}}$ where P_n is a set of functions from D_n to R_n).⁴ Still, we shall just write P, D, R , and (in order for our asymptotic statements to make sense) one should think of P_n, D_n, R_n . In particular, when we say that some quantity is a “constant”, we actually think of D as growing (along with P and possibly R), while the said quantity remains fixed. Thus, in the rest of our presentation, D and R should be considered as generic sets having a variable size, although they will be often omitted from definitions and notations.

2.1 Characterization by Generated Constraints

First we need to define what we mean by a constraint. A constraint will be a pair consisting of domain elements and a Boolean predicate applied to the corresponding values, and it is satisfied by a function f if applying the predicate to the f -values at the specified locations yields the Boolean value 1 (representing true).

Definition 2.1 (constraints): *A constraint is a pair $((e_1, \dots, e_c), \phi)$ such that e_1, \dots, e_c are distinct elements in D , and $\phi : R^c \rightarrow \{0, 1\}$ is an arbitrary predicate. We say that the foregoing is a constraint of arity c (or a c -constraint). A function $f : D \rightarrow R$ is said to satisfy the foregoing constraint if $\phi(f(e_1), \dots, f(e_c)) = 1$.*

Note that at this point the predicate ϕ may depend on the sequence of elements (e_1, \dots, e_c) . Such a dependence will not exist in the case that a large set of constraints is generated based on few constraints (as in Definition 2.3).

The next notion is of characterization by a set of constraints. A property P of functions is characterized by a set of constraints if f is in P if and only if f satisfies all the constraints in the set.

Definition 2.2 (characterization by constraints): *Let C be a set of constraints and P be a property. We say that P is characterized by C if for every $f : D \rightarrow R$ it holds that $f \in P$ if and only if f satisfies each constraint in C .*

Next, we consider the set of constraints generated by the combination of (1) a fixed set of constraints, (2) a group of permutations over D , and (3) a group of

⁴ The reader may think of $n = |D_n|$, but it is helpful not to insist that $D_n = [n]$. On the other hand, the set R_n may be independent of n (cf., e.g., the case of Boolean functions).

permutations over R . For starters, the reader is advised to think of the second group as of the trivial group containing only the identity permutation. In general, we shall consider a subset of the set of all pairs consisting of a permutation as in (2) and a permutation as in (3).

Definition 2.3 (generated constraints): *Let C be a finite set of c -constraints, and M be a set of pairs consisting of a permutation over D and a permutation over R (i.e., for any $(\pi, \mu) \in M$ it holds that π is a permutation of D and μ is a permutation of R). The set of constraints generated by C and M , denoted $\text{CONS}(C, M)$, is defined by*

$$\text{CONS}(C, M) \stackrel{\text{def}}{=} \{((\pi(e_1), \dots, \pi(e_c)), \phi \circ \mu^{-1}) : ((e_1, \dots, e_c), \phi) \in C, (\pi, \mu) \in M\} \tag{1}$$

where $\phi \circ \mu^{-1}(v_1, \dots, v_c)$ denotes $\phi(\mu^{-1}(v_1), \dots, \mu^{-1}(v_c))$.

Note that saying that f satisfies $((\pi(e_1), \dots, \pi(e_c)), \phi \circ \mu^{-1})$ means that

$$(\phi \circ \mu^{-1})(f(\pi(e_1)), \dots, f(\pi(e_c))) = \phi(\mu^{-1}(f(\pi(e_1))), \dots, \mu^{-1}(f(\pi(e_c)))) = 1,$$

which means that $\mu^{-1} \circ f \circ \pi$ satisfies the constraint $((e_1, \dots, e_c), \phi)$. Regarding the use of $\mu^{-1} \circ f \circ \pi$ rather than $\mu \circ f \circ \pi$, see the discussion following Definition 2.5.

Notation: As in Definition 2.3, it will be convenient to generalize functions to sequences over their domain. That is, for any function F defined over some set S , and for any $e_1, \dots, e_t \in S$, we denote the sequence $(F(e_1), \dots, F(e_t))$ by $F(e_1, \dots, e_t)$. Throughout the text, id will be used to denote the identity permutation, where the domain is understood from the context.

2.2 The Invariance Condition

We are now ready to formulate the invariance condition. We consider a group of pairs (π, μ) such that π is a permutation over D and μ is a permutation over R with a group operation that corresponds to component-wise composition of permutations (i.e., $(\pi_1, \mu_1) \odot (\pi_2, \mu_2) = (\pi_1 \circ \pi_2, \mu_1 \circ \mu_2)$, where \odot denotes the group operation). We call such a group a **group of permutation pairs**, and note that it need not be a direct product of a group of permutation over D and a group of permutations over R .

Definition 2.4 (the invariance condition): *A property P satisfies the invariance condition if there exists a constant, denoted c , a finite set of c -constraints, denoted C , and a group, denoted M , of permutation pairs over $D \times R$ such that P is characterized by $\text{CONS}(C, M)$. In this case, we also say that P satisfies the invariance condition w.r.t M .*

The Invariance Condition and Covering the Domain. *We confine our discussion to the case that the domain contains only elements that are influential w.r.t the property P ; that is, for every $e \in D$, there exists $f_1 \in P$ and $f_0 \notin P$ such*

that $f_1(x) = f_0(x)$ for every $x \in D \setminus \{e\}$. Observe that if property P satisfies the invariance condition w.r.t M , then M induces a transitive permutation group on a constant fraction of D . This follows because the permutation group (over D) induced by M must map a constant number of elements (i.e., those appearing in the constraint set C) to all elements of D .

The Main Question. We are interested in the relation between satisfying the invariance condition and having a proximity oblivious tester (of constant-query complexity). One natural conjecture, hereafter referred to as the **invariance conjecture**, is that *a property satisfies the invariance condition if and only if it has a proximity oblivious tester*. Weaker forms of this conjecture refer to its validity within various models of property testing. This leads us to ask what “models of property testing” are.

2.3 Models of Property Testing

Natural models of property testing can be defined by specifying the domain and range of functions (i.e., D and R) as well as the closure features of the properties in the model⁵. We elaborate below (and mention that this view was elaborated independently by Sudan [19]).

For example, the model of testing graph properties in the adjacency matrix representation, introduced in [7], refers to $D = \binom{[N]}{2}$ and $R = \{0, 1\}$ as well as to the permutation group over D that is defined by all relabeling of $[N]$. Specifically, an N -vertex graph is represented by the Boolean function $g : \binom{[N]}{2} \rightarrow \{0, 1\}$ such that $g(\{u, v\}) = 1$ if and only if u and v are adjacent in the graph. Here an adequate closure feature gives rise to graph properties, where P is a graph property if, for every such function g and every permutation ψ over $[N]$, it holds that $g \in P$ iff $g_\psi \in P$, where $g_\psi(\{u, v\}) \stackrel{\text{def}}{=} g(\{\psi(u), \psi(v)\})$.

In general, closure features are defined by groups of pairs of permutations, just as those in Definition 2.4.

Definition 2.5 (closure features): *Let M be as in Definition 2.4. We say that a property P is closed under M if, for every $(\pi, \mu) \in M$, it holds that $f \in P$ if and only if $\mu \circ f \circ \pi^{-1} \in P$.*

Note that $\mu \circ f \circ \pi^{-1}$ (rather than $\mu \circ f \circ \pi$) is indeed the natural choice, since f maps D to R whereas the new function $f' = \mu \circ f \circ \pi^{-1}$ is meant to map $\pi(D)$ to $\mu(R)$; thus, when f' is applied to $e' = \pi(e)$ this results in first recovering e , next applying f , and finally applying μ .

Definition 2.6 (closure-based models of property testing): *The model of M consists of the class of all properties that are closed under M .*

⁵ In addition, one may consider sub-models that are obtained by requiring the functions in such a model to satisfy some auxiliary properties.

For example, the model of testing graph properties in the adjacency matrix equals the *model of M* , where M is the set of all pairs (π, id) such that $\pi : \binom{[N]}{2} \rightarrow \binom{[N]}{2}$ is induced by a permutation of $[N]$ (i.e., there exists a permutation ψ over $[N]$ such that $\pi(\{u, v\}) = \{\psi(u), \psi(v)\}$, for all $\{u, v\} \in D = \binom{[N]}{2}$). We comment that not all “popular models of property testing” can be reduced to Definition 2.6, but nevertheless Definition 2.6 is a good starting point; that is, various models can be naturally defined as subclasses of the class of all properties that are closed under some group M (where typically in such cases the subclass are characterized by a set of constraints that are generated as in Definition 2.3).⁶

We observe that closure under M is a necessary condition for satisfying the invariance condition with respect to M .

Proposition 2.7 *If P satisfies the invariance condition w.r.t M , then P is closed under M .*

The quite straightforward proof can be found in the full version of this work [8].

3 The Invariance Conjecture Holds in the Dense Graph Model

We prove that the invariance conjecture holds in the special case of graph properties in the adjacency matrix representation model (a.k.a the dense graph model). Recall that in this model, an N -vertex graph is represented by the (symmetric) Boolean function $g : [N] \times [N] \rightarrow \{0, 1\}$ such that $g(u, v) = 1$ if and only if u and v are adjacent in the graph.

We rely on a recent result of [10], which states that (in this model) P has a proximity oblivious tester if and only if it is a subgraph-freeness property. We observe that being a subgraph-freeness property is equivalent to satisfying the invariance condition *with respect to the canonical set*, where the *canonical set* has the form $M = M' \times \{\text{id}\}$ such that M' is the group of permutations over vertex-pairs that is induced by vertex-relabeling⁷ (Indeed, the canonical set is the very set that defines the current model; see Section 2.3). So it is left to show that P *satisfies the invariance condition if and only if P satisfies the invariance condition with respect to the canonical set*. We thus get

Theorem 3.1 *Suppose that P is a set of Boolean functions over the set of unordered pairs over $[N]$ such that P is closed under relabeling of the base set*

⁶ Indeed, an alternative formulation of the model of testing graph properties in the adjacency matrix representation is obtained by starting from $D = [N] \times [N]$ and M that equals all pairs (π, id) such that $\pi(u, v) = (\psi(u), \psi(v))$, for some permutation ψ over $[N]$ and all $(u, v) \in D = [N] \times [N]$. In such a case, we consider the subclass of symmetric functions (i.e., functions g such that $g(u, v) = g(v, u)$ for all $(u, v) \in D$).

⁷ Note that M' is a permutation group over $\binom{[N]}{2}$; it contains only permutations of the form π_ψ such that $\pi_\psi(\{u, v\}) = \{\psi(v), \psi(u)\}$, where ψ is an arbitrary permutation over $[N]$.

(i.e., P is a graph property that refers to the adjacency representation of graphs). Then, P has a proximity oblivious tester if and only if P satisfies the invariance condition. Furthermore, if P satisfies the invariance condition, then it satisfies this condition with respect to the canonical set.

The theorem follows by using the aforementioned ideas and proving the further claim. The latter is proved by using the canonization technique of [11, Thm. 2]. The details appear in the full version of this work [8].

4 The Invariance Conjecture in the Bounded-Degree Graph Model

The next natural challenge is proving a result analogous to Theorem 3.1 for the bounded-degree graph model (introduced in [9]). Unfortunately, only a partial result is established here, because of a difficulty that arises in [10, Sec. 5] (regarding “non-propagation”), to be discussed below.

But first, we have to address a more basic difficulty that refers to fitting the bounded-degree graph model within our framework (i.e., Section 2.3). Recall that the standard presentation of the bounded-degree model represents an N -vertex graph of maximum degree d by a function $g : [N] \times [d] \rightarrow \{0, 1, \dots, N\}$ such that $g(v, i) = u \in [N]$ if u is the i^{th} neighbor of v and $g(v, i) = 0$ if v has less than i neighbors. This creates technical difficulties, which can be resolved in various ways.⁸ The solution adopted here is to modify the representation of the bounded-degree graph model such that N -vertex graphs are represented by functions from $[N]$ to subsets of $[N]$. Specifically, such a graph is represented by a function $g : [N] \rightarrow 2^{[N]}$ such that $g(v)$ is the set of neighbors of vertex v . Furthermore, we are only interested in functions g that describe undirected graphs, which means that $g : [N] \rightarrow 2^{[N]}$ should satisfy $u \in g(v)$ iff $v \in g(u)$ (for every $u, v \in [N]$).

Theorem 4.1 *Suppose that P is a set of functions from $[N]$ to $\{S \subset [N] : |S| \leq d\}$ that corresponds to an undirected graph property; in particular, P is closed under the following canonical set M_0 defined by $(\pi, \mu) \in M_0$ if and only if π is a permutation over $[N]$ and μ acts analogously on sets (i.e., $\mu(S) = \{\pi(v) : v \in S\}$).⁹ Then:*

1. If P has a proximity oblivious tester, then it satisfies the invariance condition.

⁸ The problem is that here it is important to follow the standard convention of allowing the neighbors of each vertex to appear in arbitrary order (as this will happen under relabeling of vertex names), but this must allow us to permute over $[d]$ without distinguishing vertices from the 0-symbol. One possibility is to give up the standard convention by which the vertices appear first and 0-symbols appear at the end of the list. We choose a different alternative.

⁹ Recall that we also assume that for every $g \in P$ it holds that $u \in g(v)$ iff $v \in g(u)$ (for every $u, v \in [N]$). We note that this extra property is easy to test.

2. If P satisfies the invariance condition, then it satisfies it with respect to the canonical set, and it follows that P is a generalized subgraph freeness property (as defined in [10, Def. 5.1]).

Recall that by [10, Sec. 5], if P is a generalized subgraph freeness property that is non-propagating, then P has a proximity oblivious tester. But it is unknown whether each generalized subgraph freeness property is non-propagating. (We note that this difficulty holds even for properties that satisfy the invariance condition with respect to the canonical set.¹⁰)

As in the dense graph model (i.e., Theorem 3.1), the key observation is that a property in this model satisfies the invariance condition with respect to the canonical set if and only if it is a *generalized* subgraph-freeness property (as defined in [10, Def. 5.1]). Thus, Part (1) of Theorem 4.1 follows immediately from [10, Thm. 5.5], and the point is proving Part (2). For details, see the full version of this work [8].

5 The Invariance Conjecture Fails in Some Cases

We show that, in general, the invariance condition is neither necessary nor sufficient for the existence of proximity oblivious testers (POTs).

5.1 The Invariance Condition Is Not Necessary for POT

We present two examples (i.e., properties) that demonstrate that satisfying the invariance condition is not necessary for having a proximity oblivious tester. Both examples are based on sparse linear codes that have (proximity oblivious) codeword tests (i.e., these codes are locally testable). In both cases, the key observation is that satisfying the invariance condition with respect to M (as in Definition 2.4) requires that M is “rich enough” since the domain permutations should map a fixed number of elements to all the domain elements. On the other hand, Proposition 2.7 requires that the property is closed under M , whereas this is shown to be impossible in both examples. In the first example, presented next, the property will be shown to be closed only under the trivial pair (id, id) .

Theorem 5.1 *There exists a property, denoted P , of Boolean functions such that P has a proximity oblivious tester but does not satisfy the invariance condition. Furthermore, P is a linear property; that is, if $f_1, f_2 \in P$ then $f_1 + f_2 \in P$, where $(f_1 + f_2)(x) = f_1(x) \oplus f_2(x)$ for every x .*

¹⁰ In fact, the negative example in [10, Prop. 5.4] can arise in our context. Specifically, consider the set of constraints generated by the constraint $((1, 2), \phi)$ such that $\phi(S_1, S_2) = 1$ iff both (1) $|\{i \in \{1, 2\} : S_i = \emptyset\}| \neq 1$ and (2) $|S_1| \in \{0\} \cup \{2i - 1 : i \in \mathbb{N}\}$. (Indeed, condition (1) mandates that if the graph contains an isolated vertex then it contains no edges, whereas condition (2) mandates that all non-isolated vertices have odd degree.)

Theorem 5.1 is proved by considering a random linear property of dimension $\ell = O(\log n)$. That is, for uniformly selected functions $g_1, \dots, g_\ell : [n] \rightarrow \{0, 1\}$, we consider the property $P_n = \{\sum_{i \in I} g_i : I \subseteq [\ell]\}$. It was shown in [13] that, with high probability over these random choices, the property P has a proximity oblivious tester. In the full version of this work [8] we show that, with high probability over these random choices, the property P does not satisfy the invariance condition.

Testing the Long-Code (a.k.a dictatorship tests). We refer to the property $P = (P_n)$, where for $n = 2^\ell$, it holds that $f : \{0, 1\}^\ell \rightarrow \{0, 1\}$ is in P_n if and only if there exists $i \in [\ell]$ such that $f(\sigma_1 \cdots \sigma_\ell) = \sigma_i$. Such a function f is a dictatorship (determined by bit i) and can be viewed as the i^{th} codeword in the long-code (i.e., the long-code encoding of i). Note that this property is closed under the pair (π, id) , where π is a permutation π over $\{0, 1\}^\ell$, if and only if there exists a permutation ϕ over $[\ell]$ such that $\pi(\sigma_1 \cdots \sigma_\ell) = \sigma_{\phi(1)} \cdots \sigma_{\phi(\ell)}$. (An analogous consideration applies to pairs (π, flip) , where $\text{flip}(\sigma) = 1 - \sigma$ for every $\sigma \in \{0, 1\}$.) We shall show that these are the only pairs under which the dictatorship property is closed, and it will follow that the dictatorship property violates the invariance condition.

Theorem 5.2 *The dictatorship property violates the invariance condition, although it has a proximity oblivious tester.*

The fact that the dictatorship property has a proximity oblivious tester is established in [3][15][11]. In the full version of this work [8] we show that this property violates the invariance condition, since it is not closed under pairs (π, μ) unless π either preserves the (Hamming) weight of the strings or preserves this weight under flipping.

5.2 The Invariance Condition Is Not Sufficient for POT

We next demonstrate that the invariance condition does not suffice for obtaining a proximity oblivious tester. Actually, the following example also shows that the invariance condition does not suffice for the standard definition of testing (with query complexity that only depends on the proximity parameter).

Theorem 5.3 *There exists a property, denoted P , of Boolean functions such that P satisfies the invariance condition but has no proximity oblivious tester. Furthermore, the invariant condition holds with respect to a single constraint that refers to four domain elements, and a group of domain permutations that is 1-transitive. Moreover, P cannot be tested (in the standard sense) within query complexity that only depends on the proximity parameter.*

Theorem 5.3 is proved by considering the set of Eulerian orientations of fixed (and highly regular) bounded-degree graphs, and relying on [6, Thm. 9.14], which

¹¹ The longcode test of [3] only refers to the case that ℓ is a power of 2.

proves an $\Omega(\log \ell)$ query lower bound on the complexity of testing whether the orientation of an ℓ -by- ℓ cyclic grid is Eulerian. It follows that this property has no proximity oblivious tester, while we show (in the full version of this work [8]) that this property satisfies the invariance condition.

6 Conclusions

While the invariance conjecture holds in two natural models of testing graph properties, it was shown to fail in other settings. These failures, described in Section 5, are of three different types.

1. As shown in Theorem 5.1, proximity oblivious testers exist also for properties that are only closed under the identity mapping. That is, a strong notion of testability is achievable also in the absence of any invariants.
2. As shown in Theorem 5.2, the existence of proximity oblivious testers for properties that do not satisfy the invariance condition is not confined to unnatural properties and/or to properties that lack any invariance.
3. As shown in Theorem 5.3, the invariance condition does not imply the existence of a standard tester of query complexity that only depends on the proximity parameter. (Note that the non-existence of such testers implies the non-existence of proximity oblivious testers.) Furthermore, this holds even if the invariance condition holds with respect to a group of domain permutations that is 1-transitive and the set of local conditions is generated by a single condition (closed under this permutation group).

Our feeling is that the fact that the invariance condition is not necessary for proximity oblivious testing is less surprising than the fact that the former is insufficient for the latter. Giving up on the necessity part, we wonder whether a reasonable strengthening of the invariance condition may suffice for proximity oblivious testing.

A natural direction to consider is imposing additional restrictions on the group of domain permutations. As indicated by Theorem 5.3, requiring this group to be 1-transitive does not suffice, and so one is tempted to require this group to be 2-transitive¹² (as indeed suggested in [12] w.r.t standard testing)¹³ Recalling that if P is closed under a 2-transitive group (over the domain) then P is self-correctable (and thus consists of functions that are pairwise far apart), one may also wonder about only requiring 1-transitivity but restricting attention to properties that consist of functions that are pairwise far apart. We mention that

¹² A permutation group G over D is called 2-transitive if for every $(e_1, e_2), (e'_1, e'_2) \in \binom{D}{2}$ there exists a $\pi \in G$ such that $\pi(e_1) = e'_1$ and $\pi(e_2) = e'_2$.

¹³ Recall that here we refer to a set of local conditions that is generated by a *constant* number of local condition (closed under a 2-transitive permutation group). In contrast, Ben-Sasson *et al.* [4] have recently shown that a set of local conditions that is generated by a *non-constant* number of local condition (closed under a 2-transitive permutation group) can yield a non-testable property.

the property used in the proof of Theorem 5.3 contains functions that are close to one another.

Actually, restricting attention to properties that are closed under a 1-transitive group of domain permutations, we may return to the question of necessity and ask whether the existence of proximity oblivious testers in this case implies the invariance condition. Note that our proofs of Theorems 5.1 and 5.2 rely on the fact that the corresponding group is not 1-transitive (e.g., in the first case the group action is trivial and in the second case it has a non-constant number of orbits).

An Alternative Perspective. We mention that Sudan’s perspective on the role of invariance (cf. [19,20]) is different from the one studied in the current work. In particular, Sudan suggests to view the role invariance as a theme (or a technique, akin to others surveyed in [17,20]), which is indeed surveyed in [20, Sec. 5]. From this perspective, Sudan [20, Sec. 6] views our work as pointing out inherent limitations on the applicability of the “theme of invariances”, and concludes that “despite the limitations, invariances have significant unifying power (even if they do not explain everything).”

Acknowledgments. This work was partially supported by the Israel Science Foundation (grant No. 1041/08). Tali Kaufman was partially supported by a Koshland Fellowship and by an Alon fellowship. We are grateful to Dana Ron for useful discussions. We also thank the anonymous reviewers of *RANDOM’11* for comments regarding a previous write-up of this work.

References

1. Alon, N., Fischer, E., Newman, I., Shapira, A.: A Combinatorial Characterization of the Testable Graph Properties: It’s All About Regularity. In: 38th STOC, pp. 251–260 (2006)
2. Alon, N., Shapira, A.: A Characterization of Easily Testable Induced Subgraphs. *Combinatorics Probability and Computing* 15, 791–805 (2006)
3. Bellare, M., Goldreich, O., Sudan, M.: Free bits, PCPs and non-approximability – towards tight results. *SIAM Journal on Computing* 27(3), 804–915 (1998)
4. Ben-Sasson, E., Maatouk, G., Shpilka, A., Sudan, M.: Symmetric LDPC codes are not necessarily locally testable. *ECCC*, TR10-199 (2010)
5. Blum, M., Luby, M., Rubinfeld, R.: Self-Testing/Correcting with Applications to Numerical Problems. *JCSS* 47(3), 549–595 (1993)
6. Fischer, E., Lachish, O., Matsliah, A., Newman, I., Yahalom, O.: On the Query Complexity of Testing Orientations for Being Eulerian. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) *APPROX and RANDOM 2008*. LNCS, vol. 5171, pp. 402–415. Springer, Heidelberg (2008) Full version available from, <http://www.cs.technion.ac.il/~oyahalom>
7. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *Journal of the ACM*, 653–750 (July 1998)
8. Goldreich, O., Kaufman, T.: Proximity Oblivious Testing and the Role of Invariances. In: Goldreich, O. (ed.) *Studies in Complexity and Cryptography*. LNCS, vol. 6650, pp. 173–190. Springer, Heidelberg (2011)

9. Goldreich, O., Ron, D.: Property Testing in Bounded Degree Graphs. *Algorithmica* 32(2), 302–343 (2002)
10. Goldreich, O., Ron, D.: On Proximity Oblivious Testing. *ECCC*, TR08-041 (2008); Also in the Proceedings of the 41st STOC 2009
11. Goldreich, O., Trevisan, L.: Three theorems regarding testing graph properties. *Random Structures and Algorithms* 23(1), 23–57 (2003)
12. Grigorescu, E., Kaufman, T., Sudan, M.: 2-Transitivity is Insufficient for Local Testability. In: 23rd CCC, pp. 259–267 (2008)
13. Kaufman, T., Sudan, M.: Sparse Random Linear Codes are Locally Testable and Decodable. In: The Proceedings of the 48th FOCS, pp. 590–600 (2007)
14. Kaufman, T., Sudan, M.: Algebraic Property Testing: The Role of Invariances. In: 40th STOC, pp. 403–412 (2008)
15. Parnas, M., Ron, D., Samorodnitsky, A.: Testing basic boolean formulae. *SIAM Journal on Discrete Math.* 16(1), 20–46 (2002)
16. Ron, D.: Property Testing: A Learning Theory Perspective. *Foundations and Trends in Machine Learning* 1(3), 307–402 (2008)
17. Ron, D.: Algorithmic and Analysis Techniques in Property Testing. In: *Foundations and Trends in TCS* (to appear)
18. Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing* 25(2), 252–271 (1996)
19. Sudan, M.: Invariance in Property Testing. *ECCC*, TR10-051 (2010)
20. Sudan, M.: Testing Linear Properties: Some General Themes. *ECCC*, TR11-005 (2011)

Appendix: Property Testing and Proximity Oblivious Testers

We first recall the standard definition of property testing.

Definition A.1 (property tester): *Let $P = \bigcup_{n \in \mathbb{N}} P_n$, where P_n is a set of functions defined over the domain D_n . A tester for property P is a probabilistic oracle machine T that satisfies the following two conditions:*

1. *The tester accepts each $f \in P$ with probability at least $2/3$; that is, for every $n \in \mathbb{N}$ and $f \in P_n$ (and every $\epsilon > 0$), it holds that $\Pr[T^f(n, \epsilon) = 1] \geq 2/3$.*
2. *Given $\epsilon > 0$ and oracle access to any f that is ϵ -far from P , the tester rejects with probability at least $2/3$; that is, for every $\epsilon > 0$, every $n \in \mathbb{N}$ and f over D_n , if $\delta_P(f) > \epsilon$, then $\Pr[T^f(n, \epsilon) = 0] \geq 2/3$, where $\delta_P(f) \stackrel{\text{def}}{=} \min_{g \in P_n} \{\delta(f, g)\}$ and $\delta(f, g) \stackrel{\text{def}}{=} |\{e \in D_n : f(e) \neq g(e)\}| / |D_n|$.*

If the tester accepts every function in P with probability 1, then we say that it has one-sided error; that is, T has one-sided error if for every $f \in P$ and every $\epsilon > 0$, it holds that $\Pr[T^f(n, \epsilon) = 1] = 1$. A tester is called non-adaptive if it determines all its queries based solely on its internal coin tosses (and the parameters n and ϵ); otherwise it is called adaptive.

The query complexity of a tester is measured in terms of the size parameter, n , and the proximity parameter, ϵ . In this paper we focus on the case that the complexity only depends on ϵ (and is independent of n).

Turning to the definition of proximity-oblivious testers, we stress that they differ from standard testers in that they do not get a proximity parameter as input. Consequently, assuming these testers have sublinear complexity, they can only be expected to reject functions not in \mathbf{P} with probability that is related to the distance of these functions from \mathbf{P} . This is captured by the following definition.

Definition A.2 (proximity-oblivious tester): *Let $\mathbf{P} = \bigcup_{n \in \mathbb{N}} \mathbf{P}_n$ be as in Definition A.1. A proximity-oblivious tester for \mathbf{P} is a probabilistic oracle machine T that satisfies the following two conditions:*

1. *The machine T accepts each function in \mathbf{P} with probability 1; that is, for every $n \in \mathbb{N}$ and $f \in \mathbf{P}_n$, it holds that $\Pr[T^f(n)=1] = 1$.*
2. *For some (monotone) function $\rho : (0, 1] \rightarrow (0, 1]$, each function $f \notin \mathbf{P}$ is rejected by T with probability at least $\rho(\delta_{\mathbf{P}}(f))$, where $\delta_{\mathbf{P}}(f)$ is as in Definition A.1.*

The function ρ is called the detection probability of the tester T .

In general, the query complexity of a proximity-oblivious tester may depend on the size parameter, n , but in this paper we focus on the case that this complexity is constant.

Note that a proximity-oblivious tester with detection probability ρ yields a standard (one-sided error) property tester of query complexity $O(1/\rho)$.

Optimal Rate List Decoding via Derivative Codes^{*}

Venkatesan Guruswami and Carol Wang

Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract. The classical family of $[n, k]_q$ Reed-Solomon codes over a field \mathbb{F}_q consist of the evaluations of polynomials $f \in \mathbb{F}_q[X]$ of degree $< k$ at n distinct field elements. In this work, we consider a closely related family of codes, called (order m) *derivative codes* and defined over fields of large characteristic, which consist of the evaluations of f as well as its first $m - 1$ formal derivatives at n distinct field elements. For large enough m , we show that these codes can be list-decoded in polynomial time from an error fraction approaching $1 - R$, where $R = k/(nm)$ is the rate of the code. This gives an alternate construction to folded Reed-Solomon codes for achieving the optimal trade-off between rate and list error-correction radius.

Our decoding algorithm is linear-algebraic, and involves solving a linear system to interpolate a multivariate polynomial, and then solving another structured linear system to retrieve the list of candidate polynomials f . The algorithm for derivative codes offers some advantages compared to a similar one for folded Reed-Solomon codes in terms of efficient unique decoding in the presence of side information.

Keywords: Reed-Solomon codes, list error-correction, noisy polynomial interpolation, multiplicity codes, subspace-evasive sets, pseudorandomness.

1 Introduction

Consider the task of communicating information via transmission of n symbols from a large alphabet Σ over an adversarial channel that can arbitrarily corrupt any subset of up to pn symbols (for some error parameter $p \in (0, 1)$). Error-correcting codes can be used to communicate reliably over such a channel. A code C is a judiciously chosen subset of Σ^n that enables recovery of any $\mathbf{c} \in C$ from its distorted version $\mathbf{c} + \mathbf{r}$ so long as \mathbf{r} has at most pn nonzero entries. The rate R of the code C equals $\frac{\log |C|}{n \log |\Sigma|}$, which is the ratio of number of bits of information in the message to the total number of bits transmitted. A basic trade-off in this setting is the one between rate R and error fraction p . Clearly, $R \leq 1 - p$, since the channel can always zero-out the last pn symbols.

^{*} Research supported in part by NSF CCF-0963975 and MSR-CMU Center for Computational Thinking.

1.1 Background

Perhaps surprisingly, the above simple limit can in fact be met, in the model of list decoding. Under list decoding, the error-correction algorithm is allowed to output a list of all codewords within the target error bound pn from the noisy received word. If this output list-size is small, say a constant or some polynomially growing function of the block length, then this is still useful information to have in the worst-case instead of just settling for decoding failure. For a survey of algorithmic results in list decoding, see [4].

List decoding allows one to decode from an error fraction approaching the optimal limit of $1 - R$. In fact, there *exist* codes of rate R that enable decoding up to a fraction $1 - R - \epsilon$ of errors with a list-size bound of $O(1/\epsilon)$ (this follows from standard random coding arguments). However, this is a nonconstructive result, with no deterministic way to construct a good code or an efficient algorithm to list decode it. Recently, it was shown that list decoding from an error rate approaching $1 - R$ is possible constructively, with an explicit code (the *folded Reed-Solomon code*) and a polynomial time decoding algorithm [8]. However, the list-size guarantee is much larger than the $O(1/\epsilon)$ bound achieved by random codes, and is a large polynomial in the block length.

Before we state the result, let us first recall the definition of the well-known *Reed-Solomon codes*. For integer parameters $1 < k < n$, a field \mathbb{F} of size $\geq n$, and a sequence $S = (a_1, \dots, a_n)$ of n distinct elements of \mathbb{F} , the associated Reed-Solomon (RS) code is

$$RS_{\mathbb{F},S}[n, k] = \{ (p(a_1), \dots, p(a_n)) \mid p \in \mathbb{F}[X] \text{ of degree } < k \}.$$

The code $RS_{\mathbb{F},S}[n, k]$ has rate $R = k/n$, and can be list-decoded from up to a $1 - \sqrt{R}$ fraction of errors [2,9]. It is *not* known if list decoding some instantiation of Reed-Solomon codes from a larger radius is possible. At the same time, it is also not known if there are some RS codes for which the list-size could grow super-polynomially beyond this radius. For a more general problem called “list recovery,” it is known that the error fraction cannot be improved for certain RS codes [7].

It turns out one can decode beyond the $1 - \sqrt{R}$ bound by augmenting the RS encoding with some extra information. Parvaresh and Vardy used the evaluations of polynomials carefully correlated with the message polynomial p also in the encoding [11]. However, the encodings of the extra polynomial(s) cost a lot in terms of rate, so their improvement is confined to low rates (at most 1/16) and does not achieve the optimal $1 - R$ radius. Later, Guruswami and Rudra considered a “folded” version of RS codes [8], which is really just the RS code viewed as a code over a larger alphabet. More precisely, the order- m folded Reed-Solomon code is defined as follows.

Definition 1. *Let \mathbb{F} be a field of size q with nonzero elements $\{1, \gamma, \dots, \gamma^{n-1}\}$ for $n = q - 1$. Let $m \geq 1$ be an integer which divides n . Let $1 \leq k < n$ be the degree parameter.*

The folded Reed-Solomon code $FRS_{\mathbb{F}}^{(m)}[k]$ is a code over alphabet \mathbb{F}^m that encodes a polynomial $f \in \mathbb{F}[X]$ of degree k as

$$f(X) \mapsto \left(\begin{bmatrix} f(1) \\ f(\gamma) \\ \vdots \\ f(\gamma^{m-1}) \end{bmatrix}, \begin{bmatrix} f(\gamma^m) \\ f(\gamma^{m+1}) \\ \vdots \\ f(\gamma^{2m-1}) \end{bmatrix}, \dots, \begin{bmatrix} f(\gamma^{n-m}) \\ f(\gamma^{n-m+1}) \\ \vdots \\ f(\gamma^{n-1}) \end{bmatrix} \right). \quad (1)$$

It is shown in [8] that the above code can be decoded up to an error fraction $\approx 1 - \left(\frac{mR}{m-s+1}\right)^{\frac{s}{s+1}}$ for any parameter s , $1 \leq s \leq m$, where $R = k/n$ is the rate of the code. (For $s = 1$, the performance ratio is the $1 - \sqrt{R}$ bound, but the radius improves for large s and $m \gg s$. For example, picking $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$, the list decoding radius exceeds $1 - R - \varepsilon$.) The bound on list-size is q^{s-1} , and the decoding complexity is of the same order. Getting around this exponential dependence on s remains an important theoretical question.

The above algorithm involved finding roots of a univariate polynomial over an extension field of large degree over the base field \mathbb{F} . Recently, an entirely linear-algebraic algorithm was discovered in [6] which avoids the use of extension fields. Although the error fraction decoded by the linear-algebraic algorithm is smaller — it is $\frac{s}{s+1} \left(1 - \frac{mR}{m-s+1}\right)$ for the above folded RS codes — it can still be made to exceed $1 - R - \varepsilon$ for any $\varepsilon > 0$ by the choices $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$. The advantage of the algorithm in [6] is that except for the step of pruning an $(s-1)$ -dimensional subspace to filter the close-by codewords, it has quadratic running time.

1.2 This Work

In this work, we consider another natural variant of Reed-Solomon codes (over fields of large characteristic), called *derivative codes*, defined formally in Section 2. Informally, rather than bundling together evaluations of the message polynomial at consecutive powers of γ , in an order- m derivative code, we bundle together the evaluations of f as well as its first $(m-1)$ derivatives at each point. This might appear to cause a loss in rate (similar to the Parvaresh-Vardy construction [11]), but it does not, as one can pick higher degree polynomials while still maintaining the distance. (For two distinct degree ℓ polynomials, there can be at most ℓ/m points where they and their first $(m-1)$ derivatives agree.)

In Theorem 1 and Corollary 1, we show our main result that derivative codes also achieve list-decoding capacity; that is, for any $\varepsilon > 0$, for the choice $m \approx 1/\varepsilon^2$, we can list decode order- m derivative codes of rate R from a $1 - R - \varepsilon$ fraction of errors. The list-size and running time behavior is similar to the linear-algebraic algorithm for folded RS codes [6], and once again one can find, by just solving two linear systems, a low-dimensional space that contains all the close-by codewords.

Recently, multivariate versions of derivative codes were used in [10] to give locally decodable codes. In that work, these codes were referred to as *multiplicity*

codes, but we refer to our codes as *derivative codes* to emphasize our use of formal derivatives rather than Hasse derivatives in the encoding. A side benefit of the changed terminology is to single out the important univariate case with a different name.

Motivation. Prior to this work, the only known explicit codes list decodable up to the optimal $1 - R$ bound were based on folded Reed-Solomon codes (or with smaller alphabets, certain folded algebraic-geometric codes [5], though these are not fully explicit). It seems like a natural question to seek alternate algebraic constructions of such codes. In addition, there is the possibility that a different construction would have better complexity or list-size guarantees, or offer other advantages.

The derivative code construction is arguably just as natural as the folded Reed-Solomon one. Interestingly, it falls in the framework of Parvaresh-Vardy codes, where the correlated polynomials are formal derivatives. The special properties of derivatives ensures that one need not suffer any loss in rate, and at the same time enable list decoding up to a much larger radius than the bound for RS codes. Further, our algorithm for list decoding derivative codes has some nice properties with respect to decoding with side information, and might have some benefits in practice as well. However, as with the case of folded RS codes, the proven bound on the worst-case list size has an exponential dependence on ε (when the decoding radius is $1 - R - \varepsilon$), and it remains a challenge to improve this. We should note that we cannot rule out the possibility that a better analysis can improve the bound; in general it is a very hard problem to show list-size *lower bounds* for these algebraic codes.

We end the introduction with a brief overview of the algorithm, and speculate on a possible benefit it offers compared to the folded RS case. At a high level, our decoding algorithm is similar to those used for Reed-Solomon and folded Reed-Solomon codes — it consists of an interpolation step, and then a second step to retrieve the list of all polynomials satisfying a certain algebraic condition. The interpolation step consists of fitting a polynomial of the form $A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \dots + A_s(X)Y_s$. (Note that the total degree in the Y_i 's is 1, and we do not use “multiplicities” in the interpolation.) The second step consists of solving the “differential equation” $A_0(X) + A_1(X)f(X) + A_2(X)f'(X) + \dots + A_s(X)f^{(s-1)}(X) = 0$ for low-degree polynomials f . (Independently, a list decoding guarantee similar to the Guruswami-Rudra bound for folded RS codes has been obtained by Bombieri and Kopparty [1] based on using higher powers of Y_i as well as multiplicities in the interpolation.)

The differential equation imposes a system of linear equations on the coefficients of f . The specific structure of this linear system is different from the one for folded RS codes in [6]. In particular, once the values of f and its first $s - 2$ derivatives at some point α (at which the interpolated polynomial A_s doesn't vanish) are known, the rest are determined by the system. This has two advantages. First, having these values (at a random α) as side information immediately leads to an efficient unique decoding algorithm. Second, in practice, A_s may not have many zeroes amongst the evaluation points, in which case we can obtain the

values of $f(a_i), \dots, f^{(s-2)}(a_i)$ from the received word (instead of trying all q^{s-1} possibilities). While we have not been able to leverage this structure to improve the worst-case list-size bound, it is conceivable that additional ideas could lead to some improvements.

2 Derivative Codes

We denote by \mathbb{F}_q the field of q elements. For a polynomial $f \in \mathbb{F}_q[X]$, we denote by f' its formal derivative, i.e. if $f(X) = f_0 + f_1X + \dots + f_\ell X^\ell$, then $f'(X) = \sum_{i=1}^\ell i f_i X^{i-1}$. We denote by $f^{(i)}$ the formal i 'th derivative of f .

Definition 2 (m 'th order derivative code). Let $0 \leq m \in \mathbb{Z}$. Let $a_1, \dots, a_n \in \mathbb{F}_q$ be distinct, and let the parameters satisfy $m \leq k < nm \leq q$. Further assume that $\text{char}(\mathbb{F}_q) > k$.

The derivative code $\text{Der}_q^{(m)}[n, k]$ over the alphabet \mathbb{F}_q^m encodes a polynomial $f \in \mathbb{F}_q[X]$ of degree $k - 1$ by

$$f \mapsto \left(\begin{bmatrix} f(a_1) \\ f'(a_1) \\ \vdots \\ f^{(m-1)}(a_1) \end{bmatrix}, \begin{bmatrix} f(a_2) \\ f'(a_2) \\ \vdots \\ f^{(m-1)}(a_2) \end{bmatrix}, \dots, \begin{bmatrix} f(a_n) \\ f'(a_n) \\ \vdots \\ f^{(m-1)}(a_n) \end{bmatrix} \right). \tag{2}$$

Remark 1. Note that the case $m = 1$ is a Reed-Solomon code.

This code has block length n and rate $R = \frac{k}{nm}$. The minimum distance is $n - \lfloor \frac{k-1}{m} \rfloor \approx (1 - R)n$.

3 List Decoding Derivative Codes

Suppose we have received the corrupted version of a codeword from the derivative code $\text{Der}_q^{(m)}[n, k]$ as a string $\mathbf{y} \in (\mathbb{F}_q^m)^n$, which we will naturally consider as an $m \times n$ matrix over \mathbb{F}_q :

$$\begin{pmatrix} y_{11} & y_{12} & \dots & y_{1n} \\ y_{21} & y_{22} & \dots & y_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ y_{m1} & y_{m2} & \dots & y_{mn} \end{pmatrix}. \tag{3}$$

The goal is to recover all polynomials f of degree $k - 1$ whose encoding (2) agrees with \mathbf{y} in at least t columns. This corresponds to decoding from $n - t$ symbol errors for the derivative code $\text{Der}_q^{(m)}[n, k]$. When $t > (n + k/m)/2$, the polynomial f , if it exists, is unique, and in this regime an efficient decoding algorithm was given in [10] by adapting the Welch-Berlekamp algorithm for Reed-Solomon codes [14, 2].

We adapt the algebraic list-decoding method used for Reed-Solomon and folded Reed-Solomon codes to the derivative code setting. The decoding algorithm consists of two steps — (i) interpolation of an algebraic condition (that

must be obeyed by all candidate polynomials f), and (ii) retrieving the list of candidate solutions f (from the algebraic condition found by the interpolation step).

Our algorithm can be viewed as a higher dimensional analog of the Welch-Berlekamp algorithm, where we use multivariate polynomials instead of bivariate polynomials in the interpolation. This has been used in the context of folded Reed-Solomon codes in [13, Chap. 5] and [6], and here we show that derivative codes can also be list decoded in this framework.

3.1 Interpolation

Let \mathcal{W} denote the \mathbb{F}_q -linear subspace of $\mathbb{F}_q[X, Y_1, \dots, Y_m]$ consisting of polynomials that have total degree at most 1 in the Y_i 's, i.e. \mathcal{W} contains polynomials of the form $B_0(X) + B_1(X)Y_1 + B_2(X)Y_2 + \dots + B_m(X)Y_m$ for some polynomials $B_i \in \mathbb{F}_q[X]$.

Let D be the \mathbb{F}_q -linear map on \mathcal{W} defined as follows: For $p \in \mathbb{F}_q[X]$, and $1 \leq i \leq m$,

$$D(p)(X, Y_1, \dots, Y_m) = p'(X) \tag{4}$$

and

$$D(pY_i)(X, Y_1, \dots, Y_m) = p'(X)Y_i + p(X)Y_{i+1}. \tag{5}$$

where we take $Y_{m+1} = Y_1$.

Let $s, 1 \leq s \leq m$, be an integer parameter in the decoding algorithm. The goal in the interpolation step is to interpolate a **nonzero** polynomial $Q \in \mathbb{F}_q[X, Y_1, Y_2, \dots, Y_s]$ of the form

$$A_0(X) + A_1(X)Y_1 + A_2(X)Y_2 + \dots + A_s(X)Y_s \tag{6}$$

satisfying the following conditions for each $i, 1 \leq i \leq n$:

$$Q(a_i, y_{1i}, \dots, y_{si}) = 0 \text{ and } (D^k Q)(a_i, y_{1i}, \dots, y_{mi}) = 0 \quad (k = 1, \dots, m - s), \tag{7}$$

where D^k denotes the k -fold composition of the map D .

Observation 1. For each i , the conditions (7) are a collection of $(m - s + 1)$ homogeneous linear constraints on the coefficients of the polynomial Q .

The following shows why the interpolation conditions are useful in the decoding context.

Lemma 1. *Suppose Q of the form (6) satisfies the conditions (7). If the received word (3) agrees with the encoding of f at location i , that is, $f^{(j)}(a_i) = y_{j+1,i}$ for $0 \leq j < m$, then the univariate polynomial $\hat{Q}(X) := Q(X, f(X), \dots, f^{(s-1)}(X))$ satisfies $\hat{Q}(a_i) = 0$ as well as $\hat{Q}^{(k)}(a_i) = 0$ for $k = 1, \dots, m - s$, where $\hat{Q}^{(k)}(X)$ is that the k 'th derivative of \hat{Q} .*

Proof. Notice the form that our definition of the map D takes when $Y_i = f^{(i-1)}(X)$ for $1 \leq i \leq m$. We have $D(p) = p'$ for $p \in \mathbb{F}_q[X]$, and $D(pf^{(i-1)}) = p'f^{(i-1)} + pf^{(i)}$, which is simply the product rule for derivatives. Thus when $(y_{1i}, y_{2i}, \dots, y_{mi}) = (f(a_i), f'(a_i), \dots, f^{(m-1)}(a_i))$, the conditions (7) enforce that \hat{Q} and its first $m - s$ derivatives vanish at a_i .

We next argue that a nonzero interpolation polynomial Q exists and can be found efficiently.

Lemma 2. *Let*

$$d = \left\lfloor \frac{n(m - s + 1) - k + 1}{s + 1} \right\rfloor. \tag{8}$$

Then, a nonzero Q of the form (6) satisfying the conditions (7) with $\deg(A_0) \leq d + k - 1$ and $\deg(A_j) \leq d$ for $1 \leq j \leq s$ exists and can be found in $O((nm)^3)$ field operations over \mathbb{F}_q .

Proof. Under the stated degree restrictions, the number of monomials in Q is

$$(d + 1)s + d + k = (d + 1)(s + 1) + k - 1 > n(m - s + 1).$$

where the last inequality follows from the choice (8) of d . The number of homogeneous linear equations imposed on the coefficients of Q in order to meet the interpolation conditions (7) is $n(m - s + 1)$. As this is less than the number of monomials in Q , the existence of a nonzero Q follows, and it can be found by solving a linear system over \mathbb{F}_q with at most nm constraints.

3.2 Retrieving Candidate Polynomials

Suppose we have a polynomial $Q(X, Y_1, \dots, Y_s)$ satisfying the interpolation conditions (7). The following lemma gives an identity satisfied by any f which has good agreement with the received word.

Lemma 3. *If $f \in \mathbb{F}[X]$ has degree at most $k - 1$ and an encoding (2) agreeing with the received word \mathbf{y} in at least t columns for $t > \frac{d+k-1}{m-s+1}$, then*

$$Q(X, f(X), f'(X), \dots, f^{(s-1)}(X)) = 0.$$

Proof. Let $\hat{Q}(X) = Q(X, f(X), \dots, f^{(s-1)}(X))$. By Lemma 1, an agreement in column i means that $\hat{Q}(X)$ satisfies $\hat{Q}(a_i) = 0$ and that the k th derivative $\hat{Q}^{(k)}(a_i)$ is also zero for $k = 1, \dots, m - s$. In particular, t column agreements yield at least $t(m - s + 1)$ roots (counting multiplicities) for \hat{Q} .

The degree of \hat{Q} is at most $d + k - 1$, as f and each of its derivatives has degree at most $k - 1$. Then as \hat{Q} is univariate of degree at most $d + k - 1$, \hat{Q} has at most $d + k - 1$ roots if it is nonzero. Thus if $t > (d + k - 1)/(m - s + 1)$, it must be that $\hat{Q}(X) = 0$.

With our chosen value of d from (8), this means that any f which agrees with \mathbf{y} on more than

$$\frac{n}{s+1} + \frac{s}{s+1} \frac{k-1}{m-s+1} \tag{9}$$

columns satisfies $Q(X, f(X), f'(X), \dots, f^{(s-1)}(X)) = 0$. So in the second step, our goal is to find all polynomials f of degree at most $k-1$ such that

$$A_0(X) + A_1(X)f(X) + A_2(X)f'(X) + \dots + A_s(X)f^{(s-1)}(X) = 0 \tag{10}$$

Let $A_i(X) = \sum_{j=0}^{\deg(A_i)} a_{ij}X^j$ for each i . Note that the above constraint (10) gives a *linear system* over \mathbb{F} in the coefficients of $f = f_0 + f_1X + \dots + f_{k-1}X^{k-1}$. In particular, the set of solutions $(f_0, f_1, \dots, f_{k-1})$ is an affine space, and we can find it by solving the linear system. Our goal now is to bound the dimension of the space of solutions by exposing its special structure and also use this to efficiently find an explicit basis for the space.

Lemma 4. *It suffices to give an algorithm in the case that the constant term a_{s0} of A_s is nonzero.*

Proof. If $A_s(X) \neq 0$, since $\deg(A_s) \leq d < nm \leq q$, then there is some $\alpha \in \mathbb{F}_q$ such that $A_s(\alpha) \neq 0$, so we can consider a “translate” of this problem by α ; that is, $A_s(X + \alpha)$ has nonzero constant term, so we can solve the system with the translated polynomial $Q(X + \alpha, Y_1, \dots, Y_m)$ and recover candidate messages by translating each solution $g(X)$ to $f(X) = g(X - \alpha)$.

If $A_s(X) = 0$, we simply reduce the problem to a smaller one with s rather than $s + 1$ interpolation variables. Note that this must terminate since Q is nonzero and so at least one A_i for $i \geq 1$ is nonzero.

We can now show:

Lemma 5. *If $a_{s0} \neq 0$, the solution space to (10) has dimension at most $s - 1$.*

Proof. For each power X^i , the coefficient of X^i in $A_0(X) + A_1(X)f(X) + \dots + A_s(X)f^{(s-1)}(X)$ is

$$\begin{aligned} & a_{0i} + (a_{10}f_i + a_{11}f_{i-1} + \dots + a_{1i}f_0) + (a_{20}(i+1)f_{i+1} + a_{21}if_i + \dots + a_{2i}f_1) \\ & + \dots + (a_{s0}(i+s-1)(i+s-2) \dots (i+1)f_{i+s-1} + \dots + a_{si}(s-1)!f_{s-1}) \\ = & a_{0i} + \sum_{j=1}^s \sum_{k=0}^i \frac{(k+j-1)!}{k!} a_{j(i-k)} f_{k+j-1}. \end{aligned}$$

If (f_0, \dots, f_{k-1}) is a solution to (10), then this coefficient is zero for every i .

The coefficient of X^i for each i depends only on f_j for $j < i + s$, and the coefficient of f_{i+s-1} is $a_{s0}(i+s-1)(i+s-2) \dots (i+1)$, which is nonzero when $i + s \leq k$ since $\text{char}(\mathbb{F}_q) > k$. Thus, if we fix f_0, f_1, \dots, f_{s-2} , the rest of the coefficients f_{s-1}, \dots, f_{k-1} are uniquely determined. In particular, the dimension of the solution space is at most $s - 1$.

Remark 2. The bound of Lemma 5 is tight for arbitrary linear systems. Indeed, if

$$Q(X, Y_1, \dots, Y_s) = \sum_{i=0}^{s-1} \frac{(-1)^i}{i!} X^i Y_{i+1},$$

then any polynomial of degree less than s with zero constant term satisfies $Q(X, f(X), \dots, f^{(s-1)}(X)) = 0$. This is because any monomial $f(X) = X^j$ for $0 < j \leq s - 1$ is a solution, and our solution space is linear. Of course, we do not know if such a bad polynomial can occur as the output of the interpolation step when decoding a noisy codeword of the derivative code.

Combining these lemmas and recalling the bound (9) on the number of agreements for successful decoding, we have our main result.

Theorem 1 (Main). *For every $1 \leq s \leq m$, the derivative code $Der_q^{(m)}[n, k]$ (where $\text{char}(\mathbb{F}_q) > k$) satisfies the property that for every received word $\mathbf{y} \in \mathbb{F}_q^{nm}$, an affine subspace $S \subseteq \mathbb{F}_q[X]$ of dimension at most $s - 1$ can be found in polynomial time such that every $f \in \mathbb{F}_q[X]$ of degree less than k whose derivative encoding differs from \mathbf{y} in at most*

$$\frac{s}{s + 1} \left(n - \frac{k}{(m - s + 1)} \right)$$

positions belongs to S .

Now by setting $s \approx 1/\varepsilon$ and $m \approx 1/\varepsilon^2$, and recalling that the rate of $Der_q^{(m)}[n, k]$ equals $k/(nm)$, we can conclude the following.

Corollary 1. *For all $R \in (0, 1)$ and all $\varepsilon > 0$, for a suitable choice of parameters, there are derivative codes $Der_q^{(m)}[n, k]$ of rate at least R which can be list decoded from a fraction $1 - R - \varepsilon$ of errors with a list-size of $q^{O(1/\varepsilon)}$.*

4 Some Remarks

We now make a couple of remarks on coping with the large list-size bound in our decoding algorithms.

4.1 Reducing the List Size

One approach to avoid the large list size bound of $\approx q^s$ for the number of codewords near f is to draw codewords from so-called *subspace-evasive* subsets of \mathbb{F}_q^k rather than all of \mathbb{F}_q^k . This approach was used in [6] to reduce the list-size for folded Reed-Solomon codes, and we can gain a similar benefit in the context of list decoding derivative codes. A subset of \mathbb{F}_q^k is (s, L) -subspace-evasive if it intersects with every linear subspace $S \subseteq \mathbb{F}_q^k$ of dimension at most s in at most L points.

For any $\varepsilon > 0$, a probabilistic argument shows that there exist $(s, O(s/\varepsilon))$ -subspace-evasive subsets of \mathbb{F}_q^k of size $q^{(1-\varepsilon)k}$. In fact, we have the following stronger statement, proved in [6]. Fix a basis $1, \beta, \dots, \beta^{k-1}$ of \mathbb{F}_q^k over \mathbb{F}_q and denote $\mathbb{K} = \mathbb{F}_q[X]$. For $P \in \mathbb{K}[X]$ and an integer $r, 1 \leq r \leq k$, define

$$\mathbb{S}(P, r) = \{(a_0, \dots, a_{k-1}) \in \mathbb{F}_q^k \mid P(a_0 + a_1\beta + \dots + a_{k-1}\beta^{k-1}) \in \mathbb{F}_q\text{-span}(1, \beta, \dots, \beta^{r-1})\}.$$

Lemma 6 ([6]). *Let q be a prime power, $k \geq 1$ an integer. Let $\zeta \in (0, 1)$ and $s \in \mathbb{Z}$ satisfying $1 \leq s \leq \zeta k/2$. Let $P \in \mathbb{K}[X]$ be a random polynomial of degree t and define $\mathcal{V} = \mathbb{S}(P, (1 - \zeta)k)$. Then for $t \geq \Omega(s/\zeta)$, with probability at least $1 - q^{-\Omega(k)}$ over the choice of P , \mathcal{V} is an (s, t) -subspace-evasive subset of \mathbb{F}_q^k of size at least $q^{(1-\zeta)k/2}$.*

By taking messages from \mathcal{V} rather than all of \mathbb{F}_q^k , we suffer a small loss in rate, but give a substantial improvement to the list size bound; since our solution space is linear, the number of candidate messages is reduced from $\approx q^s$ to $O(s/\varepsilon)$. In particular, setting our parameters as in Theorem 1, we can list-decode from a $1 - R - \varepsilon$ fraction of errors with a list size of at most $O(1/\varepsilon^2)$. However, the code construction is not explicit but only a randomized (Monte Carlo) one that satisfies the claimed guarantees on list-decoding with high probability.

4.2 Decoding with Side Information

The decoding described in the previous section consists of trying all choices for the coefficients f_0, \dots, f_{s-2} and using each to uniquely determine a candidate for f . Note however that for each i , the f_i is essentially the i th derivative of f evaluated at 0, and can be recovered as $f^{(i)}(0)/i!$. Thus if the decoder somehow knew the correct values of f and its first $s - 1$ derivatives at 0, f could be recovered uniquely (as long as $A_s(0) \neq 0$).

Now, suppose the encoder could send a small amount of information along a noiseless side channel in addition to sending the (much longer) codeword on the original channel. In such a case, the encoder could choose $\alpha \in \mathbb{F}_q$ uniformly at random and transmit $f(\alpha), f'(\alpha), \dots, f^{(s-1)}(\alpha)$ on the noiseless channel. The decoding then fails only if $A_i(\alpha) = 0$ for i which is the largest index such that $A_i(X) \neq 0$. As the $A_i(X)$ have bounded degree, by increasing the field size q , f can be uniquely recovered with probability arbitrarily close to 1. More precisely, we have the following claim.

Theorem 2. *Given a uniformly random $\alpha \in \mathbb{F}_q$ and the values $f(\alpha), f'(\alpha), \dots, f^{(s-1)}(\alpha)$ of the message polynomial f , the derivative code $Der_q^{(m)}[n, k]$ can be uniquely decoded from up to*

$$\frac{s}{s+1} \left(n - \frac{k}{m-s+1} \right)$$

errors with probability at least $1 - \frac{nm}{sq}$ over the choice of α .

Proof. As in the proof of Lemma 4, as long as $A_s(\alpha) \neq 0$, we may translate the problem by α and use the values $f(\alpha), f'(\alpha), \dots, f^{(s-1)}(\alpha)$ to uniquely determine the shifted coefficients g_0, \dots, g_{s-1} .

As $A_s \neq 0$, and A_s is univariate of degree at most d , A_s has at most d roots, and so the probability that $A_s(\alpha) \neq 0$ is at least $1 - d/q \geq 1 - \frac{nm}{sq}$, where the last inequality follows from our choice of $d \leq nm/s$ in (8).

Remark 3. In the context of communicating with side information, there is a generic, black-box solution combining list-decodable codes with hashing to guarantee unique recovery of the correct message with high probability [3]. In such a scheme, the side information consists of a random hash function h and its value $h(f)$ on the message f . The advantage of the solution in Theorem 2 is that there is *no need* to compute the full list (which is the computationally expensive step, since the list size bound depends exponentially on s) and then prune it to the unique solution. Rather, we can uniquely identify the first $(s-1)$ coefficients of the polynomial $f(X + \alpha)$ in the linear system (10), after applying the shift $X \mapsto X + \alpha$, as $f(\alpha), f'(\alpha), \dots, f^{(s-2)}(\alpha)$. Then, as argued in the proof of Lemma 5, the remaining coefficients are determined as linear combinations of these $s-1$ coefficients. So the whole algorithm can be implemented in quadratic time.

Remark 4. The decoder could use the columns of the received word \mathbf{y} as a guess for the side information $f(a_i), f'(a_i), \dots, f^{(s-2)}(a_i)$ for $i = 1, 2, \dots, n$. Since f agrees with \mathbf{y} on more than $t > Rn$ positions, as long as $A_s(a_i) = 0$ for less than t of the evaluation points a_i , we will recover every solution f this way. This would lead to a list size bound of at most $n - t < n$. Unfortunately, however, there seems to be no way to ensure that A_s does not vanish at most (or even all) of the points a_i used for encoding. But perhaps some additional ideas can be used to make the list size polynomial in both q, s , or at least $\exp(O(s))q^c$ for some absolute constant c .

References

1. Bombieri, E., Kopparty, S.: List decoding multiplicity codes (2011) (manuscript)
2. Gemmell, P., Sudan, M.: Highly resilient correctors for multivariate polynomials. *Information Processing Letters* 43(4), 169–174 (1992)
3. Guruswami, V.: List decoding with side information. In: *Proceedings of the 18th IEEE Conference on Computational Complexity (CCC)*, pp. 300–309 (2003)
4. Guruswami, V.: *Algorithmic Results in List Decoding*. *Foundations and Trends in Theoretical Computer Science (FnT-TCS)*, vol. 2. NOW publishers (January 2007)
5. Guruswami, V.: Cyclotomic function fields, Artin-Frobenius automorphisms, and list error-correction with optimal rate. *Algebra and Number Theory* 4(4), 433–463 (2010)
6. Guruswami, V.: Linear-algebraic list decoding of folded Reed-Solomon codes. In: *Proceedings of the 26th IEEE Conference on Computational Complexity* (June 2011)

7. Guruswami, V., Rudra, A.: Limits to list decoding Reed-Solomon codes. *IEEE Transactions on Information Theory* 52(8), 3642–3649 (2006)
8. Guruswami, V., Rudra, A.: Explicit codes achieving list decoding capacity: Error-correction up to the Singleton bound. *IEEE Transactions on Information Theory* 54(1), 135–150 (2008)
9. Guruswami, V., Sudan, M.: Improved decoding of Reed-Solomon and Algebraic-geometric codes. *IEEE Transactions on Information Theory* 45(6), 1757–1767 (1999)
10. Kopparty, S., Saraf, S., Yekhanin, S.: High-rate codes with sublinear-time decoding. *Electronic Colloquium on Computational Complexity*, TR10-148 (2010)
11. Parvaresh, F., Vardy, A.: Correcting errors beyond the Guruswami-Sudan radius in polynomial time. In: *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pp. 285–294 (2005)
12. Sudan, M.: Decoding of Reed-Solomon codes beyond the error-correction bound. *Journal of Complexity* 13(1), 180–193 (1997)
13. Vadhan, S.: Pseudorandomness. *Foundations and Trends in Theoretical Computer Science (FnT-TCS)*. NOW publishers (2010) (to appear) Draft available at, <http://people.seas.harvard.edu/~salil/pseudorandomness/>
14. Welch, L.R., Berlekamp, E.R.: Error correction of algebraic block codes. US Patent Number 4, 633, 470 (December 1986)

Public Key Locally Decodable Codes with Short Keys

Brett Hemenway^{1,*}, Rafail Ostrovsky^{2,**}, Martin J. Strauss^{1,***},
and Mary Wootters¹

¹ University of Michigan

{bhemem,martinjs,wootters}@umich.edu

² UCLA

rafail@cs.ucla.edu

Abstract. This work considers locally decodable codes in the computationally bounded channel model. The computationally bounded channel model, introduced by Lipton in 1994, views the channel as an adversary which is restricted to polynomial-time computation. Assuming the existence of IND-CPA secure public-key encryption, we present a construction of public-key locally decodable codes, with constant codeword expansion, tolerating constant error rate, with locality $\mathcal{O}(\lambda)$, and negligible probability of decoding failure, for security parameter λ . Hemenway and Ostrovsky gave a construction of locally decodable codes in the public-key model with constant codeword expansion and locality $\mathcal{O}(\lambda^2)$, but their construction had two major drawbacks. The keys in their scheme were proportional to n , the length of the message, and their schemes were based on the Φ -hiding assumption. Our keys are of length proportional to the security parameter instead of the message, and our construction relies only on the existence of IND-CPA secure encryption rather than on specific number-theoretic assumptions. Our scheme also decreases the locality from $\mathcal{O}(\lambda^2)$ to $\mathcal{O}(\lambda)$. Our construction can be modified to give a generic transformation of any private-key locally decodable code to a public-key locally decodable code based only on the existence of an IND-CPA secure public-key encryption scheme.

Keywords: public-key cryptography, locally decodable codes, bounded channel.

* Supported in part by ONR under contract N00014-11-1-0392.

** This material is based upon work supported in part by NSF grants 0830803 and 09165174, US-Israel BSF grant 2008411, grants from OKAWA Foundation, IBM, Lockheed-Martin Corporation and the Defense Advanced Research Projects Agency through the U.S. Office of Naval Research under Contract N00014-11-1-0392. The views expressed are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

*** Supported in part by NSF grant CCF 0743372 and DARPA/ONR grant N66001-08-1-2065.

1 Introduction

Error-correcting codes were designed to facilitate message transmission through noisy channels. An error-correcting code consists of two algorithms, an encoding algorithm which takes a message and adds redundancy transforming it into a (longer) codeword. A decoding algorithm takes a (corrupted) codeword and recovers the original message. Although error-correcting codes were designed for data transmission, they have seen widespread use in storage applications. In storage environments, *random access* to the data is often of importance. A code that can recover a single bit of the underlying message by reading a small number of bits of a corrupted codeword is called *locally decodable*. Locally decodable codes were introduced in the context of Probabilistically-Checkable Proofs (PCPs) [BELS91, Sud92, PS94], and were formalized explicitly in the work of Katz and Trevisan [KT00]. Despite significant research (see [Tre04, Yek10] for surveys) locally decodable codes have much larger codeword expansion than their classical counterparts. The most efficient 3-query LDCs are given by Efremenko [Efr09], and have codeword expansion of $\exp(\exp(\mathcal{O}(\sqrt{\log n \log \log n})))$ for messages of length n . While these codes have found many applications towards Probabilistically-Checkable Proofs and Private Information Retrieval, their expansion rate is far too large for data storage applications. Recent work by Kopparty, Saraf and Yekhanin [KSY11] gives constant rate locally decodable codes with locality $\mathcal{O}(n^\epsilon)$. These codes provide a drastic reduction in codeword expansion at the price of fairly high locality.

There are a number of models for the introduction of errors. Shannon’s original work [Sha48], considered errors that were introduced by a binary symmetric channel, where every bit of a codeword was independently “flipped” with some constant probability. This model is relatively weak; a significantly stronger model is Hamming’s adversarial model. In the adversarial model, the channel is viewed as an adversary who is attempting to corrupt a codeword. The channel’s only limitation is on the number of symbols it is allowed to corrupt. Shannon’s random errors and Hamming’s worst-case errors provide two extreme models, and much work has gone into designing codes that are robust against some intermediate forms of error.

We will focus on the computationally-bounded channel model proposed by Lipton [Lip94, GLD04]. In this model, like in Hamming’s model, we view the channel as an adversary who is attempting to cause a decoding error. As in Hamming’s model the channel is restricted in the number of symbols (or bits) it can corrupt, but we further restrict the channel to feasible (polynomial-time) computations. This computationally-bounded channel model has been studied in the context of classical error-correction [Lip94, GLD04, MPSW05], and locally decodable codes [OPS07, HO08].

In this work, we present a construction of locally decodable codes in the computationally-bounded channel model with constant codeword expansion and locality $\mathcal{O}(\lambda)$ where λ is the security parameter. In addition to improved locality, our results offer significant improvements over previous constructions constructions of locally decodable codes in the computationally-bounded channel

model. Our codeword expansion matches that of [HO08], but we address the two main drawbacks of that construction. Our keys are much shorter ($\mathcal{O}(\lambda)$ instead of $\mathcal{O}(n)$), and our construction requires only the existence of an IND-CPA secure cryptosystem, while their result relies on the relatively strong Φ -hiding assumption [CMS99].

1.1 Previous Work

The computationally bounded channel model was introduced by Lipton [Lip94, GLD04], where he showed how a shared key can reduce worst-case (adversarial) noise to random noise. Lipton’s construction worked as follows. The sender and receiver share a permutation $\sigma \in S_n$, and a blinding factor $r \in \{0, 1\}^n$. If **ECC** is an error-correcting code with codewords of length n , robust against random noise, then $m \mapsto \sigma(\mathbf{ECC}(m)) \oplus r$ is an encoding robust against adversarial noise. If the channel is not polynomially-bounded the sender and receiver must share $n \log n + n$ bits to communicate an n -bit codeword. If, however, the channel is polynomially-bounded, and one-way functions exist, then the sender and receiver can share a (short) seed for a pseudo-random generator rather than the large random objects σ and r .

One drawback of Lipton’s construction is that it requires the sender and receiver to share a secret key. In [MPSW05], Micali, Peikert, Sudan and Wilson considered public-key error correcting codes against a bounded channel. They observed that if $\text{Sign}(sk, \cdot)$ is an existentially unforgeable signature scheme, and **ECC** is a *list-decodable* error correcting code, then $\mathbf{ECC}(m, \text{Sign}(sk, m))$ can tolerate errors up to the list-decoding bound of **ECC** against a computationally bounded channel. The receiver needs only to list decode the corrupted codeword and choose the item in the list with a valid signature. Since the channel is computationally bounded it cannot produce valid signatures, so with all but negligible probability there will be only one message in the list with a valid signature. This technique allowed them to create codes that could decode beyond the Shannon bound.

A new twist on the code-scrambling technique was employed by Guruswami and Smith [GS10] to construct optimal rate error correcting codes against a bounded channel in the setting where the sender and receiver do not share a key (and there is no public-key infrastructure). In the Guruswami and Smith construction, the sender chooses a random permutation and blinding factor, but then embeds this “control information” into the codeword itself and sends it along with the message. The difficulty lies in designing the code so that the receiver can parse the codeword and extract the control information which then allows the receiver to recover the intended message (called the “payload information”). Their codes are not locally decodable. However, unlike those of Guruswami and Smith, our codes require setup assumptions (a public-key infrastructure) and only achieve constant (not optimal) rate.

Locally decodable codes were first studied in the computationally bounded channel model by Ostrovsky, Pandey and Sahai [OPS07]. In their work, they showed how to adapt Lipton’s code-scrambling to achieve locally decodable codes

when the sender and receiver share a secret key. Their constructions achieved constant ciphertext expansion and locality $\omega(\log^2 \lambda)$.

In [HO08], Hemenway and Ostrovsky considered locally decodable codes in the public-key setting. They used Private Information Retrieval (PIR) to implement a hidden permutation in the public-key model. Their construction achieves constant ciphertext expansion, and locality $\mathcal{O}(\lambda^2)$. Their construction suffered from two major drawbacks, first the key-size was $\mathcal{O}(n)$ since it consisted of PIR queries implementing a hidden permutation, and second the only one of their constructions to achieve constant ciphertext expansion was based on the Φ -hiding assumption [CMS99]. Prior to this work, however, these were the only locally decodable codes in the public-key model.

The work of Bhattacharyya and Chakraborty [BC11] considers locally decodable codes in the bounded channel model, but their work concerns negative results. They show that public-key locally decodable codes with *constant* locality and linear decoding algorithm must be smooth, and hence the restriction on the channel does not make the constructions easier. The codes constructed in this paper have a non-linear decoding algorithm as well as super-constant locality, so the negative results of [BC11] do not apply.

1.2 Our Contributions

We address the problem of constructing locally decodable codes in the public key computationally bounded channel model. Prior to this work, the best known constructions of locally decodable codes in the computationally bounded channel model were due to Hemenway and Ostrovsky [HO08]. While both their construction and ours yield locally decodable codes in the computationally bounded channel model with constant codeword expansion, our construction has a number of significant advantages over the previous constructions.

For security parameter λ , and messages of length n , our construction has keys that are size $\mathcal{O}(\lambda)$, while [HO08] has keys that are of size $\mathcal{O}(n)$, indeed, this is a primary drawback of their scheme. Our construction has locality $\mathcal{O}(\lambda)$, improving the locality $\mathcal{O}(\lambda^2)$ in [HO08]. The scheme of [HO08] only achieves constant codeword expansion under the Φ -hiding assumption, while our schemes require only the existence of IND-CPA secure encryption. Like [OPS07, HO08], our codes have constant ciphertext expansion and fail to decode with negligible probability.

In previous schemes, relying on a hidden permutation [Lip94, GLD04, OPS07, HO08], the permutation is fixed by the key, and thus an adversary who monitors the bits read by the decoder can efficiently corrupt future codewords.¹ In the private key setting [Lip94, GLD04, OPS07] this can be remedied by forcing the sender and receiver to keep state. Public-key schemes which rely on a hidden permutation cannot be modified in the same way to permit re-usability. Indeed, even in the case of codes without local decodability creating optimal rate codes

¹ This notion of re-usability is different than [OPS07], where they call a code re-usable if it remains secure against an adversary who sees multiple codewords, but who cannot see the read pattern of the decoder.

in the bounded channel model that do not require sender and receiver to keep state was indicated as a difficult problem in [MPSW05]²

These claims require that the message length n be greater than λ^2 , (where λ is the security parameter). This is a minor restriction, however, since the Locally Decodable Codes are aimed at settings where the messages are large databases.

As in [Lip94, GLD04, OPS07, HO08] our construction can be viewed as a permutation followed by a blinding. In these types of constructions, the difficulty is how the sender and receiver can agree on the permutation and the blinding factor. The blinding can easily be achieved by standard PKE, so the primary hurdle is how the sender and receiver can agree on the permutation. In [OPS07] the sender and receiver were assumed to have agreed on the permutation (or a seed for a pseudo-random permutation) prior to message transmission (this is the secret-key model). In [HO08], the receiver was able to hide the permutation in his public-key by publishing PIR queries for the permutation. This has the drawback that the public-key size must be linear in the length of the message. In both [OPS07, HO08], the permutation is fixed and remains the same for all messages. In this work we take a different approach, similar to that of [GS10]. The sender generates a fresh (pseudo) random permutation for each message and encodes the permutation into the message itself. Codewords consist of two portions, the control portion (which specifies the permutation) and the payload portion (which encodes the actual message).

1.3 Notation

If $f : X \rightarrow Y$ is a function, for any $Z \subset X$, we let $f(Z) = \{f(x) : x \in Z\}$. If A is a PPT machine, then we use $a \stackrel{\$}{\leftarrow} A$ to denote running the machine A and obtaining an output, where a is distributed according to the internal randomness of A . If R is a set, and no distribution is specified, we use $r \stackrel{\$}{\leftarrow} R$ to denote sampling from the uniform distribution on R . We say that a function ν is negligible if $\nu = o(n^{-c})$ for every constant c . For a string x , we use $|x|$ to denote the length (in bits) of x . For two strings $x, y \in \{0, 1\}^n$ we use $x \oplus y$ to denote coordinate-wise exclusive-or.

2 Locally Decodable Codes

In this section we define the codes and channel model we consider.

Definition 1 (Adversarial Channels). *An adversarial channel of error rate δ is a randomized map $A : \{0, 1\}^n \rightarrow \{0, 1\}^n$ such that for all w , $\text{dist}(w, A(w)) < \delta n$. We say that the channel is computationally bounded if A can be computed in time polynomial in n .*

² Our solution does not solve the problem posed in [MPSW05], however, because while our codes transmit data at a constant rate, they do not achieve the Shannon capacity of the channel.

Definition 2 (Locally Decodable Codes). A code $\mathbf{ECC} = (\mathbf{ECCEnc}, \mathbf{ECCDec})$ is called a $[q, \delta, \epsilon]$ locally decodable code with rate r if for all adversarial channels A of error rate δ we have

- For all x , and all $i \in [k]$ it holds that $\Pr[\mathbf{ECCDec}(A(x), i) = x_i] \geq 1 - \epsilon$.
- \mathbf{ECCDec} makes at most q queries to $A(x)$.
- The ratio $|x|/|\mathbf{ECCEnc}(x)| = r$.

Where x_i denotes the i th bit of x .

Simply letting A be computationally bounded in Definition 2 is not sufficient since it does not address A 's ability to see the public-key or adapt to previous read patterns.

Definition 3 (Public-Key Locally Decodable Codes).

A code $\mathbf{PKLDC} = (\mathbf{PKLDCGen}, \mathbf{PKLDCEnc}, \mathbf{PKLDCDec})$ is called a $[q, \delta, \epsilon]$ public-key locally decodable code with rate r if all polynomial time adversarial channels A of error rate δ have probability at most ϵ of winning the following game. The game consists of three consecutive phases.

1. **Key Generation Phase:**

The challenger generates $(pk, sk) \stackrel{\$}{\leftarrow} \mathbf{PKLDCGen}(1^\lambda)$, and gives pk to the adversary.

2. **Query Phase:**

The adversary can adaptively ask for encodings of messages x , and receives $c = \mathbf{PKLDCEnc}(pk, x)$. For any $i \in [n]$, the adversary can then ask for the decoding of the i th bit of x from c , and learn the q indices in c that were queried by $\mathbf{PKLDCDec}(sk, c, i)$.

3. **Challenge Phase:**

The adversary chooses a challenge message x , and receives $c = \mathbf{PKLDCEnc}(pk, x)$, the adversary outputs \tilde{c} . The adversary wins if $|\tilde{c}| = |c|$, $\text{dist}(\tilde{c}, c) \leq \delta|c|$, and there exists an $i \in [n]$ such that $\mathbf{PKLDCDec}(sk, \tilde{c}, i) \neq x_i$.

We also require that

- $\mathbf{PKLDCDec}(sk, c)$ makes at most q queries to the codeword c .
- The ratio $|x|/|\mathbf{PKLDCEnc}(pk, x)| = r$.

We will focus on the case where the error rate δ is constant, the transmission rate r is constant. If we specify that the probability ϵ of decoding error is a negligible function of the security parameter, then with these constraints our goal is to minimize the locality q .

Remark: In the query phase, we allowed the adversary to see indices read by the challenger when decoding a codeword created by the challenger itself. We could allow the adversary to see the indices read by decoding algorithm on any string c . Proving security in this more general setting could be achieved using the framework below by switching the IND-CPA encryption scheme in our construction for an IND-CCA one.

3 Construction

Let $\mathcal{PKE} = (\text{Gen}, \text{Enc}, \text{Dec})$ be a semantically secure public-key encryption, with plaintexts of length 2λ , and ciphertexts of length $2d\lambda$. Let $\mathbf{ECC}_1 = (\mathbf{ECCEnc}_1, \mathbf{ECCDec}_1)$ be an error correcting code with $2d\lambda$ bit messages and $2dd_1\lambda$ bit codewords. Let $\mathbf{ECC}_2 = (\mathbf{ECCEnc}_2, \mathbf{ECCDec}_2)$ be an error correcting code with t bit messages and d_2t bit codewords, and let PRG be a pseudo-random generator taking values in the symmetric group on d_2n symbols. Thus $\text{PRG}(\cdot) : \{0, 1\}^\lambda \rightarrow S_{d_2n}$. Let $\widetilde{\text{PRG}}$ be a pseudo-random generator from $\{0, 1\}^\lambda \rightarrow \{0, 1\}^{d_2n}$.

– **Key Generation:**

The algorithm $\mathbf{PKLDCGen}(1^\lambda)$ samples $(pk, sk) \xleftarrow{\$} \text{Gen}$. The public key will be pk along with the two function descriptions $\text{PRG}, \widetilde{\text{PRG}}$, while the secret key will be sk .

– **Encoding:**

To encode a message $m = m_1 \cdots m_n$, the algorithm $\mathbf{PKLDCEnc}$ breaks m into blocks of size t and set $c_i = \mathbf{ECCEnc}_2(m_i)$ for $i = 1, \dots, n/t$. Set $C = c_1 \cdots c_{n/t}$, so $|C| = d_2n$. Sample $x_1 \xleftarrow{\$} \{0, 1\}^\lambda$. $x_2 \xleftarrow{\$} \{0, 1\}^\lambda$, and let $\sigma = \text{PRG}(x_1)$, and $R = \widetilde{\text{PRG}}(x_2)$. Generate $r \xleftarrow{\$} \text{coins}(\text{Enc})$. The codeword will be

$$\underbrace{(\mathbf{ECCEnc}_1(\text{Enc}((x_1, x_2), r)), \dots, \mathbf{ECCEnc}_1(\text{Enc}((x_1, x_2), r)))}_{\ell \text{ copies}}, R \oplus \sigma(C).$$

So a codeword consists of ℓ copies of the “control information” $\mathbf{ECCEnc}_1(\text{Enc}((x_1, x_2), r))$, followed by the “payload information” $R \oplus \sigma(C)$.

– **Decoding:**

The algorithm $\mathbf{PKLDCDec}$ takes as input a codeword (c_1, \dots, c_ℓ, P) , and a desired block $i^* \in \{1, \dots, n/t\}$. First, the decoder must recover the control information. For j from 1 to $2dd_1\lambda$, $\mathbf{PKLDCDec}$ chooses a block $i_j \in [\ell]$, and reads the j th bit from the i_j th control block. Concatenating these bits, the decoder has (a corrupted version) of $c = \mathbf{ECCEnc}_1(\text{Enc}((x_1, x_2), r))$. The decoder decodes with \mathbf{ECCDec}_1 , and then decrypts using Dec to recover (x_1, x_2) . The control information (x_1, x_2) will be recovered correctly if no more than a δ_1 fraction of the bits $2dd_1\lambda$ bits read by the decoder were corrupted. Second, once the decoder has the control information. The decoder then recovers $\sigma = \text{PRG}(x_1)$, and $R = \widetilde{\text{PRG}}(x_2)$. The block i^* consists of the bits $i^*t, \dots, (i^* + 1)t - 1$ of the message m , so the decoder reads the bits $P_{\sigma(i^*d_2t)}, \dots, P_{\sigma((i^*+1)d_2t-1)}$ from the received codeword. The decoder then removes the blinding factor

$$C = P_{\sigma(i^*d_2t)} \oplus R_{\sigma(i^*d_2t)} \cdots P_{\sigma((i^*+1)d_2t-1)} \oplus R_{\sigma((i^*+1)d_2t-1)}$$

At this point C is a codeword from \mathbf{ECC}_2 , so the decoder simply outputs $\mathbf{ECCDec}_2(C)$. The locality is $2dd_1\lambda + d_2t$.

Remarks: The above scheme admits many modifications. In particular, there are a number of simple tradeoffs that can be made to increase the correctness of the scheme, while decreasing the locality. Tradeoffs of this sort between locality (or codeword expansion) and correctness are commonplace in coding theory, and we make no attempt to list them all here.

- **Codeword Length:** A codeword is of the form

$$\underbrace{(\mathbf{ECCEnc}_1(\text{Enc}((x_1, x_2), r)), \dots, \mathbf{ECCEnc}_1(\text{Enc}((x_1, x_2), r)))}_{2\ell dd_1 \lambda \text{ bits}}, \underbrace{R \oplus \sigma(C)}_{d_2 n \text{ bits}}.$$

Thus the total codeword length is $2\ell dd_1 \lambda + d_2 n$, making the codeword expansion $\frac{2\ell dd_1 \lambda + d_2 n}{n}$.

- **Locality:** The locality is $2dd_1 \lambda + d_2 t$. If we take $t = \mathcal{O}(\lambda)$, then we will successfully recover with all but negligible probability (negligible ϵ), and the locality will be $\mathcal{O}(\lambda)$.

Theorem 1. *The scheme $\mathbf{PKLDC} = (\mathbf{PKLDCGen}, \mathbf{PKLDCEnc}, \mathbf{PKLDCDec})$ is a public-key locally decodable code with locality $q = 2dd_1 \lambda + d_2 t$, and error rate δ , with failure probability*

$$\epsilon = \left(e^{\frac{\delta_1}{\alpha_1} - 1} / \left(\frac{\delta_1}{\alpha_1} \right)^{\frac{\delta_1}{\alpha_1}} \right)^{2\alpha_1 dd_1 \lambda} + ne^{-2 \frac{(\delta_2 - \alpha_2)^2 d_2^2 t^2 - 1}{d_2 t + 1}} + \nu(\lambda)$$

for some negligible function $\nu(\cdot)$. Where α_1, α_2 are any numbers with $0 \leq \alpha_1, \alpha_2 \leq 1$, satisfying

$$2\alpha_1 dd_1 \lambda \ell + \alpha_2 d_2 n \leq \delta |C| = 2\delta \ell dd_1 \lambda + \delta d_2 n,$$

and δ_i is the error rate tolerated by \mathbf{ECC}_i for $i \in \{1, 2\}$. In particular, this means that for all PPT algorithms A and for all i

$$\Pr \left[\begin{array}{l} \mathbf{PKLDCDec}(sk, \tilde{C}, i) \neq x_i : \\ (pk, sk) \xleftarrow{\$} \mathbf{PKLDCGen}(1^\lambda) \\ C \xleftarrow{\$} \mathbf{PKLDCEnc}(pk, x), \tilde{C} \xleftarrow{\$} A(C, pk) \end{array} \right] < \epsilon$$

whenever C and \tilde{C} have the same length, and differ in at most $\delta |C|$ bits.

Proof. Since the codewords are naturally divided into two types of information, *control information*, and *payload information*, we distinguish between errors in each type. Let ϵ^c be the event that the adversary succeeds in corrupting the control information read by the decoder, and let ϵ_i^p be the event that the adversary succeeds in corrupting payload block i . Given a corrupted codeword $\tilde{C} = (\tilde{c}_1, \dots, \tilde{c}_\ell, \tilde{P})$, ϵ^c is the event that more than a δ_1 fraction of the $2dd_1 \lambda$ control bits are corrupted, so the event ϵ^c corresponds to the event that the adversary succeeds in making the decoder recover erroneous control information. Similarly, ϵ_i^p is the event that more than a δ_2 fraction of the bits of the

payload block $P_{\sigma(id_2 t)} \cdots P_{\sigma((i+1)d_2 t-1)}$ are corrupted. Recall that δ_i is the error tolerance of \mathbf{ECC}_i , in particular, \mathbf{ECC}_i successfully decodes from a δ_i fraction of corrupted bits. It is easy to see that the probability of incorrect decoding is bounded above by $\Pr[\mathbf{e}^c] + \sum_i \Pr[\mathbf{e}_i^p]$.

In order to bound $\Pr[\mathbf{e}^c]$ and $\Pr[\mathbf{e}_i^p]$ it suffices (see the full version for details) to consider a game where:

- We imagine the challenger to be both the sender and receiver.
- When decoding, the challenger selects indices $i_1, \dots, i_{2dd_1\lambda} \in [\ell]$, and if more than δ_1 fraction of the bits specified by them are incorrect, the challenger outputs \perp , otherwise the challenger continues to read the appropriate payload blocks.
- When encoding, the challenger encodes $(0, 0)$ rather than (x_1, x_2) .
- σ and R are chosen uniformly from $S_{d_2 n}$ and $\{0, 1\}^{d_2 n}$ respectively.

To bound $\Pr[\mathbf{e}^c]$ and $\Pr[\mathbf{e}_i^p]$ in this game, suppose A introduces α_1 fraction of errors into the control information and α_2 error into the payload information. Since the adversary introduces at most a δ fraction of errors into the entire codeword, we have

$$2\alpha_1 dd_1 \lambda \ell + \alpha_2 d_2 n \leq \delta |C| = 2\delta dd_1 \lambda + \delta d_2 n$$

Recall that the control information $\mathbf{ECCEnc}_1(\mathbf{Enc}((x_1, x_2), r))$ is $2dd_1\lambda$ bits long, and there are ℓ copies of it in the codeword. Let Z_j denote the event that the j th control bit read by the decoder is corrupted, where the probability ranges over the decoder’s choice over which of the ℓ copies the bit is read from. Then each Z_j is an independent Bernoulli random variable, and $\sum_{i=1}^{2dd_1\lambda} \mathbb{E}(Z_i) = 2\alpha_1 dd_1 \lambda$. A Chernoff bound yields

$$\Pr[\mathbf{e}^c] = \Pr \left[\sum_{j=1}^{2dd_1\lambda} Z_j > 2\delta_1 dd_1 \lambda \right] < \left(e^{\frac{\delta_1}{\alpha_1} - 1} / \left(\frac{\delta_1}{\alpha_1} \right)^{\frac{\delta_1}{\alpha_1}} \right)^{2\alpha_1 dd_1 \lambda}$$

We observe that this will clearly be negligible in λ , whenever $\delta_1 > \alpha_1$, i.e. the error tolerance of \mathbf{ECC}_1 is greater than the proportion of the control information that is corrupted. By choosing ℓ to be large enough and δ to be small enough, we can always ensure that this is the case.

To analyze the probability that the adversary successfully corrupts a payload block, we observe that since σ and R are uniform, the adversary’s corruptions are distributed uniformly among the $d_2 n$ payload bits. The number of errors in a given payload block is distributed according the hypergeometric distribution with parameters $(\alpha_2 d_2 n, d_2 n, d_2 t)$.

Theorem 1 from [HS05] gives

$$\Pr[\mathbf{e}_i^p] = \Pr[\#\text{errors in block } i > \delta_2 d_2 t] < e^{-2 \frac{(\delta_2 - \alpha_2)^2 d_2^2 t^2 - 1}{d_2 t + 1}}.$$

It is easy to see that if $\delta_2 > \alpha_2$, then this drops exponentially quickly in t .

Corollary 1. *If there exists IND-CPA secure encryption with constant ciphertext expansion then for messages of length $n \geq \lambda^2/2$ there exists Public-Key Locally Decodable Codes of constant rate tolerating a constant fraction of errors with locality $q = \mathcal{O}(\lambda^2)$, and $\epsilon = \nu(\lambda)$ for some negligible function ν .*

The proof can be found in the full version.

A similar construction works to convert any Secret Key Locally Decodable Code [OPS07] to a PKLDC using only a standard IND-CPA secure cryptosystem. The details are in the full version.

4 Conclusion

In this work we showed how to design locally decodable codes in the computationally bounded channel model, achieving constant expansion and tolerating a constant fraction of errors, based on the existence of IND-CPA secure public-key encryption.

This is the first work giving public-key locally decodable codes in the bounded channel model with keys that are independent of the size of the message, and the only public-key locally decodable codes achieving constant rate based on standard assumptions.

Our constructions are also fairly efficient. The decoder must do a single decryption with an IND-CPA secure cryptosystem, two evaluations of PRGs, and then decode two standard error-correcting codes.

Our construction is easily modified to provide a transformation from any secret-key locally decodable code to a public-key one.

References

- [BC11] Bhattacharyya, R., Chakraborty, S.: Constant query locally decodable codes against a computationally bounded adversary (2011), <http://people.cs.uchicago.edu/sourav/papers/LDCbounded.pdf>
- [BFLS91] Babi, L., Fortnow, L., Levin, L., Szegedy, M.: Checking computations in polylogarithmic time. In: STOC 1991, pp. 21–31 (1991)
- [CMS99] Cachin, C., Micali, S., Stadler, M.A.: Computationally private information retrieval with polylogarithmic communication. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 402–414. Springer, Heidelberg (1999)
- [Efr09] Efremenko, K.: 3-query locally decodable codes of subexponential length. In: STOC 2009, pp. 39–44. ACM, New York (2009)
- [GLD04] Gopalan, P., Lipton, R.J., Ding, Y.Z.: Error correction against computationally bounded adversaries (2004) (manuscript)
- [GS10] Guruswami, V., Smith, A.: Codes for computationally simple channels: Explicit constructions with optimal rate. In: FOCS 2010 (2010)
- [HO08] Hemenway, B., Ostrovsky, R.: Public-key locally-decodable codes. In: Wagner, D. (ed.) CRYPTO 2008. LNCS, vol. 5157, pp. 126–143. Springer, Heidelberg (2008)
- [HS05] Hush, D., Scovel, C.: Concentration of the hypergeometric distribution. Statistics and Probability Letters 75, 127–132 (2005)

- [KSY11] Kopparty, S., Saraf, S., Yekhanin, S.: High-rate codes with sublinear-time decoding. In: STOC 2011 (2011)
- [KT00] Katz, J., Trevisan, L.: On the efficiency of local decoding procedures for error-correcting codes. In: STOC 2000: Proceedings of the 32nd Annual Symposium on the Theory of Computing, pp. 80–86 (2000)
- [Lip94] Lipton, R.J.: A new approach to information theory. In: Enjalbert, P., Mayr, E.W., Wagner, K.W. (eds.) STACS 1994. LNCS, vol. 775, pp. 699–708. Springer, Heidelberg (1994)
- [MPSW05] Micali, S., Peikert, C., Sudan, M., Wilson, D.A.: Optimal error correction against computationally bounded noise. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 1–16. Springer, Heidelberg (2005)
- [OPS07] Ostrovsky, R., Pandey, O., Sahai, A.: Private locally decodable codes. In: Arge, L., Cachin, C., Jurdziński, T., Tarlecki, A. (eds.) ICALP 2007. LNCS, vol. 4596, pp. 387–398. Springer, Heidelberg (2007)
- [PS94] Polishchuk, A., Spielman, D.: Nearly linear size holographic proofs. In: STOC 1994, pp. 194–203 (1994)
- [Sha48] Shannon, C.E.: A mathematical theory of communication. *Bell System Technical Journal* 27, 343–379, 623–656 (1948)
- [Sud92] Sudan, M.: Efficient Checking of Polynomials and Proofs and the Hardness of Approximation Problems. PhD thesis, UC Berkeley (1992)
- [Tre04] Trevisan, L.: Some applications of coding theory in computational complexity. *Quaderni di Matematica* 13, 347–424 (2004)
- [Yek10] Yekhanin, S.: Locally decodable codes. *Foundations and Trends in Theoretical Computer Science* (2010)

On Sampling from Multivariate Distributions*

Zhiyi Huang and Sampath Kannan

University of Pennsylvania, Philadelphia PA 19104, USA

{hzhayi,kannan}@cis.upenn.edu

Abstract. Let X_1, X_2, \dots, X_n be a set of random variables. Suppose that in addition to the prior distributions of these random variables we are also given *linear constraints relating them*. We ask for necessary and sufficient conditions under which we can efficiently sample the constrained distributions, find constrained marginal distributions for each of the random variables, etc. We give a tight characterization of the conditions under which this is possible. The problem is motivated by a number of scenarios where we have separate probabilistic inferences in some domain, but domain knowledge allows us to relate these inferences. When the joint prior distribution is a product distribution, the linear constraints have to be carefully chosen and are crucial in creating the lower bound instances. No such constraints are necessary if arbitrary priors are allowed.

Keywords: sampling, algorithm, complexity.

1 Introduction

Suppose we are solving a crossword puzzle, sudoku or other grid puzzle. Looking at the column in which a grid cell lies, we could come up with a random variable X (with known prior distribution) that is the value in the cell. Similarly, looking at the row in which the cell lies, we could come up with another random variable Y for the value in the cell. Of course, these are two random variables for the value in one cell, and hence we have the constraint $X = Y$. If we could come up with a set of such random variables and constraints on them and compute the constrained distributions on these random variables, we would go a long way towards solving these puzzles.

More usefully, consider the following scenarios where we make related inferences:

In bioinformatics we have programs that attempt to locate the starting position of a gene along a genome. These programs typically produce a distribution on positions. We also have programs that infer the location of other genomic elements, such as transcription factor binding sites, and again produce distributions on the possible positions of these elements. Biological domain knowledge

* This material is based upon work supported by the National Science Foundation under Grant No. 0716172 and ONR MURI Grant N000140710907.

tells us something about how far apart the gene and the binding site can be, and this can be modeled as a constraint between these random variables.

In image segmentation and processing, say of human images, the positions of limbs, torsos, heads, etc., are identified probabilistically. Anatomical constraints are then the linear constraints between these random variables, and determining their constrained distributions helps localize the image more precisely.

We model these problems as follows: X_1, X_2, \dots, X_n are a given set of random variables. We assume we know their prior joint probability distribution F . (Later in this paper we consider the special case where the random variables are originally independent and this joint distribution is a product distribution.) Without loss of generality, we assume that each random variable has a support that is a subset of $[0, 1]$. The linear constraints on the random variables define a feasible region, which is a convex body K .

Thus we will study the problem of sampling vectors in the unit hypercube $[0, 1]^n$ according to some multivariate distribution F subject to the condition that we want to sample only inside a convex body K . Here, F is a multivariate distribution of an n -dimensional random vector $\mathbf{X} = (X_1, \dots, X_n) \in [0, 1]^n$ (We will always assume that F is defined on the entire unit hypercube.) We seek to answer the following question: *What is the necessary and sufficient condition for F such that the sampling can be done in polynomial time?*

Model and Definition. We will follow the notation in [1] in general. For every vector $\mathbf{x} = (x_1, \dots, x_n) \in [0, 1]^n$, we let $F(\mathbf{x})$ denote the probability density of \mathbf{x} in distribution F . For every $i \in [n]$, we let $F_i(\cdot)$ denote the marginal density function of the i^{th} coordinate. We will let $f(\cdot)$ and $f_i(\cdot)$ denote the functions $\log F(\cdot)$ and $\log F_i(\cdot)$.

We let $K(\mathbf{x})$ denote the membership indicator function of the convex body K , that is, $K(\mathbf{x}) = 1$ if $\mathbf{x} \in K$; and $K(\mathbf{x}) = 0$ otherwise.

While the algorithms for sampling (e.g. [1]) only require oracle access to F and K , our lower bound results in this paper hold even if $F(\mathbf{x})$ and $K(\mathbf{x})$ are explicitly given in closed form and can be computed in polynomial time.

Lipschitz Condition. We will assume that $f(\mathbf{x})$ satisfies the Lipschitz condition for a polynomially large Lipschitz constant α , that is, $|f(\mathbf{x}) - f(\mathbf{x}')| \leq \alpha \|\mathbf{x} - \mathbf{x}'\|_\infty$. This parameter specifies how smooth function $f(\cdot)$ is.

Log-concavity. Previous research showed that the log-concavity of the distribution plays an important role in efficient sampling algorithms. The distribution F is β -close to log-concave if for any $\mathbf{x}, \mathbf{y} \in [0, 1]^n$, and $\lambda \in [0, 1]$, $f(\lambda \mathbf{x} + (1 - \lambda)\mathbf{y}) \geq \lambda f(\mathbf{x}) + (1 - \lambda)f(\mathbf{y}) - \beta$.

Problem Statement. Concretely, we consider the following problem $\text{SAMPLE}(\epsilon, \alpha, \beta)$: *Sample $\mathbf{X} \in \mathbb{R}^n$ from the constrained distribution $F|_K$ with error at most ϵ , where K is a convex body and F satisfies the Lipschitz condition for Lipschitz constant α and is β -close to log-concave. That is, sample \mathbf{X} from some distribution \tilde{F} , such that $\forall \mathbf{x} : \left| \tilde{F}(\mathbf{X} = \mathbf{x}) - F(\mathbf{X} = \mathbf{x} \mid \mathbf{X} \in K) \right| \leq \epsilon$.*

Related Work. Inspired by the work by Dyer, Frieze, and Kannan [4] that gave a polynomial time algorithm for estimating the volume of a convex body, Applegate and Kannan [1] proposed an efficient sampling algorithm based on fast convergence rate of a carefully chosen random walk. More concretely, they showed that there is an algorithm for $\text{SAMPLE}(\epsilon, \alpha, \beta)$ that runs in $\tilde{O}(n^3 \alpha^2 e^{2\beta} \log(\frac{1}{\epsilon}))$ time if $K = [0, 1]^n$. This is a polynomial-time algorithm on the unit cube if the distribution F is $O(\log n)$ -close to log-concave and the sampling error ϵ is at most inverse exponentially small. Their result implicitly implies that there is a polynomial time algorithm for general convex bodies K via a simple reduction as follows [6]: Let $d_K(\mathbf{x})$ denote the distance between \mathbf{x} and the closest point in the convex body K . Consider the smoothed membership function K_γ for some large enough γ (i.e. $\gamma \gg \alpha$): $K_\gamma(\mathbf{x}) = e^{-\gamma d_K(\mathbf{x})}$. The function K_γ is log-concave. So the product of functions F and K_γ has the same distance from log-concavity as F does. Moreover, sampling with density proportional to $F(\mathbf{x}) K_\gamma(\mathbf{x})$ approximates the constrained distribution $F|_K$ very well. Hence, we can use the sampling algorithm by Applegate and Kannan to handle the case of general convex bodies K via the above reduction.

There have been a series of improvements in the running time [5, 8–10]. Very recently, Chandrasekaran, Deshpande, and Vempala [3] showed that there exists a polynomial time algorithm for sampling according to harmonic concave density functions, a slightly broader family of functions than log-concave functions.

On the other hand, Koutis [7] proved that there is no polynomial time sampling algorithm if the density function is allowed to be $\tilde{\Omega}(\log^3 n)$ -far from log-concave, unless there is a $2^{o(n)}$ algorithm for the HAMILTONIAN PATH problem.

Our Results. We close the gap between the upper and lower bounds above by showing that there is no polynomial time algorithm for sampling from density functions that are $\omega(\log n)$ -far from log-concave, unless there exists a $2^{o(n)}$ algorithm for 3SAT. (Section 3.1)

Next, we turn to the more restricted case where the distribution F is a product distribution. The study of this special case is triggered by the fact that in many applications each random variable is the result of an independent experiment or algorithm. Domain knowledge relating these random variables is applied later. Note that if $K = [0, 1]^n$, then we can tolerate the product distribution F to be $O(\log n)$ -far from log-concave *on each coordinate* since we can sample each coordinate independently. The independence of random variables seems to provide more power to sampling algorithms. Surprisingly, we show that in fact independence does not help too much in the sense that for general convex bodies K , there is no polynomial time algorithm for sampling from product distributions that are $\omega(\log n \log \log n)$ -far from log-concave, unless there is a $2^{o(n)}$ algorithm for 3SAT. (Section 3.2)

2 Warm-Up: Discrete Case

It is fairly easy to show hardness if the random variables are discrete. The intuitive idea is to consider a uniform distribution on the vertices of a hypercube.

These vertices represent assignments to a 3SAT instance and we will use constraints that enforce that each clause is satisfied (i.e. consider the linear constraints in the standard LP-relaxation). We also add a special variable Z that serves as a “switch” on the constraints. More precisely, for each of the linear constraints $X_i + X_j + X_k \geq 1$, we will convert it into $X_i + X_j + X_k + Z \geq 1$. Hence, when $Z = 1$ the constraints are “off” and any vertex of the hypercube is feasible; but when $Z = 0$, the only feasible vertices correspond to satisfying assignments. Then, computing the marginal distribution of Z is equivalent to solving 3SAT. The details of the construction and the proof of its correctness are deferred to the full version.

3 Continuous Case

Now we turn to the continuous case. Instead of attacking the original sampling problem itself, we will consider the following easier problem of

INTEGRATION (δ, α, β) : Compute the integral $\int_{x \in K} F(x) dx$ up to multiplicative error δ , where F satisfies the Lipschitz condition for Lipschitz constant α and is β -close to log-concave. That is, compute an approximate integral \tilde{I} , such that $(1 + \delta)^{-1} \int_{x \in K} F(x) dx \leq \tilde{I} \leq (1 + \delta) \int_{x \in K} F(x) dx$.

Proposition 1. *If there is a poly-time algorithm for SAMPLE $((\frac{\epsilon}{24\alpha n})^n, \alpha, \beta)$, then there is a polynomial time algorithm for INTEGRATION $(\epsilon, \alpha, \beta)$.*

Proposition 1 indicates that it suffices to show our lower bound for the problem of computing the approximate integral with an $1 + \epsilon$ error for some constant ϵ . The readers are referred to [1] or the full version of this paper for the proof of this proposition.

3.1 Correlated Distributions

In this section, we will consider the general case where the distribution F could be correlated. We develop the following Theorem 1 that closes the gap in the previous results.

Theorem 1. *For any $\beta = \omega(\log n)$, there is no polynomial time algorithm for the problem INTEGRATION $(\frac{1}{30}, 2\beta n, \beta)$ even when $K = [0, 1]^n$, unless there exists a $2^{o(n)}$ algorithm for 3SAT.*

The following problem is not directly related to our problems. But we will use it as an intermediate problem in our proofs.

GAP-#3SAT $(g(n))$: *Decide whether a given 3SAT instance with n variables has no feasible assignment or has at least $2^n/n^{g(n)}$ feasible assignments.*

High-level Sketch of Our Approach. Let us first consider the following overly simplified approach. Divide the hypercube $[0, 1]^n$ into 2^n smaller hypercubes via n hyperplanes $x_i = \frac{1}{2}, i = 1, 2, \dots, n$. Each small hypercube contains exactly one integral point. We will hardwire a 3SAT instance with the function F as follows: For every $\mathbf{x} \in [0, 1]^n$, let $[\mathbf{x}]$ denote the integral point that is in the same small hypercube with \mathbf{x} (if \mathbf{x} is on the boundary of two or more small hypercubes, define $[\mathbf{x}]$ to be an arbitrary one of the corresponding integral points); let $F(\mathbf{x}) = 1$ if $[\mathbf{x}]$ is a satisfying assignment for the 3SAT instance and let $F(\mathbf{x}) = 0$ otherwise. It is easy to see that the integral value equals the number of satisfying assignments times $\frac{1}{2^n}$. However, the function F in this approach is highly discontinuous and infinitely far from log-concave. Therefore, we will proceed by considering a smoothed version of this function.

Proof (Proof of Theorem 7). In the following discussion, we assume that $g(n) = \omega(1)$ is a monotone and slowly-growing function of n . We will prove Theorem 11 via the following two lemmas.

Lemma 1. *If there is a poly-time algorithm for INTEGRATION $(\frac{1}{30}, 2\beta n, \beta)$ where $\beta = g(n) \log n$, then there is a poly-time algorithm for GAP-#3SAT $(g(n))$.*

Proof. Suppose there is a polynomial time algorithm for INTEGRATION $(\frac{1}{30}, 2\beta n, \beta)$ where $\beta = g(n) \log n$. Let I be a GAP-#3SAT $(g(n))$ instance with n variables X_1, \dots, X_n . We will demonstrate how to encode I with an instance of INTEGRATION $(\frac{1}{30}, 2\beta n, \beta)$.

We consider an n -dimensional distribution of variables X_1, \dots, X_n . For any $\mathbf{x} = (x_1, \dots, x_n) \in [0, 1]^n$, we let $[\mathbf{x}] = ([x]_1, \dots, [x]_n) \in \{0, 1\}^n$ denote the rounded version of \mathbf{x} , such that for each $1 \leq i \leq n$, we let $[x]_i = 1$ if $x_i \geq \frac{1}{2}$; let $[x]_i = 0$ if $x_i < \frac{1}{2}$.

We will let $\hat{f}(\mathbf{x})$ denote the following function:

$$\hat{f}(\mathbf{x}) = \begin{cases} 0 & , \text{ if } \|\mathbf{x} - [\mathbf{x}]\|_\infty > \frac{1}{2} - \frac{1}{2n}, \\ g(n) \log n & , \text{ if } \|\mathbf{x} - [\mathbf{x}]\|_\infty < \frac{1}{2} - \frac{1}{n}, \\ (n - 1 - 2n \|\mathbf{x} - [\mathbf{x}]\|_\infty)g(n) \log n & , \text{ otherwise.} \end{cases}$$

$\hat{f}(\mathbf{x})$ is a piecewise-linear function that satisfies the Lipschitz condition with Lipschitz constant $2n g(n) \log n$. Further, we have that $\max_{\mathbf{x}} \hat{f}(\mathbf{x}) - \min_{\mathbf{x}} \hat{f}(\mathbf{x}) = g(n) \log n$. Therefore, for every \mathbf{x}, \mathbf{y} and λ , we have $\hat{f}(\lambda \mathbf{x} + (1 - \lambda)\mathbf{y}) \geq \lambda \hat{f}(\mathbf{x}) + (1 - \lambda)\hat{f}(\mathbf{y}) - g(n) \log n$.

So if we let $\hat{F}(\mathbf{x}) = 2\hat{f}(\mathbf{x})$, then $\hat{F}(\mathbf{x})$ is $g(n) \log n$ -close to log-concave. Furthermore, the value of $\hat{F}(\mathbf{x})$ is large if \mathbf{x} is “close” to being integral, that is, $\|\mathbf{x} - [\mathbf{x}]\|_\infty < \frac{1}{2} - \frac{1}{n}$; and it is small if \mathbf{x} is far from being integral.

Next, we will hardwire the GAP-#3SAT $(g(n))$ instance I by constructing the following distribution F based on \hat{F} . Essentially, we will “iron” the function value to 1 for the areas that do not correspond to a satisfying assignment of I .

$$F(\mathbf{x}) = \begin{cases} \hat{F}(\mathbf{x}) & , \text{ if } [\mathbf{x}] \text{ is a satisfying assignment for } I; \\ 1 & , \text{ otherwise.} \end{cases}$$

It is easy to verify that $F(x)$ is also $g(n) \log n$ -close to log-concave and $\log F(x)$ satisfies Lipschitz condition with Lipschitz constant $2n g(n) \log n = 2\beta n$. On the one hand, if I is not satisfiable, then clearly $\int_{x \in [0,1]^n} F(x) dx = 1$. On the other hand, if I has at least $2^n n^{-g(n)}$ satisfying assignments, then

$$\int_{x \in [0,1]^n} F(x) dx = 1 + \int_{\mathbf{x}: I([\mathbf{x}])=1} (\hat{F}(\mathbf{x}) - 1) dx \geq 1 + (n^{g(n)} - 1) \frac{2^n}{n^{g(n)}} \left(\frac{1}{2} - \frac{1}{n}\right)^n = 1 + \left(1 - n^{-g(n)}\right) \left(1 - \frac{2}{n}\right)^n > 1 + \frac{1}{2e^2}.$$

The last inequality holds for sufficiently large n because $1 - n^{-g(n)} \rightarrow 1$ and $\left(1 - \frac{2}{n}\right)^n \rightarrow e^{-2}$ as n approaches ∞ . By our assumption, there is an polynomial time algorithm for the problem INTEGRATION $(\frac{1}{30}, 2\beta n, \beta)$ for $\beta = g(n) \log n$. We can use this algorithm to distinguish these two cases because $(1 + \frac{1}{30})^2 < 1 + \frac{1}{2e^2}$. So we can solve GAP-#3SAT($g(n)$) in polynomial time.

Lemma 2. *If there is a polynomial time algorithm for GAP-#3SAT($g(n)$), then there exists a $2^{O(n/g(n))}$ algorithm for 3SAT.*

Proof. The proof of this lemma relies on padding redundant variables to a 3SAT instance. Suppose there is a polynomial time algorithm A for GAP-#3SAT($g(n)$). Let I be a 3SAT instance with n variables X_1, \dots, X_n and m clauses C_1, \dots, C_m . We shall let I^* denote the 3SAT instance with the same set of clauses and $N - n$ redundant variables X_{n+1}, \dots, X_N , where $N = 2^{n/g(n)}$. We have that $\#3SAT(I^*) = 2^{N-n} \cdot \#3SAT(I)$.

Hence, if I is not satisfiable, then $\#3SAT(I^*) = 0$, otherwise, the number of feasible assignments for I^* is at least $\#3SAT(I^*) \geq 2^{N-n} = 2^N / N^{g(n)} \geq 2^N / N^{g(N)}$. By our assumption, we can use algorithm A to distinguish these two cases. The running time is $\text{POLY}(N) = 2^{O(n/g(n))}$.

Theorem 1 follows easily from these two lemmas.

3.2 Product Distributions

The previous reduction does not apply to product distributions, unless we allow each of the component distributions to deviate from log-concave by $\omega(\log n)$. There are two places where the previous approach heavily relies on the correlation of the distributions: First, hardwiring a 3SAT instance with the function F requires the function F to be correlated. Second, the construction of \hat{F} , the function that simulates a discrete hypercube by taking on a large value at points that are “close” to integral in all coordinates and a small value otherwise, is also highly correlated. If we construct a product distribution whose total deviation from log-concave is at most $O(\log n)$, then on average each dimension is only $O(\frac{\log n}{n})$ apart from being log-concave. So the areas with only a few fractional entries will have density that is close to that of the almost integral points, and hence contribute too large a fraction in the integration.

On the other hand, the algorithm for efficient sampling only works if the *total* deviation from log-concave is $O(\log n)$. In this section we close the gap, showing a lower bound that matches the upper bound. More precisely, we will show the following theorem.

Theorem 2. *For any $\beta = \omega(\log N \log \log N)$, there is no polynomial time algorithm that solves the problem INTEGRATION $(1, \beta n^2, \beta)$ for N -variate product distributions subject to a convex body K , unless there exists a $2^{o(n)}$ randomized algorithm for 3SAT.*

Main Idea. To get around the first obstacle mentioned above, we will simply encode the 3SAT instance with linear inequalities as we did in the discrete case. The challenge is to overcome the second obstacle. The main idea is to embed an n -dimensional hypercube into an N -dimensional hypergrid where $N > n$ such that any fractional point in the n -dimensional hypercube has a lot of fractional entries in the N -dimensional hypergrid. The purpose of such an embedding is to create a larger gap between the density of the near-integral parts and the parts where at least one of the coordinates is far from integral for a product distribution. The embedding will be in the flavor of a linear error-correcting code: In error-correcting codes the goal is to ensure that changing one bit in the original string will change many bits in the encoded string; in our embedding scheme, the goal is to ensure that one fractional coordinate in the n -dimensional hypercube will lead to many fractional coordinates of the embedded N dimensional hypergrid. Indeed, the embedding scheme used in our construction is inspired by random linear error-correcting codes. For instance, Figure 1 shows how to embed a 2-dimensional cube into a 3-dimensional grid.

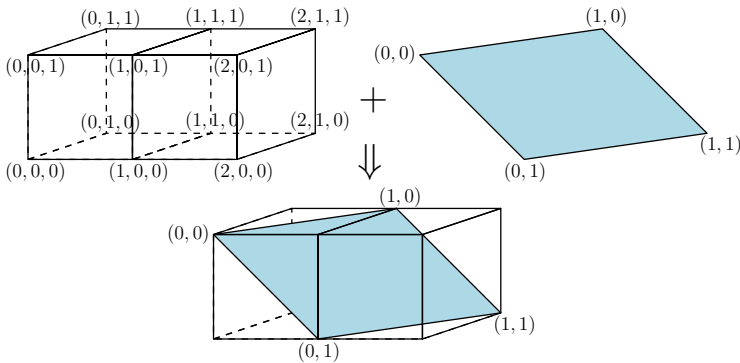


Fig. 1. Embedding 2-dimensional cube into a 3-dimensional grid

In the following discussion, we will again assume that $g(\cdot) = \omega(1)$ is a slowly growing monotone function. Suppose there is a polynomial time algorithm that tolerates $O(g(N) \log N \log \log N)$ deviation from log-concavity. Then, we will

prove Theorem 2 by showing that there is a $2^{n/g(n)}$ algorithm for 3SAT. The key technical step in our construction will be embedding an n -dimensional hypercube into a $2^{n/g(n)}$ -dimensional hypergrid. For the sake of exposition, we will omit the Lipschitz condition in the hard instance in this paper. The Lipschitz condition can be easily fixed by considering a smoothed version of our instance (e.g. a piecewise linear density function).

Some Notation. We let $N = 2^{n/g(n)}$. Hence, we have that $\log N = n/g(n)$ and $\log n = \Theta(\log \log N)$. We let $\tau = 2^{\frac{6n \log n}{N}}$. So $\tau^N = n^{6n}$. For any real number $z \in \mathbb{R}$, we let $[z]$ denote the nearest integer value to z , that is, $[z] = \lfloor z \rfloor$ if $z - \lfloor z \rfloor \leq \frac{1}{2}$; and $[z] = \lceil z \rceil$ otherwise. We let $\{z\} \stackrel{\text{def}}{=} |z - [z]|$ denote the distance between z and $[z]$.

The Basic Instance. We define the *basic instance* Π as follows. Let $Z_i \in [0, n]$, $i \in \{0, 1\}^n$, be 2^n i.i.d. random variables. Here we relax the support of the distributions from $[0, 1]$ to $[0, n]$ in order to simplify notation. It is clear that such a relaxation is without loss of generality for constructing hard instances. For each $i \in \{0, 1\}^n$, Z_i follows the distribution with density: $F_Z(Z_i = z_i) = c\tau$ if $0 \leq \{z_i\} \leq \frac{1}{10}$ and $F_Z(Z_i = z_i) = c$ otherwise. Here $c = \frac{5}{(\tau+4)n}$ is the normalization factor. Hence, points that are close to integer have larger density.

By the above definition, the distribution of each Z_i is $\log \tau = \frac{6n \log n}{N}$ -close to log-concave. So the joint distribution of any N of these Z_i 's is $6n \log n$ -close to log-concave. By definition of N , we get $6n \log n = O(g(N) \log N \log \log N)$.

Fact 1. *The joint distribution of any N of the Z_i 's is $O(g(N) \log N \log \log N)$ -close to log-concave.*

We will consider another n i.i.d. random variables $X_1, \dots, X_n \in [0, 1]$. Each X_i follows a uniform distribution on interval $[0, 1]$, i.e. $F_{\mathbf{X}}(X_i = x_i) = 1$ for any $x_i \in [0, 1]$. Let us consider the joint distribution of (\mathbf{Z}, \mathbf{X}) subject to:

$$\forall i = (i_1, \dots, i_n) \in \{0, 1\}^n : \quad Z_i = \sum_{j=1}^n i_j X_j \tag{1}$$

Note that under these constraints, the value of \mathbf{z} is uniquely determined by the value of \mathbf{x} . The joint density function subject to these constraints is proportional to $F(\mathbf{x}, \mathbf{z}) \stackrel{\text{def}}{=}} F_{\mathbf{X}}(\mathbf{x})F_Z(\mathbf{z}) = \prod_{i \in \{0,1\}^n} F_Z(z_i)$.

Properties of the Basic Instance. We will prove two properties of the basic instance. Lemma 3 and Corollary 4 state that in the n -dimensional hypercube defined by variables \mathbf{X} , the density of any *almost integral point* is relatively large. Here, almost integral points are points \mathbf{x} 's that satisfy $\{x_j\} \leq \frac{1}{n^2}$ for all

¹ For the sake of exposition, we use equality constraints in our construction of the hard instance. Strictly speaking, this integral is always 0 in N -dimensional space since it is over a body of lower dimension. But we can easily fix this by replacing each equality constraint such as $Z_i = \sum_{j=1}^n i_j X_j$ by two inequality constraints $Z_i - \sum_{j=1}^n i_j X_j \geq \delta$ and $Z_i - \sum_{j=1}^n i_j X_j \leq \delta$ for some sufficiently small δ .

$1 \leq j \leq n$. In contrast, Lemma 4 and Corollary 2 states that the density of the any fractional point is small. Fractional points are points \mathbf{x} 's such that there exists $1 \leq j \leq n$ such that $\{x_j\} \geq \frac{1}{4}$.

Lemma 3. *Suppose $\mathbf{x} \in [0, 1]^n$ satisfies that for any $1 \leq j \leq n$, $\{x_j\} \leq \frac{1}{n^2}$. Then, for each $i \in \{0, 1\}^n$, the corresponding $\{z_i\} \leq \frac{1}{10}$.*

Proof. Since $\{x_j\} \leq \frac{1}{n^2}$ for all $j \in [n]$, for any $i = (i_1, \dots, i_n) \in \{0, 1\}^n$, we have that $\{z_i\} = \left\{ \sum_{j=1}^n i_j x_j \right\} \leq \sum_{j=1}^n \{i_j x_j\} \leq \sum_{j=1}^n \frac{1}{n^2} = \frac{1}{n} < \frac{1}{10}$.

By Lemma 3 and the definition of F_Z , we have the the following corollary.

Corollary 1. *Suppose $\mathbf{x} \in [0, 1]^n$ satisfies that for any $1 \leq j \leq n$, $\{x_j\} \leq \frac{1}{n^2}$. Then $\forall i \in \{0, 1\}^n : F_Z(Z_i = z_i) = c\tau$. So $F(\mathbf{x}, \mathbf{z}) = (c\tau)^{2^n}$.*

Now we move to the second property.

Lemma 4. *Suppose $\mathbf{x} \in [0, 1]^n$ satisfies that there exists $1 \leq j \leq n$, $\{x_j\} \geq \frac{1}{4}$. Then, for at least half of $i \in \{0, 1\}^n$, the corresponding z_i satisfies $\{z_i\} \geq \frac{1}{8}$.*

Proof. For any $i_{-j} \in \{0, 1\}^{n-1}$, let $i = (i_{-j}, i_j = 0)$ and $i' = (i_{-j}, i_j = 1)$. By constraint (II), we have $z_{i'} - z_i = x_j$. So we get that $\{z_{i'}\} + \{z_i\} \geq \{x_j\} \geq \frac{1}{4}$. Hence, either $\{z_i\} \geq \frac{1}{8}$ or $\{z_{i'}\} \geq \frac{1}{8}$. Therefore, we can pair up the variables z_i 's such that in each pair at least one variable lies in the interval $[\frac{1}{8}, \frac{7}{8}]$. This finishes our proof.

By Lemma 4 and definition of the F_Z , we get the following.

Corollary 2. *Suppose $\mathbf{x} \in [0, 1]^n$ satisfies that there exists $1 \leq j \leq n$, $\{x_j\} \geq \frac{1}{4}$. Then, for at least half of $i \in \{0, 1\}^n$, we have $F_Z(Z_i = z_i) = c$ and hence $F(\mathbf{x}, \mathbf{z}) \leq c^{2^{n-1}} (c\tau)^{2^{n-1}}$.*

The Random Basic Sub-instance. We want to embed the n -dimensional hypercube into an $N = 2^{n/g(n)}$ -dimensional hypergrid. So we cannot afford to use all of the Z_i 's in our hard instance construction. Instead, we will use N randomly and independently chosen Z_i 's and show that properties similar to Corollary 1 and Corollary 2 hold with high probability.

A random basic sub-instance is constructed as follows. We will consider N i.i.d. random variables $\hat{Z}_1, \dots, \hat{Z}_N$ each of which follows the density function F_Z . Moreover, for each $1 \leq k \leq N$, we randomly choose $i \in \{0, 1\}^n$ and impose a constraint between \hat{Z}_k and \mathbf{X} that is the same as the constraint between Z_i and \mathbf{X} (See (II)), i.e. $\sum_{j=1}^n i_j X_j = \hat{Z}_k$. The joint density function is proportional to $\hat{F}(\mathbf{x}, \hat{\mathbf{z}}) \stackrel{def}{=} F_{\mathbf{X}}(\mathbf{x}) F_Z(\hat{\mathbf{z}}) = \prod_{k=1}^N F_Z(\hat{z}_k)$.

Properties of the Random Basic Sub-instance. We now show that with high probability a random basic sub-instance has some good properties: The almost integral points have large density; the fractional points have small density. The former can be formalized as the Lemma 5, which follows directly from Lemma 3.

Lemma 5. *Suppose $\mathbf{x} \in [0, 1]^n$ satisfies that for any $1 \leq j \leq n$, $\{x_j\} \leq \frac{1}{n^2}$. Then, for any $1 \leq i \leq N$, the corresponding \hat{z}_i satisfies that $\{\hat{z}_i\} \leq 1/10$. Hence, for every $1 \leq i \leq N$, $F_{\mathbf{Z}}(\hat{Z}_i = \hat{z}_i) = c\tau$, and $\hat{F}(\mathbf{x}, \hat{\mathbf{z}}) = (c\tau)^N$.*

The other property is less trivial. We first state it formally as follows.

Lemma 6. *With high probability, the following holds: If $\mathbf{x} \in [0, 1]^n$ satisfies that there exists $1 \leq j \leq n$ such that $\{x_j\} \geq \frac{1}{4}$, then at least a third of the \hat{z}_k , $1 \leq k \leq N$, satisfy $\{\hat{z}_k\} \geq \frac{1}{10}$. Hence, $F_{\mathbf{Z}}(\hat{Z}_k = \hat{z}_k) = c$, $\hat{F}(\mathbf{x}, \hat{\mathbf{z}}) \leq c^{\frac{N}{3}}(c\tau)^{\frac{2N}{3}}$.*

Proof. We divide the n -dimensional unit hypercube defined by \mathbf{x} into $M = (4n^2)^n$ small hypercubes with edge length $\frac{1}{4n^2}$. Let $\mathbf{x}^1, \dots, \mathbf{x}^M$ denote the centers of these small hypercubes. Further, we let $\hat{\mathbf{z}}^1, \dots, \hat{\mathbf{z}}^M$ be the corresponding $\hat{\mathbf{Z}}$ variables. We say a cube is fractional if its center is fractional.

We will first show the following claim.

Claim. With high probability, for every $1 \leq \ell \leq M$ such that \mathbf{x}^ℓ is fractional, at least a $\frac{1}{3}$ fraction of the \hat{z}_k^ℓ 's satisfies that $\{\hat{z}_k^\ell\} \geq \frac{1}{8}$.

Proof (Proof of Claim 3.2). Let Y_k^ℓ be the indicator of whether $\{\hat{z}_k^\ell\} \geq \frac{1}{8}$. Then, by Lemma 4 each \hat{z}_k^ℓ , $1 \leq k \leq N$, satisfies that $\{\hat{z}_k^\ell\} \geq \frac{1}{8}$ with probability at least half. So we have $\mathbf{E}[Y_k^\ell] \geq \frac{1}{2}$ and hence $\mathbf{E}\left[\sum_{k=1}^N Y_k^\ell\right] = \frac{N}{2}$. Furthermore, since Y_k^ℓ 's are binary variables, we get that $\sigma[Y_k^\ell] \leq 1$ and $\sigma\left[\sum_{k=1}^N Y_k^\ell\right] \leq \sqrt{N}$. By Chernoff-Hoeffding bound, for all ℓ such that \mathbf{x}^ℓ is fractional, the probability of $\sum_{k=1}^N Y_k^\ell < \frac{N}{3} \leq \mathbf{E}\left[\sum_{k=1}^N Y_k^\ell\right] - \frac{\sqrt{N}}{6}\sigma\left[\sum_{k=1}^N Y_k^\ell\right]$ is at most $2^{-\left(\frac{\sqrt{N}}{6}\right)^2/2} \ll \frac{1}{M}$. So by union bound, we get that with high probability, $\sum_{k=1}^N Y_k^\ell \geq \frac{N}{3}$ for $1 \leq \ell \leq M$.

Now we are ready to prove Lemma 6. We will show that if the property in Claim 3.2 holds, then the property in Lemma 6 holds.

Suppose we break ties on the boundary of small hypercubes in a manner that the boundary points are allocated to the most fractional hypercube adjacent to it. Then, we can easily verify the following fact holds.

Fact 2. *If a point is fractional, then it lies in a fractional small hypercube.*

For any fractional point \mathbf{x} , let $\hat{\mathbf{z}}$ be the corresponding $\hat{\mathbf{Z}}$ variables. Suppose it lies in a small cube centered at \mathbf{x}^ℓ . Then, by Fact 2 we know \mathbf{x}^ℓ is fractional. By our assumption that the property in Claim 3.2 holds, we get that at least a $\frac{1}{3}$ fraction of the corresponding \hat{z}_k^ℓ 's satisfies $\{\hat{z}_k^\ell\} \geq \frac{1}{8}$. Note that $\{\hat{z}_k^\ell - \hat{z}_k\} \leq \sum_{j=1}^n \{x_j^\ell - x_j\} \leq \sum_{j=1}^n \frac{1}{n^2} = \frac{1}{n} < \frac{1}{40}$. So, we get that $\{\hat{z}_k\} \geq \{\hat{z}_k^\ell\} - \{\hat{z}_k^\ell - \hat{z}_k\} \geq \frac{1}{8} - \frac{1}{40} = \frac{1}{10}$.

Therefore, with high probability, for every fractional point \mathbf{x} , at least a third of the corresponding \hat{z}_k 's satisfies $\{\hat{z}_k\} \geq \frac{1}{10}$. Hence, $F_{\mathbf{Z}}(\hat{Z}_k = \hat{z}_k) = c$, and $\hat{F}(\mathbf{x}, \hat{\mathbf{z}}) \leq c^{\frac{N}{3}}(c\tau)^{\frac{2N}{3}}$.

Construction of the Hard Instance. We will consider a random basic sub-instance such that Lemma 5 and Lemma 6 holds. By the relation between sampling and integration, we only need to show that estimating the integral of f subject to arbitrary convex constraint K is computationally hard, subject to our complexity assumption. We will prove this by encoding an arbitrary 3SAT instance by adding carefully chosen linear inequality constraints in the random basic sub-instance.

Consider any 3SAT instance with n variables. We will abuse notation by letting X_1, \dots, X_n denote not only the random variables in the random basic sub-instance but the n variables in the 3SAT instance as well. We will see that they are indeed closely related to each other in our construction. For each constraint $L_1 \vee L_2 \vee L_3$, where L_k 's are literals of the form X_j or $\neg X_j$, we add a linear inequality constraint $L_1 + L_2 + L_3 \geq \frac{3}{4}$. Here, for any literal L_k of the form X_j , we will replace L_k by X_j in this inequality constraint; and for any literal L_k of the form $\neg X_j$, we replace L_k by $(1 - X_j)$.

Proof Outline. For this instance, we can prove that if we can estimate the integral, then we solve the 3SAT instance. It suffices to show that the values of the integral in these two cases are well separated. We will proceed in three steps: First, we show that the contribution of the fractional part in the integral is relatively small; next, we will show that any of the 2^n integral parts has contribution comparable to that of the fractional part; finally, we will prove that an integer part is not excluded by the above constraints if and only if it corresponds to a feasible assignment. Therefore, the 3SAT instance is satisfiable if and only if the integral is much larger than the contribution solely by the fractional part. This will finish the proof of Theorem 2.

Proof (Proof of Theorem 2). Now we are ready to proof Theorem 2. Let us proceed to the first step. We will let \mathcal{F} and \mathcal{I}_i , $1 \leq i \leq 2^n$, denote the set of fractional and integer parts, that is, $\mathcal{F} = \{\mathbf{x} \in [0, 1]^n : \exists j, \{x_j\} \geq \frac{1}{4}\}$, and $\mathcal{I}_i = \{\mathbf{x} \in [0, 1]^n : \forall j, \{x_j\} < \frac{1}{4}, [x_j] = i_j\}$ for every $i(i_1, \dots, i_n) \in \{0, 1\}^n$.

Lemma 7. $\int_{\mathbf{x} \in \mathcal{F}} F(\mathbf{x}) d\mathbf{x} \leq c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}}$.

Proof. By Lemma 6, we get that for any $\mathbf{x} \in \mathcal{F}$, $F(\mathbf{x}) \leq c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}}$. Hence, we have that $\int_{\mathbf{x} \in \mathcal{F}} F(\mathbf{x}) d\mathbf{x} \leq \int_{\mathbf{x} \in \mathcal{F}} c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}} d\mathbf{x} \leq \int_{\mathbf{x} \in [0,1]^n} c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}} d\mathbf{x} = c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}}$.

Now we turn to the contribution of any integer part \mathcal{I}_i .

Lemma 8. For any $i \in \{0, 1\}^n$, we have $\int_{\mathbf{x} \in \mathcal{I}_i} F(\mathbf{x}) d\mathbf{x} \geq c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}}$.

Proof. We will let $\mathcal{I}_i^* \subseteq \mathcal{I}_i$ denote the almost integral part in \mathcal{I}_i , that is, $\mathcal{I}_i^* = \{\mathbf{x} \in [0, 1]^n : \forall j, \{x_j\} \leq \frac{1}{n^2}, [x_j] = i_j\}$. By Lemma 5, we have that for any $\mathbf{x} \in \mathcal{I}_i^*$, $F(\mathbf{x}) \geq (\tau)^N = \tau^{\frac{N}{3}} c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}} = n^{2n} c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}}$. Thus, we have $\int_{\mathbf{x} \in \mathcal{I}_i} F(\mathbf{x}) d\mathbf{x} \geq \int_{\mathbf{x} \in \mathcal{I}_i^*} F(\mathbf{x}) d\mathbf{x} \geq \int_{\mathbf{x} \in \mathcal{I}_i^*} n^{2n} c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}} d\mathbf{x} = c^{\frac{N}{3}} (\tau)^{\frac{2N}{3}}$.

Finally we will prove Lemma 9 which states that any integer point subject to the above constraints corresponds to a feasible assignment for the 3SAT instance.

Lemma 9. *For any $\mathbf{x} \in K$ such that $\{x_j\} < 1/4$ for all $1 \leq j \leq n$, we have that $[x_1], \dots, [x_n]$ is a feasible assignment for the 3SAT instance.*

Proof. For any constraint $L_1 \vee L_2 \vee L_3$, we have that $L_1 + L_2 + L_3 \geq 3/4$. So at least one of the L_k satisfies $L_k \geq 1/4$. But $\{x_j\} < 1/4$, so the value of $[x_j]$ will satisfy this constraint. Since this is true for any constraint, we get that $[x_1], \dots, [x_n]$ is a feasible assignment.

As we discuss earlier, Lemma 7, Lemma 8, and Lemma 9 prove Theorem 2.

4 Conclusions and Open Problems

While we have shown that discrete distributions are hard in general, for the discrete distributions that arise in the motivating applications in computational biology and image processing, it looks possible to fit a log-concave continuous distribution and use this to find the constrained distributions efficiently. On the other hand, the discrete distributions that arise in the puzzle-solving applications seem difficult to fit to nice continuous distributions.

We leave open the question of a deterministic construction showing the hardness of sampling from product distributions whose total deviation from log-concave is $\omega(\log n)$.

References

1. Applegate, D., Kannan, R.: Sampling and integration of near log-concave functions. In: Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, pp. 156–163. ACM, New York (1991)
2. Chandrasekaran, K., Dadush, D., Vempala, S.: Thin partitions: Isoperimetric inequalities and a sampling algorithm for star shaped bodies. In: Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1630–1645. Society for Industrial and Applied Mathematics, Philadelphia (2010)
3. Chandrasekaran, K., Deshpande, A., Vempala, S.: Sampling s-concave functions: The limit of convexity based isoperimetry. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX 2009. LNCS, vol. 5687, pp. 420–433. Springer, Heidelberg (2009)
4. Dyer, M., Frieze, A., Kannan, R.: A random polynomial-time algorithm for approximating the volume of convex bodies. *Journal of the ACM* 38(1), 1–17 (1991)
5. Frieze, A., Kannan, R., Polson, N.: Sampling from log-concave distributions. *The Annals of Applied Probability* 4(3), 812–837 (1994)
6. Kannan, R.: Personal communication (2011)
7. Koutis, I.: On the complexity of approximate multivariate integration. Tech. rep., Carnegie Mellon University (2005)
8. Lovasz, L., Vempala, S.: Fast algorithms for log-concave functions: Sampling, rounding, integration and optimization. In: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, pp. 57–68. IEEE, Los Alamitos (2006)
9. Lovasz, L., Vempala, S.: Hit-and-run from a corner. *SIAM Journal on Computing* 35(4), 985–1005 (2006)
10. Lovasz, L., Vempala, S.: The geometry of log-concave functions and sampling algorithms. *Random Structures and Algorithms* 30(3), 307–358 (2007)

Almost Optimal Explicit Johnson-Lindenstrauss Families

Daniel Kane¹, Raghu Meka², and Jelani Nelson³

¹ Harvard University, Cambridge, USA
`dankane@math.harvard.edu`

² University of Texas at Austin, USA
`raghu@cs.utexas.edu`

³ MIT, Cambridge, USA
`minilek@mit.edu`

Abstract. The Johnson-Lindenstrauss lemma is a fundamental result in probability with several applications in the design and analysis of algorithms. Constructions of linear embeddings satisfying the Johnson-Lindenstrauss property necessarily involve randomness and much attention has been given to obtain explicit constructions minimizing the number of random bits used. In this work we give explicit constructions with an almost optimal use of randomness: For $0 < \varepsilon, \delta < 1/2$, we obtain explicit generators $G : \{0, 1\}^r \rightarrow \mathbb{R}^{s \times d}$ for $s = O(\log(1/\delta)/\varepsilon^2)$ such that for all d -dimensional vectors w of norm one,

$$\Pr_{y \in_u \{0,1\}^r} [|\|G(y)w\|^2 - 1| > \varepsilon] \leq \delta,$$

with seed-length $r = O\left(\log d + \log(1/\delta) \cdot \log\left(\frac{\log(1/\delta)}{\varepsilon}\right)\right)$. In particular, for $\delta = 1/\text{poly}(d)$ and fixed $\varepsilon > 0$, we obtain seed-length $O((\log d)(\log \log d))$. Previous constructions required $\Omega(\log^2 d)$ random bits to obtain polynomially small error.

We also give a new elementary proof of the optimality of the JL lemma showing a lower bound of $\Omega(\log(1/\delta)/\varepsilon^2)$ on the embedding dimension. Previously, Jayram and Woodruff [10] used communication complexity techniques to show a similar bound.

1 Introduction

The celebrated Johnson-Lindenstrauss lemma (JLL) [9] is by now a standard technique for handling high dimensional data. Among its many known variants (see [4], [6], [8], [13]), we use the following version originally proven in [2], [4].

Theorem 1. For all $w \in \mathbb{R}^d$, $\|w\| = 1$, $0 < \varepsilon < 1/2$, $s \geq 1$,

$$\Pr_{S \in_u \{1, -1\}^{s \times d}} [|\|(1/\sqrt{s})Sw\|^2 - 1| \geq \varepsilon] \leq C \cdot e^{-C'\varepsilon^2 s}.$$

¹ Throughout, C, C' denote universal constants. For a multiset S , $x \in_u S$ denotes a uniformly random element of S . For $w \in \mathbb{R}^d$, $\|w\|$ denotes the Euclidean norm of w .

We say a family of random matrices has the JL property (or is a JL family) if the above condition holds. In typical applications of JLL, the error δ is taken to be $1/\text{poly}(d)$ and the goal is to embed a given set of $\text{poly}(d)$ points in d dimensions to $O(\log d)$ dimensions with distortion at most $1 + \varepsilon$ for a fixed constant ε . This is the setting we concern ourselves with.

Linear embeddings of Euclidean space as above necessarily require randomness as else one can take the vector w to be in the kernel of the fixed transformation. To formalize this we use the following definition.

Definition 1. For $\varepsilon, \delta > 0$, a generator $G : \{0, 1\}^r \rightarrow \mathbb{R}^{s \times d}$ is a $(d, s, \delta, \varepsilon)$ -JL generator of seed-length r if for every $w \in \mathbb{R}^d, \|w\| = 1$,

$$\Pr_{y \in_u \{0,1\}^r} [| \|G(y)w\|^2 - 1 | \geq \varepsilon] \leq \delta.$$

1.1 Derandomizing JLL

A simple probabilistic argument shows that there exists a $(d, O(\log(1/\delta)/\varepsilon^2), \delta, \varepsilon)$ -JL generator with seed-length $r = O(\log d + \log(1/\delta))$. On the other hand, despite much attention the best known explicit generators have seed-length at least $\min(\Omega(\log(1/\delta) \log d), \Omega(\log d + \log^2(1/\delta)))$ [5], [11]. Besides being a natural problem in geometry as well as derandomization, an explicit JL generator with minimal randomness would likely help derandomize other geometric algorithms and metric embedding constructions. Further, having an explicit construction is of fundamental importance for streaming algorithms as storing the entire matrix (as opposed to the randomness required to generate the matrix) is often too expensive in the streaming context.

Our main result is an explicit generator that takes roughly $O(\log d(\log \log d))$ random bits and outputs a matrix $A \in \mathbb{R}^{s \times d}$ satisfying the JL property for constant ε and $\delta = 1/\text{poly}(d)$.

Theorem 2 (Main). For every $0 < \varepsilon, \delta < 1/2$, there exists an explicit $(d, C \log(1/\delta)/\varepsilon^2, \delta, \varepsilon)$ -JL generator $G : \{0, 1\}^r \rightarrow \mathbb{R}^{s \times d}$ with seed-length

$$r = O \left(\log d + \log(1/\delta) \cdot \log \left(\frac{\log(1/\delta)}{\varepsilon} \right) \right).$$

We give two different constructions. Our constructions are elementary in nature using only standard tools in derandomization such as k -wise independence and oblivious samplers [15]. Our first construction is simpler and gives a generic template for derandomizing most known JL families. The second construction has the advantage of allowing fast matrix-vector multiplications: the matrix-vector product $G(y)w$ can be computed efficiently in time $O(d \log d + \text{poly}(\log(1/\delta)/\varepsilon))$ [2].

Further, as one of the motivations for derandomizing JLL is its potential applications in streaming, it is important that the entries of the generated matrices be

² The computational efficiency does not follow directly from the dimensions of $G(y)$, as our construction involves composing matrices of much higher dimension.

computable in small space. We observe that for any $i \in [s], j \in [d], y \in \{0, 1\}^r$, the entry $G(y)_{ij}$ can be computed in space $O(\log d \cdot \text{poly}(\log \log d))$ and time $O(d^{1+o(1)})$ (for fixed $\varepsilon, \delta > 1/\text{poly}(d)$). (See proof of [Theorem 8](#) for the exact bound)

1.2 Optimality of JLL

We also give a new proof of the optimality of the JL lemma showing a lower-bound of $s_{\text{opt}} = \Omega(\log(1/\delta)/\varepsilon^2)$ for the target dimension. Previously, Jayram and Woodruff [\[10\]](#) used communication complexity techniques to show a similar bound in the case $s_{\text{opt}} < d^{1-\gamma}$ for some fixed constant $\gamma > 0$. In contrast, our argument is more direct in nature and is based on linear algebra and elementary properties of the uniform distribution on the sphere, and only requires the assumption $s_{\text{opt}} < d/2$. Note the JLL is only interesting for $s_{\text{opt}} < d$.

Theorem 3. *There exists a universal constant $c > 0$, such that for any distribution \mathcal{A} over linear transformations from \mathbb{R}^d to \mathbb{R}^s with $s < d/2$, there exists a vector $w \in \mathbb{R}^d, \|w\| = 1$, such that $\Pr_{S \sim \mathcal{A}}[\|Sw\|^2 - 1 > \varepsilon] \geq \exp(-c(s\varepsilon^2 + 1))$.*

1.3 Related Work

The ℓ_2 streaming sketch of Alon et al. [\[3\]](#) implies an explicit distribution over ℓ_2 -embeddings with seed-length $O(\log d)$ for embedding \mathbb{R}^d into \mathbb{R}^s with distortion $1 + \varepsilon$ and error δ , where $s = O(1/(\varepsilon^2 \delta))$. Karnin et al. [\[11\]](#) construct an explicit JL family with optimal target dimension and seed-length $(1+o(1)) \log d + O(\log^2(1/(\varepsilon \delta)))$. Clarkson and Woodruff [\[5\]](#) showed that a random scaled Bernoulli matrix with $O(\log(1/\delta))$ -wise independent entries satisfies the JL lemma, giving seed-length $O(\log(1/\delta) \log d)$. We make use of their result in our construction.

We also note that there are efficient non-black box derandomizations of JLL, [\[7\], \[14\]](#). These works take as input n points in \mathbb{R}^d , and deterministically compute an embedding (that depends on the input set) into $\mathbb{R}^{O(\log n)/\varepsilon^2}$ which preserves all pairwise distances between the given set of n points.

1.4 Outline of Constructions

For intuition, suppose that $\delta > 1/d^c$ is polynomially small and ε is a constant. Our constructions are based on a simple iterative scheme: We reduce the dimension from d to $\tilde{O}(\sqrt{d})$ (we say $f = \tilde{O}(g)$ if $f = O(g \cdot \text{polylog}(g))$.) and iterate for $O(\log \log d)$ steps.

Generic Construction. Our first construction gives a generic template for reducing the randomness required in standard JL families and is based on the following simple observation. Starting with any JL family, such as the random Bernoulli construction of [Theorem 1](#), there is a trade-off that we can make between the amount of independence required to generate the matrix and the final

embedding dimension. For instance, if we only desire to embed to a dimension of $\tilde{O}(\sqrt{d})$ (as opposed to $O(\log d)$), it suffices for the entries of the random Bernoulli matrix to be $O(1)$ -wise independent. We exploit this idea by iteratively decreasing the dimension from d to $\tilde{O}(\sqrt{d})$ and so on by using a random Bernoulli matrix with an increasing amount of independence at each iteration.

Fast JL Construction. Fix a vector $w \in \mathbb{R}^d$ with $\|w\| = 1$ and suppose $\delta = 1/\text{poly}(d)$. We first use an idea of Ailon and Chazelle [1] who give a family of unitary transformations \mathcal{R} from \mathbb{R}^d to \mathbb{R}^d such that for every $w \in \mathbb{R}^d$ and $V \in_u \mathcal{R}$, the vector Vw is *regular*, in the sense that $\|Vw\|_\infty = O(\sqrt{(\log d)/d})$, with high probability. We derandomize their construction using limited independence to get a family of rotations \mathcal{R} such that for $V \in_u \mathcal{R}$, $\|Vw\|_\infty = O(d^{-(1/2-\alpha)})$ with high probability, for a sufficiently small constant $\alpha > 0$.

We next observe that for a vector $w \in \mathbb{R}^d$, with $\|w\|_\infty = O(d^{-(1/2-\alpha)}\|w\|_2)$ projecting onto a random set of $O(d^{2\alpha} \log(1/\delta)/\varepsilon^2)$ coordinates preserves the ℓ_2 norm with distortion at most ε with high probability. We then note that the random set of coordinates can be chosen using oblivious samplers as in [15]. The idea of using samplers is due to Karnin et al. [11] who use samplers for a similar purpose.

Finally, iterating the above scheme $O(\log \log d)$ times we obtain an embedding of \mathbb{R}^d to $\mathbb{R}^{\text{poly}(\log d)}$ using $O(\log d \log \log d)$ random bits. We then apply the result of Clarkson and Woodruff [5] and perform the final embedding into $O(\log(1/\delta)/\varepsilon^2)$ dimensions by using a random scaled Bernoulli matrix with $O(\log(1/\delta))$ -wise independent entries.

As all of the matrices involved in the construction are either Hadamard matrices or projection operators, the final embedding can actually be computed in $O(d \log d + \text{poly}(\log(1/\delta)/\varepsilon))$ time.

Outline of Lowerbound. To show a lowerbound on the embedding dimension s , we use Yao’s min-max principle to first transform the problem to that of finding a hard distribution on \mathbb{R}^d , such that no single linear transformation can embed a random vector drawn from the distribution well with very high probability. We then show that the uniform distribution over the d -dimensional sphere is one such hard distribution. The proof of the last fact involves elementary linear algebra and some direct calculations.

2 Preliminaries

We first state the classical Khintchine-Kahane inequalities (cf. [12]) which give tight moment bounds for linear forms.

Lemma 1 (Khintchine-Kahane). *For every $w \in \mathbb{R}^n$, $x \in_u \{1, -1\}^n$, $k > 0$,*

$$\mathbb{E}[|\langle w, x \rangle|^k] \leq k^{k/2} \mathbb{E}[|\langle w, x \rangle|^2]^{k/2} = k^{k/2} \|w\|^k.$$

We use randomness efficient oblivious samplers due to Zuckerman [15] (See Theorem 3.17 and the remark following the theorem in [15]).

Theorem 4 (Zuckerman [15]). *There exists a constant C such that for every $\varepsilon, \delta > 0$ there exists an explicit collection of subsets of $[d]$, $\mathcal{S}(d, \varepsilon, \delta)$, with each $S \in \mathcal{S}$ of cardinality $|S| = s(\varepsilon, \delta, d) = ((\log d + \log(1/\delta))/\varepsilon)^C$, such that for every function $f : [d] \rightarrow [0, 1]$,*

$$\Pr_{S \in_u \mathcal{S}} \left[\left| \frac{1}{s} \sum_{i \in S} f(i) - \mathbb{E}_{i \in_u [d]} f(i) \right| > \varepsilon \right] \leq \delta,$$

and there exists an NC algorithm that generates random elements of \mathcal{S} using $O(\log d + \log(1/\delta))$ random bits.

Corollary 1. *There exists a constant C such that for every $\varepsilon, \delta, B > 0$ there exists an explicit collection of subsets of $[d]$, $\mathcal{S}(d, B, \varepsilon, \delta)$, with each $S \in \mathcal{S}$ of cardinality $|S| = s(d, B, \varepsilon, \delta) = ((\log d + \log(1/\delta))B/\varepsilon)^C$, such that for every function $f : [d] \rightarrow [0, B]$,*

$$\Pr_{S \in_u \mathcal{S}} \left[\left| \frac{1}{s} \sum_{i \in S} f(i) - \mathbb{E}_{i \in_u [d]} f(i) \right| > \varepsilon \right] \leq \delta,$$

and there exists an NC algorithm that generates random elements of \mathcal{S} using $O(\log d + \log(1/\delta))$ random bits.

Proof. Apply the above theorem to $\bar{f} : [d] \rightarrow [0, 1]$ defined by $\bar{f}(i) = f(i)/B$. ■

Let $H_d \in \{-1/\sqrt{d}, 1/\sqrt{d}\}^{d \times d}$ be the normalized Hadamard matrix such that $H_d^T H_d = I_d$ (we drop the suffix d when dimension is clear from context). While the Hadamard matrix is known to exist for powers of 2, for clarity, we ignore this technicality and assume that it exists for all d . Finally, let \mathcal{S}^{d-1} denote the Euclidean sphere $\{w : w \in \mathbb{R}^d, \|w\| = 1\}$.

The following definitions will be useful in giving an abstract description of our constructions.

Definition 2. *A distribution \mathcal{D} over $\mathbb{R}^{s \times d}$ is said to be a $(d, s, \delta, \varepsilon)$ -JL distribution if for any $w \in \mathcal{S}^{d-1}$, $\Pr_{S \sim \mathcal{D}} [\|Sw\|^2 - 1] > \varepsilon] < \delta$.*

Definition 3. *A distribution \mathcal{D} over $\mathbb{R}^{s \times d}$ is said to have the $(d, s, t, \delta, \varepsilon)$ -JL moment property if for any $w \in \mathcal{S}^{d-1}$, $\mathbf{E}_{S \sim \mathcal{D}} [\|Sw\|^2 - 1]^t] < \varepsilon^t \cdot \delta$.*

Definition 4. *A distribution \mathcal{D} is called a strong (d, s) -JL distribution if it is a $(d, s, \exp(-\Omega(\min\{\varepsilon, \varepsilon^2\} \cdot s)), \varepsilon)$ -JL distribution for all $\varepsilon > 0$. If \mathcal{D} has the $(d, s, \ell, O(\max\{\sqrt{\ell}/(\varepsilon^2 s), \ell/(\varepsilon s)\})^\ell, \varepsilon)$ -JL moment property for all $\varepsilon > 0$ and integer $\ell \geq 2$, then we say \mathcal{D} has the strong (d, s) -JL moment property.*

Theorem 1 shows the conditions for being a strong (d, s) -JL distribution are met by random Bernoulli matrices when $0 < \varepsilon \leq 1$, though in fact the conditions are also met for all $\varepsilon > 0$ (see the proof in [5] for example). Sometimes we omit the d, s terms in the notation above if these quantities are clear from context, or if it is not important to specify them.

Throughout, we let logarithms be base-2 and often assume various quantities, like $1/\varepsilon$ or $1/\delta$, are powers of 2; this is without loss of generality.

3 Strong JL Distributions

It is not hard to show that having the strong JL moment property and being a strong JL distribution are equivalent. We use the following standard fact.

Fact 5. *Let Y, Z be nonnegative random variables such that $\Pr[Z \geq t] = O(\Pr[Y \geq t])$ for any $t \geq 0$. Then for $\ell \geq 1$ if $\mathbf{E}[Y^\ell] < \infty$, we have $\mathbf{E}[Z^\ell] = O(\mathbf{E}[Y^\ell])$.*

Theorem 6. *A distribution \mathcal{D} is a strong (d, s) -JL distribution if and only if it has the strong (d, s) -JL moment property.*

Proof. First assume \mathcal{D} has the strong JL moment property. Then, for arbitrary $w \in \mathcal{S}^{d-1}, \varepsilon > 0$,

$$\Pr_{S \sim \mathcal{D}} [|\|Sw\|^2 - 1| > \varepsilon] < \varepsilon^{-\ell} \cdot \mathbf{E}[|\|Sw\|^2 - 1|^\ell] < O(\max\{\sqrt{\ell/(\varepsilon^2 s)}, \ell/(\varepsilon s)\}^\ell).$$

The claim follows by setting $\ell = O(\min\{\varepsilon, \varepsilon^2\} \cdot s)$.

Now assume \mathcal{D} is a strong JL distribution. Set $Z = |\|Sw\|^2 - 1|$. Since \mathcal{D} is a strong JL distribution, the right tail of Z is big-Oh of that of the absolute value of the nonnegative random variable Y which is the sum of a Gaussian with mean 0 and variance $O(1/s)$, and an exponential random variable with parameter s . Now, apply Fact 5. ■

Remark 1. Theorem 6 implies that any strong JL distribution can be derandomized using $2 \log(1/\delta)$ -wise independence giving an alternate proof of the derandomized JL result of Clarkson and Woodruff (Theorem 2.2 in [5]). This is because, by Markov’s inequality with ℓ even, and for $\varepsilon < 1$,

$$\Pr_{S \sim \mathcal{D}} [|\|Sw\|^2 - 1| > \varepsilon] < \varepsilon^{-\ell} \cdot \mathbf{E}_{S \sim \mathcal{D}} [(\|Sw\|^2 - 1)^\ell] \leq 2^{O(\ell)} \cdot (\varepsilon^{-1} \cdot \sqrt{\ell/s})^\ell. \tag{3.1}$$

Setting $\ell = \log(1/\delta)$ and $s = C\ell/\varepsilon^2$ for $C > 0$ sufficiently large makes the above probability at most δ . Now, note the ℓ th moment is determined by 2ℓ -wise independence of the entries of S .

4 A Generic JL Derandomization Template

Theorem 6 and Remark 1 provide the key insight for our construction. If we use $\ell = 2 \log(1/\delta)$ -wise independent Bernoulli entries as suggested in Remark 1, the seed length would be $O(\ell \log d) = O(\log(1/\delta) \log d)$ for $s = \Theta(\varepsilon^{-2} \log(1/\delta))$. However, note that in Eq. (3.1), a trade-off can be made between the amount of independence needed and the final embedding dimension without changing the error probability. In particular, it suffices to use 4-wise independence if we embed into $s = \Omega(\varepsilon^{-2} \delta^{-1})$ dimensions. In general, if $s = C\varepsilon^{-2} q$ for $\log^2(1/\delta) \leq q \leq 1/\delta$, it suffices to set $\ell = O(\log_q(1/\delta))$ to make the right hand side of Eq. (3.1) at most δ . By gradually reducing the dimension over the course of several iterations, using higher independence in each iteration, we obtain shorter seed length.

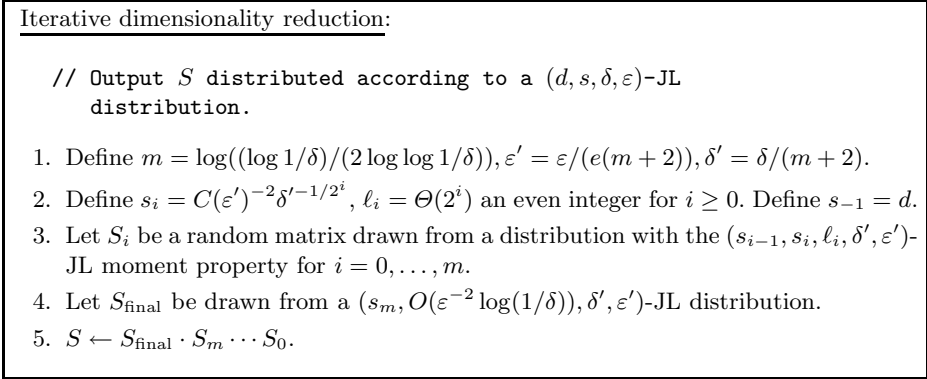


Fig. 1. A general derandomization scheme for distributions with JL moment properties

Our main construction is described in Figure 1. We first embed into $O(\varepsilon^{-2} \delta^{-1})$ dimension using 4-wise independence. We then iteratively project from $O(\varepsilon^{-2} \delta^{-1/2^i})$ dimensions into $O(\varepsilon^{-2} \delta^{-1/2^{i+1}})$ dimensions until we have finally embedded into $O(\varepsilon^{-2} \log^2(1/\delta))$ dimensions. In our final step, we embed into the optimal target dimension using $2 \log(1/\delta)$ -wise independence. Note the Bernoulli distribution is not special here; we could use any family of strong JL distributions.

Theorem 7. *The output matrix S in Figure 1 is distributed according to a $(d, s, \delta, \varepsilon)$ -JL distribution for $s = O(\log(1/\delta)/\varepsilon^2)$.*

Proof. For a fixed vector w , let $w^i = S_i \cdots S_0 w$, and let w^{-1} denote w . Then by our choice of s_i and a Markov bound on the ℓ_i th moment,

$$\Pr [\|w^i\|^2 - \|w^{i-1}\|^2 \| > \varepsilon' \|w^{i-1}\|^2] < \varepsilon'^{-\ell_i} \cdot \mathbf{E}[(\|w^i\|^2 / \|w^{i-1}\|^2 - 1)^{\ell_i}] < \delta'$$

for $0 \leq i \leq m$. We also have $\Pr [\|S_{\text{final}} w^m\|^2 - \|w^m\|^2 \| > \varepsilon' \|w^m\|^2] < \delta'$. By a union bound, $\|S_{\text{final}} w^m\|^2 \leq (1 + \varepsilon')^{m+2} \leq e^{(m+2)\varepsilon'} \leq 1 + \varepsilon$ with probability $1 - (m+2)\delta' = 1 - \delta$. ■

As a corollary, we obtain our main theorem, Theorem 2.

Proof. [of Theorem 2] We let the distributions in Steps 3 and 4 of Figure 1 be strong JL distributions. Then Steps 3 and 4 are satisfied by Remark 1.

The seed length required to generate S_0 is $O(\log d)$. For S_i for $i > 0$ the seed length is $O(\ell_i \log(\varepsilon'^{-2} \delta'^{1/2^i})) = O(2^i \log(1/\varepsilon') + \log(1/\delta'))$, which is never larger than $O((\log(1/\delta')/\log \log(1/\delta')) \log(1/\varepsilon') + \log(1/\delta'))$, which is $O((\log(1/\delta)/\log \log(1/\delta)) \log(1/\varepsilon) + \log(1/\delta))$. The seed length required for S_{final} is $O(\log(1/\delta') \log(\log(1/\delta')/\varepsilon')) = O(\log(1/\delta) \log(\log(1/\delta)/\varepsilon))$. Thus, the total seed length is dominated by generating S_0 and S_{final} , giving the claim. The distortion and error probabilities can be bounded by a union bound. ■

5 Explicit JL Families via Samplers

We now give an alternate construction of an explicit JL family. The construction is similar in spirit to that of the previous section and has the additional

property that matrix-vector products for matrices output by the generator can be computed in time roughly $O(d \log d + s^3)$, as it is based on the Fast Johnson-Lindenstrauss Transform (FJLT) of [1]. For clarity, we concentrate on the case of $\delta = \Theta(1/d^c)$ polynomially small. The case of general δ can be handled similarly with some minor technical issues³ that we skip in this extended abstract. Further, we assume that $\log(1/\delta)/\varepsilon^2 < d$ as else JLL is not interesting.

As outlined in the introduction, we first give a family of rotations to regularize vectors in \mathbb{R}^d . For a vector $x \in \mathbb{R}^d$, let $D(x) \in \mathbb{R}^{d \times d}$ be the diagonal matrix with $D(x)_{ii} = x_i$.

Lemma 2. *Let $x \in \{1, -1\}^d$ be drawn from a k -wise independent distribution. Then, for every $w \in \mathbb{R}^d$ with $\|w\| = 1$, $0 < \alpha < 1/2$,*

$$\Pr[\|HD(x)w\|_\infty > n^{-(1/2-\alpha)}] \leq \frac{k^{k/2}}{n^{\alpha k - 1}}.$$

Proof. Let $v = HD(x)w$. Then, for $i \in [d]$, $v_i = \sum_j H_{ij}x_jw_j$ and $\mathbb{E}[v_i^2] = \sum_j H_{ij}^2w_j^2 = 1/d$. By Markov’s inequality and the Khintchine-Kahane inequality (Lemma 1),

$$\Pr[|v_i| > d^{-(1/2-\alpha)}] \leq \mathbb{E}[v_i^k] \cdot d^{(1/2-\alpha)k} \leq k^{k/2}d^{(1/2-\alpha)k}/d^{k/2} = k^{k/2}d^{-\alpha k}.$$

The claim now follows from a union bound over $i \in [d]$. ■

We now give a family of transformations for reducing d dimensions to $\tilde{O}(d^{1/2}) \cdot \text{poly}(s_{\text{opt}})$ dimensions using oblivious samplers. For $S \subseteq [d]$, let $\mathcal{P}_S : \mathbb{R}^d \rightarrow \mathbb{R}^{|S|}$ be the projection onto the coordinates in S . In the following let C be the universal constant from Corollary 1.

Lemma 3. *Let $\mathcal{S} \equiv \mathcal{S}(d, d^{1/2C}, \varepsilon, \delta)$, $s = O(d^{1/2} \log^C(1/\delta)/\varepsilon^C)$ be as in Corollary 1 and let \mathcal{D} be a k -wise independent distribution over $\{1, -1\}^d$. For $S \in \mathcal{S}$, $x \leftarrow \mathcal{D}$, define the random linear transformation $A_{S,x} : \mathbb{R}^d \rightarrow \mathbb{R}^s$ by $A_{S,x} = \sqrt{d/s} \cdot \mathcal{P}_S \cdot HD(x)$. Then, for every $w \in \mathbb{R}^d$ with $\|w\| = 1$,*

$$\Pr[\|A_{S,x}(w)\|^2 - 1 \geq \varepsilon] \leq \delta + k^{k/2}/d^{k/4C-1}.$$

Proof. Let $v = HD(x)w$. Then, $\|v\| = 1$ and by Lemma 2 applied for $\alpha = 1/4C$,

$$\Pr[\|v\|_\infty > d^{-(1/2-1/4C)}] \leq k^{k/2}/d^{k/4C-1}.$$

Now condition on the event $\|v\|_\infty \leq d^{-(1/2-1/4C)}$. Define $f : [d] \rightarrow \mathbb{R}$ by $f(i) = d \cdot v_i^2 \leq d^{1/2C} = B$. Then,

$$\|A_{S,x}(w)\|^2 = (d/s)\|\mathcal{P}_S(v)\|^2 = \frac{1}{s} \sum_{i \in S} dv_i^2 = \frac{1}{s} \sum_{i \in S} f(i),$$

³ In case of very small δ , we need to ensure that we never increase the dimension - which can be done trivially by using the identity transformation. In case of large δ , we first embed the input vector into $O(1/\delta\varepsilon^2)$ dimensions using 4-wise independence as in Section 4.

and $\mathbb{E}_{i \in_u [d]} f(i) = (1/d) \sum_i d \cdot v_i^2 = 1$. Therefore, by **Corollary 1**,

$$\Pr[| \|A_{S,x}(w)\|^2 - 1 | \geq \varepsilon] = \Pr_{S \in_u \mathcal{S}} \left[\left| \frac{1}{s} \sum_{i \in S} f(i) - \mathbb{E}_{i \in_u [d]} f(i) \right| \geq \varepsilon \right] \leq \delta.$$

The claim now follows. ■

We now recursively apply the above lemma. Fix $\varepsilon, \delta > 0$. Let $\mathcal{A}(d, k) : \mathbb{R}^d \rightarrow \mathbb{R}^{s(d)}$ be the collection of transformations $\{A_{S,x} : S \in_u \mathcal{S}, x \leftarrow D\}$ as in the above lemma for $s(d) = s(d, d^{1/2C}, \varepsilon, \delta) = c_1 d^{1/2} (\log d / \varepsilon)^C$, for a constant c_1 . Note that we can sample from $\mathcal{A}(d, k)$ using $r(d, k) = k \log d + O(\log d + \log(1/\delta)) = O(k \log d)$ random bits.

Let $d_0 = d$, and let $d_{i+1} = s(d_i)$. Let $k_0 = 8C(c + 1)$ (recall that $\delta = 1/d^c$) and $k_{i+1} = 2^i k_0$. The parameters d_i, k_i are chosen so that $1/d_i^{k_i}$ is always polynomially small. Fix $t > 0$ to be chosen later so that $k_i < d_i^{1/4C}$ for $i < t$.

Lemma 4. *For $A_0 \in_u \mathcal{A}(d_0, k_0), A_1 \in_u \mathcal{A}(d_1, k_1), \dots, A_{t-1} \in_u \mathcal{A}(d_{t-1}, k_{t-1})$ chosen independently, and $w \in \mathbb{R}^d, \|w\| = 1$,*

$$\Pr[(1 - \varepsilon)^t \leq \|A_{t-1} \cdots A_1 A_0(w)\|^2 \leq (1 + \varepsilon)^t] \geq 1 - t\delta - \sum_{i=0}^{t-1} \frac{k_i^{k_i/2}}{d_i^{k_i/4C-1}}.$$

Proof. The proof is by induction on $i = 1, \dots, t$. For $i = 1$, the claim is same as **Lemma 3**. Suppose the statement is true for $i - 1$ and let $v = A_{i-1} \cdots A_0(w)$. Then, $v \in \mathbb{R}^{d_i}$ and the lemma follows by **Lemma 3** applied to $\mathcal{A}(d_i, k_i)$, and v . ■

What follows is a series of elementary calculations to bound the seed-length and error from the above lemma. Observe that

$$d^{(1/2)^i} \leq d_i = d^{(1/2)^i} \cdot \left(\frac{c_1 \log^C(d)}{\varepsilon^C} \right)^{1+(1/2)^i+\dots+(1/2)^{i-1}} \leq d^{(1/2)^i} \left(\frac{c_1 \log^C d}{\varepsilon^C} \right)^2. \tag{5.1}$$

Let $t = O(\log \log d)$ be such that $2^t = \log d / 4C \log \log d$. Then, $d_t \leq \log^{4C} d \cdot (c_1 \log^C d / \varepsilon^C)^2 = O(\log^{6C} d / \varepsilon^{2C})$, and for $i < t$,

$$k_i < k_t = 8C(c + 1)2^t = 2(c + 1) \log d / \log \log d < \log d = d^{(1/2)^t/4C} < d_t^{1/4C} < d_i^{1/4C}, \tag{5.2}$$

where we assumed that $\log \log d > 2c + 2$. Therefore, the error in **Lemma 4** can be bounded by

$$t\delta + \sum_{i=0}^{t-1} \frac{k_i^{k_i/2}}{d_i^{k_i/4C-1}} \leq t\delta + d \sum_{i=0}^{t-1} d_i^{-k_i/8C} \tag{Equation 5.2}$$

$$\leq t\delta + d \sum_{i=0}^{t-1} (d^{1/2^i})^{-8C(c+1) \cdot 2^i/8C} \tag{Equation 5.1}$$

$$\leq t\delta + t/d^c \leq 2t\delta \tag{as $\delta > 1/d^c$ }.$$

Note that,

$$k_i \log d_i \leq 8C(c + 1) \cdot 2^i (\log d/2^i + 2C \log \log d + 2C \log(1/\varepsilon)) = O(\log d + \log d \log(1/\varepsilon) / \log \log d).$$

Therefore, the randomness needed after $t = O(\log \log d)$ iterations is

$$\sum_{i=0}^{t-1} O(k_i \log d_i) = O(\log d \log \log d + (\log d) \log(1/\varepsilon)).$$

Combining the above arguments (applied to $\delta' = \delta / \log \log d$ and $\varepsilon' = \varepsilon / \log \log d$ and simplifying the resulting expression for seed-length) we obtain our fast de-randomized JL family.

Theorem 8 (Fast Explicit JL Family). *There exists a $(d, O(\log(1/\delta)/\varepsilon^2), \delta, \varepsilon)$ -JL generator with seed-length $r = O(\log d + \log(1/\delta)(\log(\log(1/\delta)/\varepsilon)))$ such that for every vector $w \in \mathbb{R}^d$, $y \in \{0, 1\}^r$, $G(y)w$ can be evaluated in time $O(d \log d + \text{poly}(\log(1/\delta)/\varepsilon))$.*

Proof. We suppose that $\delta = \Theta(1/d^c)$ - the analysis for the general case is similar. From the above arguments there is an explicit generator that takes $O(\log(d/\delta) \cdot \log(\log(d/\delta)/\varepsilon))$ random bits and outputs a linear transformation $A : \mathbb{R}^d \rightarrow \mathbb{R}^m$ for $m = \text{poly}(\log(d/\delta), 1/\varepsilon)$, satisfying the JL property with error at most δ and distortion at most ε . The theorem now follows by composing the transformations of the above theorem with a Bernoulli matrix having $2 \log(1/\delta)$ -wise independence. The additional randomness required is $O(\log(1/\delta) \log m) = O(\log(1/\delta)(\log \log(d/\delta) + \log(1/\varepsilon)))$.

We next bound the time for computing matrix-vector products for the matrices we output. Note that for $i < t$, the matrices A_i of [Lemma 4](#) are of the form $\mathcal{P}_S \cdot H_{d_i} D(x)$ for a k -wise independent string $x \in \{1, -1\}^{d_i}$. Thus, for any vector $w_i \in \mathbb{R}^{d_i}$, $A_i w_i$ can be computed in time $O(d_i \log d_i)$ using the discrete Fourier transform. Therefore, for any $w = w_0 \in \mathbb{R}^{n_0}$, the product $A_{t-1} \cdots A_1 A_0 w_0$ can be computed in time

$$\begin{aligned} \sum_{i=0}^{t-1} O(d_i \log d_i) &\leq O(d \log d) + \log d \cdot \sum_{i=1}^{t-1} O\left(d^{1/2^i} (\log(1/\delta)/\varepsilon^2)^2\right) \quad \text{(Equation 5.1)} \\ &= O(d \log d + \sqrt{d} \log d \log^2(1/\delta)/\varepsilon^4). \end{aligned}$$

The above bound dominates the time required to perform the final embedding.

A similar calculation shows that for indices $i \in s, j \in [d]$, the entry $G(y)_{ij}$ of the generated matrix can be computed in space $O(\sum_i \log d_i) = O(\log d + \log(1/\varepsilon) \cdot \log \log d)$ by expanding the product of matrices and enumerating over all intermediary indices⁴. The time required to perform the calculation is $O(s \cdot d_t \cdot d_{t-1} \cdots d_1) = d \cdot (\log d/\varepsilon)^{O(\log \log d)}$. ■

⁴ We also need to account for the time and space needed by the samplers and for generating k -wise independent strings. However, these are dominated by the task of enumerating over all indices; for instance, the samplers of [15](#) are in NC.

Remark 2. We can use the FJLT of [1] in the framework of Figure 1 to get seed length and update time as above. Details are deferred to the full version.

6 Optimality of JL Lemma

We next prove [Theorem 3](#), the optimality of the number of rows in the JL Lemma. Let \mathcal{A} be a distribution over linear transformations from \mathbb{R}^d to \mathbb{R}^k such that $\Pr_{S \sim \mathcal{A}}[\|Sw\|^2 - 1] < \delta$ for any $w \in \mathcal{S}^{d-1}$. Then, it must be the case that $\Pr_{S \sim \mathcal{A}}[\Pr_{w \in \mathcal{S}^{d-1}}[\|Sw\|^2 - 1]] < \delta$. By an averaging argument, there must exist a linear transformation S in the support of \mathcal{A} such that $\Pr_{w \in \mathcal{S}^{d-1}}[\|Sw\|^2 - 1] < \delta$. We show this cannot happen unless k is sufficiently large.

Theorem 9. *If $S : \mathbb{R}^d \rightarrow \mathbb{R}^k$ is a linear transformation with $d > 2k$ and $\varepsilon > 0$ sufficiently small, then for w a randomly chosen vector in \mathcal{S}^{d-1} , $\Pr[\|Sw\|^2 - 1 > \varepsilon] \geq \exp(-O(k\varepsilon^2 + 1))$.*

Proof. First note that we can assume S is surjective as else, we may replace \mathbb{R}^k by the image of S . Let $V = \ker(S)$ and let U be the orthogonal complement of V in \mathbb{R}^d . Then $\dim(U) = k$, $\dim(V) = d - k$. Now, any $w \in \mathbb{R}^d$ can be written uniquely as $w_V + w_u$ where w_V and w_u are the components of w in V and U respectively. We may then write $w_V = r_V \Omega_V$, $w_u = r_u \Omega_u$, where r_V, r_u are positive real numbers and Ω_V and Ω_u are unit vectors in V and U respectively.

Let $s_V = r_V^2$ and $s_u = r_u^2$. We may now parameterize the unit sphere by $(s_V, \Omega_V, s_u, \Omega_u) \in [0, 1] \times \mathcal{S}^{d-k-1} \times [0, 1] \times \mathcal{S}^{k-1}$, so that $s_V + s_u = 1$. It is clear that the uniform measure on the sphere is given in these coordinates by $f(s_u) ds_u d\Omega_V d\Omega_u$ for some function $f : [0, 1] \rightarrow [0, 1]$. We next show that

$$f(s_u) = C_f \cdot (1 - s_u)^{(d-k-2)/2} s_u^{(k-2)/2}, \tag{6.1}$$

where C_f is a normalization constant. Observe that $f(s_u)$ should be proportional to the limit as $\delta_1, \delta_2 \rightarrow 0^+$ of $(\delta_1 \delta_2)^{-1}$ times the volume of points w satisfying $\|w_u\|^2 \in [s_u, s_u + \delta_2]$ and $\|w_V\|^2 \in [1 - \|w_u\|^2, 1 - \|w_u\|^2 + \delta_1]$. For fixed w_u , the latter volume is within $O(\delta_1 \delta_2)$ of the volume of w_V so that $\|w_V\|^2 \in [s_V, s_V + \delta_1]$. Now the measure on V is $r_V^{d-k-1} dr_V d\Omega_V$. Therefore it also is $\frac{1}{2} s_V^{(d-k-2)/2} ds_V d\Omega_V$. Therefore this volume over V is proportional to $s_V^{(d-k-2)/2} (\delta_1 + O(\delta_1 \delta_2 + \delta_1^2))$. Similarly the volume of w_u so that $\|w_u\|^2 \in [s_u, s_u + \delta_2]$ is proportional to $s_u^{(k-2)/2} (\delta_2 + O(\delta_2^2))$. Hence f is proportional to $s_V^{(d-k-2)/2} s_u^{(k-2)/2}$.

We are now prepared to prove the theorem. The basic idea is to first condition on Ω_V, Ω_u . We let $C = \|S\Omega_u\|^2$. Then if w is parameterized by $(s_V, \Omega_V, s_u, \Omega_u)$, $\|Sw\|^2 = C s_u$. Choosing w randomly, we know that $s = s_u$ satisfies the distribution $\frac{s^{(k-2)/2} (1-s)^{(d-k-2)/2}}{\beta((k-2)/2, (d-k-2)/2)} ds = f(s) ds$ on $[0, 1]$. We need to show that for any $c = \frac{1}{C}$, the probability that s is not in $[(1 - \varepsilon)c, (1 + \varepsilon)c]$ is $\exp(-O(\varepsilon^2 k))$. Note that $f(s)$ attains its maximum value at $s_0 = \frac{k-2}{d-4} < \frac{1}{2}$. Notice that $\log(f(s_0(1+x)))$ is some constant plus $\frac{k-2}{2} \log(s_0(1+x)) + \frac{d-k-2}{2} \log(1 - s_0 - x s_0)$. If $|x| <$

$1/2$, then this is some constant plus $-O(kx^2)$. So for such w , $f(s_0(1+x)) = f(s_0) \exp(-O(kx^2))$. Furthermore, for all x , $f(s_0(1+x)) = f(s_0) \exp(-\Omega(kx^2))$. This says that f is bounded above by a normal distribution and checking the normalization we find that $f(s_0) = \Omega(s_0^{-1}k^{1/2})$.

We now show that both $\Pr(s < (1-\varepsilon)s_0)$ and $\Pr(s > (1+\varepsilon)s_0)$ are reasonably large. We can lower bound either as

$$\begin{aligned} s_0 \int_{\varepsilon}^{1/2} f(s_0) \exp(-O(kx^2)) dx &\geq \Omega(k^{1/2}) \int_{\varepsilon}^{\varepsilon+k^{-1/2}} \exp(-O(kx^2)) dx \\ &\geq \Omega(\exp(-O(k(\varepsilon+k^{-1/2})^2))) \\ &\geq \exp(-O(k\varepsilon^2+1)). \end{aligned}$$

Hence since one of these intervals is disjoint from $[(1-\varepsilon)c, (1+\varepsilon)c]$, the probability that s is not in $[(1-\varepsilon)c, (1+\varepsilon)c]$ is at least $\exp(-O(k\varepsilon^2+1))$. ■

References

1. Ailon, N., Chazelle, B.: The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. *SIAM J. Comput.* 39(1), 302–322 (2009)
2. Achlioptas, D.: Database-friendly random projections: Johnson-lindenstrauss with binary coins. *J. Comput. Syst. Sci.* 66(4), 671–687 (2003)
3. Alon, N., Matias, Y., Szegedy, M.: The Space Complexity of Approximating the Frequency Moments. *J. Comput. Syst. Sci.* 58(1), 137–147 (1999)
4. Arriaga, R.I., Vempala, S.: An algorithmic theory of learning: Robust concepts and random projection. *Machine Learning* 63(2), 161–182 (2006)
5. Clarkson, K.L., Woodruff, D.P.: Numerical linear algebra in the streaming model. In: *STOC*, pp. 205–214 (2009)
6. Dasgupta, S., Gupta, A.: An elementary proof of a theorem of johnson and lindenstrauss. *Random Struct. Algorithms* 22(1), 60–65 (2003)
7. Engebretsen, L., Indyk, P., O’Donnell, R.: Derandomized dimensionality reduction with applications. In: *SODA*, pp. 705–712 (2002)
8. Indyk, P., Motwani, R.: Approximate nearest neighbors: towards removing the curse of dimensionality. In: *STOC*, pp. 604–613 (1998)
9. Johnson, W.B., Lindenstrauss, J.: Extensions of Lipschitz mappings into a Hilbert space. *Contemp. Math.* 26, 189–206 (1984)
10. Jayram, T.S., Woodruff, D.P.: Optimal bounds for Johnson-Lindenstrauss transforms and streaming problems with low error. In: *SODA*, pp. 1–10 (2011)
11. Karnin, Z., Rabani, Y., Shpilka, A.: Explicit dimension reduction and its applications. In: *CCC*, pp. 262–272 (2011)
12. Ledoux, M., Talagrand, M.: *Probability in Banach spaces: isoperimetry and processes*. Springer, Heidelberg (1991)
13. Matousek, J.: On variants of the Johnson-Lindenstrauss lemma. *Random Struct. Algorithms* 33(2), 142–156 (2008)
14. Sivakumar, D.: Algorithmic derandomization via complexity theory. In: *STOC*, pp. 619–626 (2002)
15. Zuckerman, D.: Randomness-optimal oblivious sampling. *Random Struct. Algorithms* 11(4), 345–367 (1997)

Correlation Bounds for Poly-size AC^0 Circuits with $n^{1-o(1)}$ Symmetric Gates

Shachar Lovett* and Srikanth Srinivasan*

Institute of Advanced Study, Princeton, USA
{slovett,srikanth}@math.ias.edu

Abstract. Average-case circuit lower bounds are one of the hardest problems tackled by computational complexity, and are essentially only known for bounded-depth circuits with AND,OR,NOT gates (i.e. AC^0). Faced with this adversity, one line of research has been to gradually augment AC^0 circuits with a small number of more general gates. Most results to date give lower bounds for quasi-polynomial size AC^0 circuits augmented by a poly-logarithmic number of gates, and the correlation bounds obtained are inverse quasi-polynomial.

We continue this line of research, but restrict our attention to polynomial size AC^0 circuits. Surprisingly, we show that this restriction allows us to prove much stronger results: we can augment the AC^0 circuit with $n^{1-o(1)}$ many gates, and still obtain inverse exponential correlation bounds. Explicitly,

1. Poly-size AC^0 circuits with $n^{1-o(1)}$ arbitrary symmetric gates have exponentially small correlation with an explicitly given function.
2. Poly-size AC^0 circuits with $n^{1/2-o(1)}$ threshold gates have exponentially small correlation with the same explicit function.
3. Poly-size AC^0 circuits with $n^{1-o(1)}$ counting gates modulo s have exponentially small correlation with the sum of the bits modulo q , where s, q are co-prime.

Our proof techniques combine the meet-in-the-middle approach for circuit lower bounds with restrictions (due to Ajtai) that are tailored to polynomial-size circuits.

1 Introduction

The quest for circuit lower bounds has proven to be one of the harder challenges in computational complexity. By circuit lower bounds one can consider two alternatives: worst-case hardness, where one shows that small circuits cannot compute a given function; and average-case hardness, where one shows that small circuits cannot even predict the given function well (these are often called correlation bounds). Average case hardness results usually¹ imply also pseudo-random generators for these classes of circuits [17,19].

* Supported by NSF grants CCF-0832797 and DMS-0835373.

¹ Unless, informally, the class is too weak.

There is essentially just one general class of circuits where strong average-case lower bounds are known: AC^0 , the class of bounded-depth circuits with unbounded fan-in AND, OR and NOT gates. This class represents parallel computation with a bounded number of rounds, where basic computations correspond to gates, and only trivial basic computations (AND, OR, and NOT) are allowed. A sequence of works, culminating with the celebrated result of Håstad [14], showed that exponential size AC^0 circuits cannot predict the parity of n bits with better than exponentially small advantage. Nisan [17] used this to construct pseudorandom generators against AC^0 with poly-logarithmic seed length.

Obviously, one would like to prove strong lower bounds on more realistic models of parallel computation, where the main challenge is to allow general symmetric local computations. This amounts to constant depth circuit with arbitrary symmetric gates; which is known to also be equivalent to TC^0 , where arbitrary threshold gates are allowed [22,10]. Despite much research on these problems, no super-polynomial lower bounds are known for this class.

Therefore, research has turned towards studying more restricted models, with the goal of improving proof techniques. One line of research has been to allow arbitrary constant depth, but limit the symmetric basic gates allowed. This approach was essentially successful only for $ACC^0[p]$, where in addition to AND, OR, NOT gates, counting gates modulo p are also allowed. Razborov and Smolensky [20,23] showed a worst-case lower bound when p is a prime power. They showed that such circuits require exponential size to compute the sum of the bits modulo q , where q is co-prime to p . When p is not a prime power, proving worst-case lower bounds is still open, and the best result to date is of Williams [26], which showed that subexponential $ACC^0[p]$ circuits cannot compute all of NEXP. Average-case lower bounds are not known for any p .

Another line of research is that of considering intermediate models between AC^0 circuits and general bounded depth circuits with symmetric (or threshold) gates, where the AC^0 circuit is augmented with a *few* more general gates (such as arbitrary symmetric gates, threshold gates, modular counting gates, etc.). Here, researchers have considered various types of general gates. Initial work considered AC^0 circuits augmented by a few majority gates. A sequence of works [2,6,5,4] showed that even if one allows $n^{o(1)}$ gates, one still requires exponential size in order to compute the parity of n bits (or more generally, their sum modulo any constant q). Later research allowed more general types of augmented gates, however success has been more limited: the size of the circuits for which lower bounds were shown decreased to quasi-polynomial, and the number of augmented gates reduced to poly-logarithmic. Explicitly, researchers have considered augmenting AC^0 circuits by a few modular gates [8], threshold gates [11] and arbitrary symmetric gates [21,13,24].

In this work we limit ourselves to proving lower bounds for *polynomial size* AC^0 circuits, augmented with a few general gates. The motivation is that lower bounds for such models are sufficient for proving (albeit, weak) *super-polynomial* lower-bounds. Our main results show that by restricting the size of the circuits

to be polynomial (instead of quasi-polynomial), the number of augmented gates can be dramatically increased. Explicitly, we show that

1. Polynomial size AC^0 circuits with $n^{1-o(1)}$ arbitrary symmetric gates cannot approximate an explicitly given function (the same function considered in [21][13][24]) with better than exponentially small advantage. Previous results [24] allowed for at most a poly-logarithmic number of arbitrary symmetric gates, and were able to achieve only inverse quasi-polynomial advantage.
2. Polynomial size AC^0 circuits with $n^{1/2-o(1)}$ threshold gates cannot approximate the same explicit function with better than exponentially small advantage. Previous results allowed for at most poly-logarithmic number of arbitrary threshold gates [11] or $n^{O(1/d)}$ many majority gates [5], where d is the depth of the circuit [2].
3. Polynomial size AC^0 circuits with $n^{1-o(1)}$ counting gates modulo m (for arbitrary constant m) cannot approximate the sum of the bits modulo q , where m, q are co-prime, with better than exponentially small advantage. Previous results [12] allowed for at most a poly-logarithmic number of modular counting gates.

The above results imply (using the Nisan-Wigderson generator [19]) a pseudorandom generator with polynomial stretch and exponentially small error against the above models; that is, for any constant $c > 1$ an explicit map $\mathcal{G} : \{0, 1\}^n \rightarrow \{0, 1\}^{n^c}$ fooling the above models with error $\exp(-n^{\Omega(1)})$.

Another model which was widely studied is that of multivariate polynomials over \mathbb{F}_2 . Correlation bounds are known when the degree of the polynomials is bounded. Bourgain [7] and Viola and Wigderson [25] showed that polynomials over \mathbb{F}_2 of degree $O(\log n)$ have exponentially small correlation with certain explicit functions, e.g., the sum of the bits modulo 3. Proving correlation bounds for polynomials of super-logarithmic degree is an important open problem. We show that when one considers *sparse polynomials*, the restriction on the degree can be lifted. The following result follows immediately as a simple case of our third result given above; but because of its easiness we provide a direct proof for it: *Polynomials over \mathbb{F}_2 with a polynomial number of monomials have exponentially small correlation with the sum of the bits modulo 3. Previous results [24] gave only an inverse quasi-polynomial correlation bounds.*

Our proofs are based on three main ingredients: tree restrictions, communication complexity lower bounds (similar to [24]) and a random restriction method of Ajtai [1] for a polynomial collection of DNFs. Due to lack of space, some proofs are omitted from this extended abstract, and are deferred to the full version.

2 Preliminaries

Given a distribution D over $\{0, 1\}^n$ and two boolean functions $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$, we define $\text{Corr}_D(f, g)$, the *correlation between f and g under D* , as

² The same proof also works for the more general model where we allow the circuit to contain $n^{1/2-o(1)}$ threshold *and* symmetric gates.

follows: $\text{Corr}_D(f, g) = |\mathbb{E}_x[(-1)^{f(x)} \cdot (-1)^{g(x)}]|$. We also use $\text{Corr}(f, g)$ to denote $\text{Corr}_U(f, g)$, where U is the uniform distribution over $\{0, 1\}^n$.

A *tree-restriction* on n boolean variables x_1, \dots, x_n is a decision tree T on these variables. Each leaf ℓ of the tree T corresponds to a subcube Q_ℓ of $\{0, 1\}^n$ and these subcubes partition the space $\{0, 1\}^n$. There is a natural probability distribution μ_T over the leaves of the tree-restriction T , which is obtained by starting from the root and choosing a child uniformly until a leaf is reached. Given a leaf ℓ of a tree restriction T on $\{0, 1\}^n$ and a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, we denote by $f|_\ell$ the restriction of f to the subcube Q_ℓ .

We are concerned with proving lower bounds against AC^0 circuits augmented with some other gates of unbounded fan-in that compute functions possibly not in AC^0 . We will use the phrase “ AC^0 circuits of size $s(n)$ ” for even superpolynomial $s(n)$, by which we will mean the class of functions computed by boolean circuits with unbounded fan-in of size at most $s(n)$. We use some standard notation ([9]) for different gate types: SYM gates compute symmetric functions (of their input variables); THR gates compute linear threshold functions; and MOD_s gates for an integer $s \geq 2$, compute the boolean function that outputs 1 iff the number of 1s in its input is not divisible by s . Note that a MOD_s gate is also a SYM gate. We will also use the ANY gate, which could compute *any* boolean function of its input variables. For a gate type G and $t \in \mathbb{N}$, G_t denotes a gate of type G that has fanin at most t . E.g., AND_t denotes the class of AND gates of fanin at most t .

We will also be interested in the communication complexity of boolean functions in the k -party Number-on-the-Forehead (NOF) model. The reader is referred to Kushilevitz and Nisan [16] for relevant definitions and results.

3 An Illustrative Example: Correlation Bounds for Sparse Polynomials

It is an important open problem to give explicit functions which have small correlation with any multivariate polynomial over \mathbb{F}_2 of degree exceeding $\log n$. The best results to date give explicit functions which have correlation $1/\sqrt{n}$ with polynomials of larger degree. We demonstrate a restriction technique which allows us to get exponential correlation bounds for polynomials of large degree, as long as they are *sparse*: i.e., they contain only polynomially many monomials. This technique is the core of our results for AC^0 circuits with $n^{1-o(1)}$ symmetric gates (or $n^{1/2-o(1)}$ threshold gates), and we wish to first demonstrate it in the simpler case of sparse polynomials. We note that previous results [24] were able to achieve only inverse quasi-polynomial correlation bounds for sparse polynomials.

Our main result of this section is that sparse polynomials over \mathbb{F}_2 have exponentially small correlation with the sum of the bits modulo 3. This easily extends to any \mathbb{F}_p and q for co-prime p, q . We say $p(x_1, \dots, x_n)$, a polynomial over \mathbb{F}_2 , is s -sparse if it can be expressed as the sum of $\leq s$ monomials, where a monomial is a product of the form $m(x) = \prod_{i \in S} (x_i - a_i)$ where $S \subseteq [n]$ is a set of variables and $a_i \in \mathbb{F}_2$ are some coefficients. The main theorem we show is the following.

Theorem 1. *Let $p(x)$ be an n -variate $n^{o(\log n)}$ -sparse polynomial over \mathbb{F}_2 . Then for any $\alpha \neq \beta \in \{0, 1, 2\}$, $|\Pr_x[p(x) = 0 \mid \sum_{i=1}^n x_i \equiv \alpha \pmod{3}] - \Pr_x[p(x) = 0 \mid \sum_{i=1}^n x_i \equiv \beta \pmod{3}]| \leq \exp(-n^{1-o(1)})$, where x is uniformly chosen from $\{0, 1\}^n$.*

An equivalent form of Theorem 1, which is more amenable for analysis, is the following. $\mathbb{E}_{x \in \{0,1\}^n} \left[(-1)^{p(x)} \omega_3^{\sum_{i=1}^n x_i} \right] \leq \exp(-n^{1-o(1)})$, where ω_3 is a cubic root of unity. The proof of equivalence is standard and is omitted. For a proof, see e.g. [25]. We will prove the above in the remainder of this section.

The main idea of the proof is to use restrictions to reduce sparse polynomials to low-degree polynomials, and apply correlation bounds for low-degree polynomials. For this, we will use the following result of Viola and Wigderson [25] (who strengthened a result of Bourgain [7]), which shows that polynomials of degree $\ll \log n$ have exponential small correlation with the sum of the bits mod 3.

Theorem 2 ([25]). *Let $p(x)$ be an n -variate polynomial over \mathbb{F}_2 of degree d . Then $\mathbb{E}_{x \in \{0,1\}^n} \left[(-1)^{p(x)} \omega_3^{\sum_{i=1}^n x_i} \right] \leq \exp(-\Omega(n/4^d))$.*

Let $p(x)$ be an n -variate s -sparse polynomial over \mathbb{F}_2 . The first step is to use tree restrictions to restrict $p(x)$ to a polynomial of small degree (with very high probability). Let T be a tree-restriction of $p(x)$. For a leaf ℓ of the tree T , let p_ℓ denote the polynomial p restricted by the partial assignment given by the leaf ℓ .

Claim 3. *Let $p(x)$ be an n -variate s -sparse polynomial. There exists a tree-restriction T of depth $n/2$ such that $\Pr_{\ell \sim \mu_T} [\deg(p_\ell) > 10 \log s] \leq \exp(-\Omega(n))$.*

Proof. We will construct a sequence of trees $T_0, \dots, T_{n/2}$, where T_i has depth i and $T_{n/2}$ is the tree given by the claim. The tree T_0 is just a root and the tree T_{i+1} is an extension of the tree T_i , defined in the following manner. Set $t := 10 \log s$, and let B be the set of “bad” monomials of $p(x)$ of degree $> t$, where $|B| \leq s$. For a leaf ℓ of T_i , let B_ℓ be the set of monomials in B which remain non-zero under the partial assignment induced by ℓ . If $|B_\ell| = 0$ then ℓ will remain a leaf in T_{i+1} ; else, we extend it further. All monomials in B_ℓ have at least t variables; hence there exists some variable x_j which appears in at least $(t/n) \cdot |B_\ell|$ monomials. We extend ℓ by the two possible assignments to x_j . Let ℓ', ℓ'' be these new leaves of T_{i+1} . Clearly, $|B_{\ell'}|, |B_{\ell''}| \leq |B_\ell|$, and each monomial containing x_j belongs to exactly one of $B_{\ell'}, B_{\ell''}$. Thus, $\min(|B_{\ell'}|, |B_{\ell''}|) \leq (1 - \frac{t}{2n}) \cdot |B_\ell|$.

Let $T := T_{n/2}$. We conclude the proof by showing that a random leaf ℓ of T has $|B_\ell| > 0$ only with exponentially small probability, $\Pr_{\ell \sim \mu_T} [|B_\ell| \geq 1] \leq \exp(-\Omega(n))$. Note that if ℓ has depth less than $n/2$ then we must have $|B_\ell| = 0$ by definition. Thus, it suffices to bound the probability that a leaf has depth $n/2$. Let $v_0, v_1, \dots, v_r = \ell$ denote a random walk in the tree T from the root v_0 to a leaf v_ℓ of depth $r \leq n/2$. We define indicator variables X_0, \dots, X_{r-1} as follows: let u_i be the other child of v_i (i.e. not v_{i+1}). Then $X_i = 1$ if $|B_{v_{i+1}}| \leq |B_{u_i}|$. Clearly, X_0, \dots, X_{r-1} are independent (given r), and $\Pr[X_i = 1] \geq 1/2$. By our bound on $\min(|B_{\ell'}|, |B_{\ell''}|)$ above, if at least q of the variables X_0, \dots, X_{r-1} are

one, then $|B_\ell| \leq (1 - \frac{t}{2n})^q |B| \leq \exp(-tq/2n) \cdot s$. Thus, if $q > (2n/t) \cdot \log s = n/5$ then $|B_\ell| < 1$. The probability that a leaf ℓ at depth $n/2$ has $|B_\ell| > 0$ can thus be bounded by the probability that a random sequence of $n/2$ Bernoulli variables (with success probability $\geq 1/2$) contains $\leq n/5$ ones, which is $\exp(-\Omega(n))$.

We extend the tree T from Claim 3 to a complete tree of depth $n/2$, by extending leaves of depth $< \frac{n}{2}$ arbitrarily. For notational ease, consider a restricted polynomial $p_\ell(x)$ at a leaf ℓ of T as a polynomial in variables $x \in \{0, 1\}^{n/2}$. Claim 3 allows us to essentially consider polynomials of degree $\leq 10 \log s$, as by the triangle inequality $\mathbb{E}_{x \in \{0,1\}^n} [(-1)^{p(x)} \omega_3^{\sum x_i}] \leq \mathbb{E}_{\ell \sim \mu_T} |\mathbb{E}_{x \in \{0,1\}^{n/2}} [(-1)^{p_\ell(x)} \omega_3^{\sum x_i}]|$, and the probability that $\deg(p_\ell) > 10 \log s$ is $\exp(-\Omega(n))$.

We thus assume from now on that $p(x)$ is an n -variate s -sparse polynomial with $\deg(p) \leq 10 \log s$, and derive correlation bounds between $p(x)$ and the sum of the bits modulo 3. The degree of p is still too large to apply Theorem 2. So we reduce its degree further by restricting it to a smaller subset of the variables.

Let S_1, \dots, S_s denote the variables in the monomials of $p(x)$, that is $p(x) = \sum_{i=1}^s \prod_{j \in S_i} (x_j - a_{i,j})$, where $a_{i,j} \in \mathbb{F}_2$ are some coefficients. Let $R \subset [n]$ be a subset of the variables. We first claim that if R is large, but has small intersection with all monomials S_1, \dots, S_s , then we can use correlation bounds for low-degree polynomials to bound the correlation of $p(x)$ with the sum of the bits modulo 3.

Claim 4. *Let $R \subset [n]$ be a subset of the variables, such that $|R \cap S_i| \leq d$ for all $i = 1, \dots, s$. Then $\mathbb{E}_{x \in \{0,1\}^n} [(-1)^{p(x)} \omega_3^{\sum x_i}] \leq \exp(-\Omega(|R|/4^d))$.*

Proof. We partition an assignment $x \in \{0, 1\}^n$ to the variables inside and outside R , denoted $x_R \in \{0, 1\}^R$ and $x_{\bar{R}} \in \{0, 1\}^{[n] \setminus R}$. Note that for any assignment to $x_{\bar{R}}$, the remaining polynomial on the variables in R is of degree at most d . We denote for any assignment $a \in \{0, 1\}^{[n] \setminus R}$ this polynomial by $p_a(x_R) = p(x_R, a)$. By Theorem 2, $p_a(x_R)$ has small correlation with the sum of the bits modulo 3. By the triangle inequality, this gives $|\mathbb{E}_{x \in \{0,1\}^n} [(-1)^{p(x)} \omega_3^{\sum_{i=1}^n x_i}]| \leq \mathbb{E}_a |\mathbb{E}_{x_R} [(-1)^{p_a(x_R)} \omega_3^{\sum_{i \in R} x_i}]| \leq \exp(-|R|/4^d)$.

Thus, we need to find a relatively large set R with a small intersection with all the monomials. Let $s = n^c$, and let $d > 2c$ be chosen later. The next claim shows that a randomly chosen R of size roughly $n^{1-2c/d}$ has w.h.p intersection of size $\leq d$ with all monomials. We stress that since we just need to establish the existence of such an R , the probability that a random R satisfies the requirements can be (and indeed is) much weaker than the exponentially small correlation we seek. We omit the standard proof of the below claim.

Claim 5. $\exists R \subset [n]$ of size $|R| = n^{1-2c/d-o(1)}$ s.t. $|R \cap S_i| \leq d \forall i = 1, \dots, s$.

The proof of Theorem 1 now follows from Claim 3, Claim 4 and Claim 5 by an optimization of the parameters. For $d > 2c$ we have $|R| = n^{1-2c/d-o(1)}$ and $|\mathbb{E}_{x \in \{0,1\}^n} [(-1)^{p(x)} \omega_3^{\sum x_i}]| \leq \exp(-\Omega(n^{1-2c/d-o(1)}/4^d))$. To conclude the proof, note that if $c = o(\log n)$ then we can choose $d = o(\log n)$ such that $c/d = o(1)$ and $4^d = n^{o(1)}$.

4 Correlation Bounds for AC⁰ with a Few Symmetric and Threshold Gates

In this section, we prove our main correlation bounds. We describe the candidate hard functions and distributions w.r.t. which these functions are hard to predict.

For any $n, m, k, r \in \mathbb{N}$ such that $n = mkr$, let $f_{m,k,r} : \{0, 1\}^n \rightarrow \{0, 1\}$ be defined as follows: $f_{m,k,r}(x) = \bigoplus_{u=1}^m \bigwedge_{v=1}^k \bigoplus_{w=1}^r x_{u,v,w}$. This function was first defined by Razborov and Wigderson [21] to show $n^{\Omega(\log n)}$ lower bounds against the class MAJ \circ SYM \circ AND. Hansen and Miltersen [13] showed that this function in fact cannot be computed by MAJ \circ SYM \circ AC⁰ circuits of size $n^{o(\log n)}$. Viola further used this function (along with ideas from the work of Nisan and Wigderson [19]) to come up with pseudorandom generators against the class of AC⁰ circuits with at most $\varepsilon \log^2 n$ many symmetric gates for a small constant $\varepsilon > 0$. We show that for some m, k, r , the function $f_{m,k,r}$ is hard to compute on the average under the uniform distribution, by polynomial size AC⁰ circuits with $n^{1-o(1)}$ symmetric gates, or $n^{1/2-o(1)}$ threshold gates.

We call a function $g : \{0, 1\}^{mkr} \rightarrow \{0, 1\}$ *f_{m,k,r}-like* if there exist bits $b, b_{u,v} \in \{0, 1\}$ for each $(u, v) \in [m] \times [k]$ s.t. $g(x) = b \oplus \bigoplus_{u=1}^m \bigwedge_{v=1}^k (\bigoplus_{w=1}^r x_{u,v,w} \oplus b_{u,v})$.

Given integer $q \in \mathbb{N}$, the function MOD_q : $\{0, 1\}^n \rightarrow \{0, 1\}$ is defined as: MOD_q(x_1, \dots, x_n) is 0 if $\sum_{i=1}^n x_i \equiv 0 \pmod{q}$ and 1 otherwise. Let D_q denote the distribution induced on $\{0, 1\}^n$ by the following sampling procedure: pick a random $b \in \{0, 1\}$ and choose uniformly an input x such that MOD_q(x) = b .

4.1 Useful Lemmas

We need the following lemma due to Chattopadhyay and Hansen [8] (which is a refinement of a lemma due to Hastad and Goldmann [15]):

Lemma 1 (Chattopadhyay and Hansen [8]). *Let $g : \{0, 1\}^n \rightarrow \{0, 1\}$ be any boolean function that is computed by an ANY_t \circ SYM \circ AND_k circuit of size M . Then, for any partition of the n input variables of g into $k + 1$ sets, the NOF $(k + 1)$ -multiparty communication complexity of g is bounded by $O(kt \log M)$.*

We need a corollary of a result of Babai, Nisan, and Szegedy [3], that says that any $f_{m,k,r}$ -like function has low correlation with functions that have efficient k -party NOF protocols w.r.t. a suitable variable partition. We omit the proof.

Corollary 1 (follows from [3]). *Consider the variable partition $X = \{x_{u,v,w} \mid u \in [m], v \in [k], w \in [r]\}$ into sets X_1, \dots, X_k such that for each $j \in [k]$, we have $X_j = \{x_{u,j,w} \mid u \in [m], w \in [r]\}$. Let P be a k -party ε -error communication complexity protocol of complexity at most $\frac{1}{10}(m/4^k - \log(1/\gamma))$ bits computing a boolean function h , where the j th party gets as input bits the values of the variables in X_j . Then $\text{Corr}(h, g) \leq \varepsilon + \gamma$, for any function g that is $f_{m,k,r}$ -like.*

Finally, we need a lemma due to Ajtai that allows us to simplify a small collection of DNFs using a tree restriction of small height. Given a collection of DNFs \mathcal{F}

and a tree restriction T over a set of n boolean variables x_1, \dots, x_n , a leaf ℓ of T , and a $J \in \mathbb{N}$, we say that ℓ is J -restricting for \mathcal{F} if for each DNF $F \in \mathcal{F}$, the function $F|_\ell$ is a J -junta (i.e., depends only on J variables).

Lemma 2 (Ajtai [1]). *Fix any collection of k -DNFs \mathcal{F} of size M . For any $t \in \mathbb{N}$, there is a tree restriction T of height at most $nk \log M / (\log n)^t$ such that $\Pr_{\ell \sim \mu_T}[\ell \text{ is not } (\log n)^{10kt2^k} \text{-restricting for } \mathcal{F}] \leq \frac{1}{2^{n/(2^{10k}(\log n)^t)}}$.*

4.2 Correlation Bounds for Polynomial-size Circuits

We now prove our main lemma, which shows that the functions defined above have low correlation with $\text{ANY}_t \circ G \circ \text{AC}^0$ circuits, where t is reasonably small and $G \in \{\text{SYM}, \text{THR}, \text{MOD}_s\}$. We sketch the case of $G = \text{SYM}$, and defer the others to the full version.

Lemma 3. *Fix constants $d \in \mathbb{N}$ and $0 \leq \varepsilon \leq \frac{1}{10d}$. Also fix $q, s \in \mathbb{N}$ that are relatively prime. Let $c \leq \varepsilon \log \log n / 100$ and let $k = 10c/\varepsilon$. Let $g : \{0, 1\}^n \rightarrow \{0, 1\}$ denote the function $f_{n^{1-\varepsilon}/k, k, n^{\varepsilon d}}$. Let C be any $\text{ANY}_t \circ G \circ \text{AC}^0$ circuit of size $M \leq n^c$ and depth $d \geq 2$ where $G \in \{\text{SYM}, \text{THR}, \text{MOD}_s\}$. Then we have: (a) If $G = \text{SYM}$ and $t \leq n^{1-2\varepsilon d}$, then $\text{Corr}(g, C) \leq \exp\{-\Omega_d(n^{1-2\varepsilon d})\}$, (b) If $G = \text{THR}$ and $t \leq n^{0.5-2\varepsilon d}$, then $\text{Corr}(g, C) \leq \exp\{-\Omega_d(n^{0.5-2\varepsilon d})\}$, and (c) If $G = \text{MOD}_s$ and $t \leq n^{1-2\varepsilon d}$, then $\text{Corr}_{D_q}(\text{MOD}_q, C) \leq \exp\{-\Omega_d(n^{1-2\varepsilon d})\}$.*

Proof (Sketch). The proof proceeds as follows: we iteratively simplify the circuit C using Lemma 2 and random restrictions until we obtain a depth-3 $\text{ANY}_t \circ G \circ \text{AND}_{k-1}$ circuit (w.h.p.). Also, we argue that the function g remains “hard” in that it contains a copy of $f_{m', k, r'}$ for a reasonably large m' and $r' \geq 1$. We then apply either Lemma 1 to show that the function computed by the circuit C has an efficient k -party protocol. Then, Corollary 1 allows us to bound the correlation between the circuit C and the restricted version of the function g .

We now describe the simplification of the circuit as a tree restriction that we construct in $d - 2$ steps. At the beginning of the i th step, we will have a tree restriction T_i (initially empty). At “most” leaves of T_i , we will have simplified the circuit considerably — these leaves are labelled *good*. The remaining leaves we will label *bad*. The restriction T_i will satisfy the following properties:

- (P1) $\Pr_{\ell \sim \mu_{T_i}}[\ell \text{ is bad}] \leq (i - 1) \exp(\Omega_d(-n^{1-\varepsilon d}))$.
- (P2) Fix any good leaf ℓ of T_i . There is a circuit C^ℓ that computes the same function as $C|_\ell$ but has depth at most $d + 2 - i$ with bottom fanin $< k$. (At $i = 1$, we assume the circuit C has depth $d + 1$ with bottom fanin 1.) After Step $d - 2$, C^ℓ is a $\text{ANY}_t \circ G \circ \text{AND}_{k-1}$ circuit of size at most $M \cdot 2^k$.
- (P3) For a good leaf ℓ , let \mathcal{F}_ℓ denote the k -DNFs F such that either F or $\neg F$ (which is a k -CNF) appears as a depth-2 subcircuit of C^ℓ . We will ensure that $|\mathcal{F}_\ell| \leq M$ at each step by never adding to the number of gates at height 2 or more in the process of simplifying the circuit C .

(P4) For any good leaf ℓ , $g|_\ell$ is f_{m_ℓ, k, r_ℓ} -like, where $m_\ell \geq n^{1-\varepsilon d}/k2^{i-1}$, $r_\ell \geq n^{\varepsilon(d+1-i)}/4^{i-1}$. Let $n_\ell = m_\ell k r_\ell$, the number of variables that $g|_\ell$ depends on. Note that $n_\ell = \Omega_d(n^{1-\varepsilon(i-1)}) > \sqrt{n}$ for each $i \in [d-2]$.

Clearly, the initial (empty) restriction T_1 satisfies all the above properties. We now describe Step i for $i \in [d-2]$, which extends the tree restriction T_i at a given good leaf. Fix some good leaf ℓ of T_i . Step i has two stages:

Stage $i.1$: We apply Lemma 2 to circuit C^ℓ with $t = 3$ and obtain a tree restriction T of height at most $n_\ell k \log M / (\log n_\ell)^3 = O(n_\ell / \log n)$. We label a leaf ℓ' of T bad if it is not $(\log n)^{30k2^k}$ -restricting for \mathcal{F}_ℓ (these are the only leaves we will label bad in Step i). The tree T is such that $\Pr_{\ell' \sim \mu_T}[\ell' \text{ is bad}] \leq \exp(-n_\ell / (2^{10k} (\log n_\ell)^3)) \leq \exp(\Omega_d(-n^{1-\varepsilon d}))$. Fix a leaf ℓ' of T . If ℓ' happens to be bad, then we do not expand the restriction any further at ℓ' . Otherwise, we label ℓ' good and continue as follows. By renaming variables if necessary, let us assume that $g|_{\ell'}(x) = b \oplus \bigoplus_{u=1}^{m_\ell} \bigwedge_{v=1}^k (\bigoplus_{w=1}^{r_\ell} x_{u,v,w} \oplus b_{u,v})$ for some $b, b_{u,v} \in \{0, 1\}$. Since at most $O(n_\ell / \log n)$ variables have been set among the input variables to $g|_{\ell'}$ at leaf ℓ' , the fraction of u such that there are at least $r_\ell/2$ many variables $x_{u,v,w}$ that have been set by the restriction corresponding to ℓ' is at most $O(k / \log n) \leq 1/2$: call these u *over-restricted at ℓ'* . We extend the tree restriction at ℓ' by setting *all* (so far unset) variables $x_{u,v,w}$ such that u is over-restricted. Note that if u is *not* over-restricted, then for each $v \in [t]$, there are at least $r_\ell/2$ many w such that $x_{u,v,w}$ has not yet been set. In particular, we still have at least $m_\ell k r_\ell / 4 = n_\ell / 4$ variables not yet set. Call the new restriction (after having extended the restriction at each good leaf ℓ' as described above) T' . All leaves of T' that are descendants of good leaves of T are labelled good.

Stage $i.2$: We now apply a random restriction to the good leaves of T' to obtain our final tree restriction T'' at leaf ℓ of T_i . Fix some good leaf ℓ'' of T' . Let ℓ' denote the ancestor of ℓ'' among the leaves of T . Fix a subset S of the surviving variables of size n_ℓ / n^ε . We say that S is good for ℓ'' if (a) For each u that is not over-restricted at ℓ' and each $v \in [t]$, the number of w such that $x_{u,v,w} \in S$ is at least $r_\ell / 4n^\varepsilon$, and (b) Each $F \in \mathcal{F}_\ell$ is a k -junta restricted to the variables in the set S . It is easy to see that a random S of the required size is good for ℓ'' with good probability (say at least $\frac{1}{2}$). We omit the standard proof.

Fix a good S . For each u that was not over-restricted at ℓ' and $v \in [t]$, let $X_{u,v}$ denote a set of exactly $r_\ell / 4n^\varepsilon$ variables $x_{u,v,w} \in S$. Let $S' = \bigcup_{u,v} X_{u,v}$. We now apply the tree restriction T'' that fixes all the variables *outside* S' to all possible values. All leaves of T'' are labelled good. This ends Step i .

It is easy to see that properties (P1)-(P4) also hold for restriction T_{i+1} obtained in the manner outlined above. We just sketch the proof for Property (P2). At steps $i < d-2$, we construct the circuit $C^{\ell''}$ at any good leaf ℓ'' by writing each $F \in \mathcal{F}_\ell$ (which is a $k-1$ junta) as a $k-1$ CNF or DNF and collapsing the circuit C^ℓ . For $i = d-2$, we can write each F as a sum of at most 2^k disjoint AND_{k-1} gates and obtain a $\text{ANY}_t \circ G \circ \text{AND}_{k-1}$ circuit of size at most $M2^k$.

After Step $d-2$, we bound the correlation of g and C as follows. At any good leaf ℓ of T_{d-2} , the function $g|_\ell$ is $f_{m', t, r'}$ -like on some subset of the variables, where $m' \geq m/2^d$ and $r' \geq 1$. Also, at any good ℓ , the function $C|_\ell$ can be

computed by a circuit of the form $ANY_t \circ G \circ AND_{k-1}$ of size at most $M \cdot 2^k$. Since $G = \text{SYM}$, Lemma 1 says that for any variable partition, the function $C|_\ell$ can be computed by a deterministic k -party protocol using at most $t(k+1) \log(M \cdot 2^k) = o(m'/4^k)$ bits of communication. Using Corollary 1, we see that $\text{Corr}(C|_\ell, g|_\ell) \leq \exp\{-\Omega(m'/4^k)\} + \exp\{-\Omega(m'/k4^k \log n)\} \leq \exp\{-\Omega_d(m/(\log n)^2)\}$.

Finally, we have $\text{Corr}(g, C) \leq \mathbb{E}_{\ell \sim \mu_{T_d}}[\text{Corr}(g|_\ell, C|_\ell)] \leq \Pr_\ell[\ell \text{ bad}] + \mathbb{E}_\ell[\text{Corr}(g|_\ell, C|_\ell) | \ell \text{ good}] \leq \exp(-\Omega_d(n^{1-\varepsilon d})) + \mathbb{E}_\ell[\text{Corr}(g|_\ell, C|_\ell) | \ell \text{ good}]$. Along with the correlation bound at good leaves, this implies that $\text{Corr}(g, C) \leq \exp(-\Omega_d(n^{1-\varepsilon d})) + \exp\{-\Omega_d(m/(\log n)^2)\} \leq \exp(-\Omega_d(n^{1-\varepsilon(d+1)}))$, which proves the lemma when $G = \text{SYM}$. The cases $G \in \{\text{THR}, \text{MOD}_s\}$ are identical (except that we use either a communication protocol of Nisan [18] or a correlation bound of Bourgain [7] in place of Lemma 1) and are omitted.

We now prove similar correlation bounds for AC^0 circuits augmented with the same number of SYM , THR , or MOD_s gates, but now with no restriction on where these gates appear in the circuit. Given a gate type G (such as SYM , THR , etc.), we denote by $AC^0[G, t(n)]$ the class of constant-depth circuits augmented with at most $t(n)$ gates of type G .

Theorem 6. *Fix constants $d \in \mathbb{N}$ and $\varepsilon > 0$ such that $\varepsilon \leq 1/20d$. Let g be as defined in the statement of Theorem 3. Let C be any $AC^0[G, t]$ circuit of size at most $M = n^c$ and depth d , where $c \leq \varepsilon \log \log n / 200$ and $G \in \{\text{SYM}, \text{THR}, \text{MOD}_s\}$ for a fixed constant $s \in \mathbb{N}$. The following hold: (a) If $G = \text{SYM}$ and $t = o(n^{1-2\varepsilon(d+2)})$, then $\text{Corr}(g, C) \leq 2^{-\Omega_d(n^{1-2\varepsilon(d+2)})}$, (b) If $G = \text{THR}$ and $t = o(n^{0.5-2\varepsilon(d+2)})$, then $\text{Corr}(g, C) \leq 2^{-\Omega_d(n^{0.5-2\varepsilon(d+2)})}$, and (c) If $G = \text{MOD}_s$, $t = o(n^{1-2\varepsilon(d+2)})$, and q a (constant) number relatively prime to s , then $\text{Corr}_{D_q}(\text{MOD}_q, C) \leq 2^{-\Omega_d(n^{1-2\varepsilon(d+2)})}$.*

Proof. Let t denote the number of gates of type G in the circuit C . We proceed as in [8] and [24]. We first show that the circuit C can be written as a decision tree T of height t , where each node of the decision tree queries the value of a $G \circ AC^0$ circuit of size at most M . The statement will then follow from Lemma 3.

Consider the following procedure that computes the output of C on a given input x by querying the output of various $G \circ AC^0$ circuits on input x . Let V_1, \dots, V_t denote the G gates in the circuit C in topologically sorted order. Without loss of generality, we assume that V_t is the output gate: otherwise, we can always add a “dummy” G gate at the output and satisfy this property at the expense of increasing the depth to $d + 1$. Query the output of gate V_1 on input x and substitute the value in the circuit. Then query the output of gate V_2 in the modified circuit, and so on. Clearly, after t queries, we can obtain the output of the circuit on input x . This shows that we have a decision tree \mathcal{T} of depth t that queries functions computed by $G \circ AC^0$ circuits of size at most M and computes the same function as C .

Consider any accepting path p in the decision tree \mathcal{T} . The boolean function C_p that accepts exactly the inputs accepted by path p is computed by a $AND_t \circ G \circ AC^0$ circuit of size at most $Mt < n^{c+1}$ and depth at most $d + 2$. Since

different paths accept disjoint sets of inputs, it is easy to see that $(-1)^{C(x)} = \left(\sum_{\text{accepting paths } p} (-1)^{C_p(x)}\right) - (N - 1)$, where $N \leq 2^t$ is the total number of accepting paths of \mathcal{T} . Thus, for any distribution D and function f , we have

$$\begin{aligned} \text{Corr}_D(C, f) &= \left| \mathbb{E}_{x \sim D} [(-1)^{f(x)} \cdot (-1)^{C(x)}] \right| \\ &\leq \sum_{\text{accepting paths } p} \text{Corr}_D(f, C_p) + 2^t \cdot \text{Corr}_D(f, 0) \end{aligned} \tag{1}$$

where 0 represents the all-zeroes function. Since each C_p and of course also the all-zeroes function can be computed by small $\text{ANY}_t \circ G \circ \text{AC}^0$ circuits of depth at most $d + 2$, we can use Lemma 3 to bound the correlation of C with our chosen hard function f . We illustrate this in the case that $G = \text{SYM}$; the cases $G = \text{THR}$ and $G = \text{MOD}_s$ are almost identical and hence omitted. If $G = \text{SYM}$, by Lemma 3, we have for each p , $\text{Corr}(g, C_p) \leq \exp\{-\Omega_d(n^{1-2\varepsilon(d+2)})\}$ (and similarly for the all-zeroes function). Hence, by (1), $\text{Corr}(g, C) \leq 2^{t+1} \cdot \exp\{-\Omega_d(n^{1-2\varepsilon(d+2)})\} = \exp\{-\Omega_d(n^{1-2\varepsilon(d+2)})\}$. Hence, we are done.

We note that the above theorem has some application to pseudorandom generator (PRG) constructions for AC^0 circuits augmented with a few SYM and THR gates. Though the seedlength of our generator is poorer than that obtained from Viola’s construction [24], our construction works for a larger number of SYM and THR gates, and also gives exponentially small error. The proof is based on the generic construction of Nisan and Wigderson [19], and is omitted.

Corollary 2. *Fix any constants $d \in \mathbb{N}$ and $\delta, \eta > 0$. Assume $c = c(n) = o(\log \log n)$. Then there exist polynomial-time computable functions $\mathcal{G}_{\text{SYM}} : \{0, 1\}^{n^\delta} \rightarrow \{0, 1\}^n$ and $\mathcal{G}_{\text{THR}} : \{0, 1\}^{n^\delta} \rightarrow \{0, 1\}^n$ such that \mathcal{G}_{SYM} is a PRG for the class of $\text{AC}^0[\text{SYM}, n^{1-\eta}]$ circuits of size n^c and depth d with error at most $\exp\{-n^{\delta(1-\eta)/2}\}$ and \mathcal{G}_{THR} is a PRG for the class of $\text{AC}^0[\text{THR}, n^{0.5-\eta}]$ circuits of size n^c and depth d with error at most $\exp\{-n^{\delta(1-\eta)/4}\}$.*

References

1. Ajtai, M.: Σ_1^1 -formulae on finite structures. *Annals of Pure and Applied Logic* 24, 1–48 (1983)
2. Aspnes, J., Beigel, R., Furst, M., Rudich, S.: The expressive power of voting polynomials. *Combinatorica* 14(2), 1–14 (1994)
3. Babai, L., Nisan, N., Szegedy, M.: Multipartite protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. Comput. Syst. Sci.* 45(2), 204–232 (1992)
4. Barrington, D.A.M., Straubing, H.: Complex polynomials and circuit lower bounds for modular counting. In: *Computational Complexity*, pp. 314–324 (1994)
5. Beigel, R.: When do extra majority gates help? Polylog(n) majority gates are equivalent to one. In: *Computational Complexity*, pp. 314–324 (1994)
6. Beigel, R., Reingold, N., Spielman, D.: The perceptron strikes back. In: *Proceedings of the 6th Conference on Structure in Complexity Theory*, pp. 286–291 (1991)

7. Bourgain, J.: Estimation of certain exponential sums arising in complexity theory. *Comptes Rendus Mathematique* 340(9), 627–631 (2005)
8. Chattopadhyay, A., Hansen, K.A.: Lower bounds for circuits with few modular and symmetric gates. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) *ICALP 2005*. LNCS, vol. 3580, pp. 994–1005. Springer, Heidelberg (2005)
9. Chattopadhyay, A., Hansen, K.A.: Lower bounds for circuits with few modular and symmetric gates. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) *ICALP 2005*. LNCS, vol. 3580, pp. 994–1005. Springer, Heidelberg (2005)
10. Goldmann, M., Hastad, J., Razborov, A.: Majority gates vs. general weighted threshold gates. In: *Proceedings of the 7th Structure in Complexity Theory Annual Conference* (1992)
11. Gopalan, P., Servedio, R.A.: Learning and lower bounds for AC^0 with threshold gates. In: *APPROX-RANDOM*, pp. 588–601 (2010)
12. Hansen, K.A.: Lower bounds for circuits with few modular gates using exponential sums. *Electronic Colloquium on Computational Complexity (ECCC)* 13(079) (2006)
13. Hansen, K.A., Miltersen, P.B.: Some meet-in-the-middle circuit lower bounds. In: Fiala, J., Koubek, V., Kratochvíl, J. (eds.) *MFCS 2004*. LNCS, vol. 3153, pp. 334–345. Springer, Heidelberg (2004)
14. Hastad, J.: Almost optimal lower bounds for small depth circuits. In: *Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC 1986)*, pp. 6–20. ACM, New York (1986)
15. Håstad, J., Goldmann, M.: On the power of small-depth threshold circuits. *Computational Complexity* 1, 113–129 (1991)
16. Kushilevitz, E., Nisan, N.: *Communication Complexity*. Cambridge University Press, Cambridge (1997)
17. Nisan, N.: Pseudorandom bits for constant depth circuits. *Combinatorica* 11(1), 63–70 (1991)
18. Nisan, N.: The communication complexity of threshold gates. In: *Proceedings of “Combinatorics, Paul Erdos is Eighty”*, pp. 301–315 (1994)
19. Nisan, N., Wigderson, A.: Hardness vs. randomness. *J. Comput. Syst. Sci.* 49(2), 149–167 (1994)
20. Razborov, A.A.: Lower bounds for the size of circuits of bounded depth with basis $\{\&, \oplus\}$. *Math. Notes Acad. Sci. USSR* 41(4), 333–338 (1987)
21. Alexander, A.R., Wigderson, A.: $n^{\Omega(\log n)}$ lower bounds on the size of depth-3 threshold circuits with and gates at the bottom. *Inf. Process. Lett.* 45(6), 303–307 (1993)
22. Siu, K.I., Bruck, J.: On the power of threshold circuits with small weights. *SIAM Journal on Discrete Mathematics* 4(3), 423–435 (1991)
23. Smolensky, R.: Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In: *Proceedings of the 19th Annual ACM Symposium on Theory of Computing (STOC 1987)*, pp. 77–82. ACM, New York (1987)
24. Viola, E.: Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM J. Comput.* 36(5), 1387–1403 (2007)
25. Viola, E., Wigderson, A.: Norms, xor lemmas, and lower bounds for polynomials and protocols. *Theory of Computing* 4(1), 137–168 (2008)
26. Williams, R.: Non-uniform ACC circuit lower bounds. In: *Conference on Computational Complexity* (2011)

Clustering in Interfering Binary Mixtures

Sarah Miracle*, Dana Randall**, and Amanda Pascoe Streib***

Georgia Institute of Technology, Atlanta GA 30332, USA
{sarah.miracle@cc, randall@cc, apascoe3@math}.gatech.edu

Abstract. Colloids are binary mixtures of molecules with one type of molecule suspended in another. It is believed that at low density typical configurations will be well-mixed throughout, while at high density they will separate into clusters. We characterize the high and low density phases for a general family of discrete *interfering binary mixtures* by showing that they exhibit a “clustering property” at high density and not at low density. The clustering property states that there will be a region that has very high area to perimeter ratio and very high density of one type of molecule. A special case is mixtures of squares and diamonds on \mathbb{Z}^2 which correspond to the Ising model at fixed magnetization.

Keywords: discrete colloids, Ising model, phase separation, Peierls argument, equilibrium distribution.

1 Introduction

Colloids are mixtures of two types of molecules in suspension where all non-overlapping arrangements are equally likely. When the density of each type of molecule is low, the mixtures are homogeneous and consequently exhibit properties that make them suitable for many industrial applications, including fogs, gels, foods, paints, and photographic emulsions (see, e.g., [1], [2]). In contrast, when the density is high, the two types of molecules separate whereby one type appears to cluster together. Although this behavior is similar to phase transitions that occur in other discrete models, such as the Ising and Potts models, here the two types of molecules do not possess any *enthalpic* forces causing like particles to attract or disparate particles to repel. In contrast, the behavior of colloids is purely *entropic* — the only restriction is a “hard-core” constraint requiring objects to remain in non-overlapping positions, and clustering occurs at high density because the overwhelming majority of configurations in the stationary distribution are believed to exhibit such a separation. While the experimental study of colloids is pervasive in surface chemistry, material science, physics, and nanotechnology, there has been little rigorous work explaining their behavior.

* Supported in part by NSF CCF-0830367 and a DOE Office of Science Graduate Fellowship.

** Supported in part by NSF CCF-0830367 and CCF-0910584.

*** Supported in part by a National Physical Sciences Consortium Fellowship, a Georgia Institute of Technology ACO Fellowship and NSF CCF-0910584.

Even running simulations has been challenging because local algorithms will be slow to converge at high density. Dress and Krauth [7] introduced an algorithm to try to overcome this obstacle, but this too was shown to require time exponential in the number of molecules in some cases [14]. Nonetheless, their algorithm seems to be well-behaved in practice, and Buhot and Krauth [3] provided simulations showing strong heuristic evidence of the presence of two distinct phases in colloid models consisting of different sized squares.

Frenkel and Louis [9] studied an interesting discrete model of colloids whose behavior can be related to the Ising model, a standard model of ferromagnetism. Their model consists of mixtures of unit squares in a region of \mathbb{Z}^2 and diamonds of area $1/2$ that sit on lattice edges (see Fig. 1). They show that this colloid model, which we call **Model 1**, corresponds to an Ising model, where the density of squares fixes the magnetization and the density of diamonds determines the temperature (see Section 2.1). The Ising model at low temperature is known to exhibit clustering of positive spins. In fact the precise limiting shape of the cluster known as the Wulff shape has been extensively studied using sophisticated techniques (see, e.g. [5], or the references therein). **Model 1** then inherits the phase transition arising in the Ising model which shows there will be clustering at high densities [13]. In this paper we study clustering using elementary methods that apply to a large class of natural colloid models. We characterize clustering directly in terms of the parameters arising from the model to distinguish between the high and low phases and understand the role the density of each type of molecule plays.

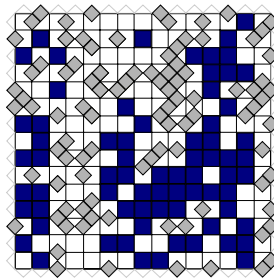


Fig. 1. Model 1, squares and diamonds on the $n \times n$ grid L_n

We consider a class of *interfering binary mixtures*. Let (Λ_A, Λ_B) be a pair of planar lattices such that a face of Λ_A and a face of Λ_B are either disjoint, intersect at a single vertex, or intersect at a simply-connected region that is isomorphic to a fixed shape s with nonzero area. For example, in **Model 1**, Λ_A is the Cartesian lattice \mathbb{Z}^2 and Λ_B is the set of diamonds bisected by edges in \mathbb{Z}^2 ; then s is an isosceles triangle with unit base and height $1/2$ (Fig. 1). We consider the intersection of these lattices with some finite region L , where $L_A = \Lambda_A \cap L$ and $L_B = \Lambda_B \cap L$. We are given a set of tiles; A -tiles lie on the faces of L_A and B -tiles lie on the faces of L_B with the additional requirement that tiles must not overlap. In Section 5, we will give examples of other interfering binary mixtures,

including independent sets, that arise naturally in combinatorics and statistical physics and contrast these with a non-interfering binary mixture that provably does not exhibit clustering.

It is often useful to switch from a model where the number of tiles of each type are fixed to a so-called *grand canonical ensemble* where these are allowed to vary. Here, however, typical configurations would have a preponderance of only one type of tile at most high densities and the balanced configurations we are interested in would be exponentially unlikely. Instead, we fix the number of A -tiles and allow the B -tiles to vary stochastically. Each configuration σ has weight proportional to $\lambda^{d(\sigma)}$, where $d(\sigma)$ is the number of B -tiles in σ . The choice of λ controls the expected density of B -tiles.

Our goal now is to understand when clustering occurs in terms of the (expected) density of each type of tile. First we define a clustering property for configurations of tiles. Informally we have clustering if there exists a dense region R in Λ_A with $\Omega(n^2)$ area and $O(n)$ perimeter. Our main theorems demonstrate that at high density interfering binary mixtures exhibit the clustering property while at low densities they do not. We give precise definitions of the clustering property and state the main theorems in Section 2. In Sections 3 and 4 we prove the two main theorems in the context of **Model 1** and in Section 5 we explain the generalization to other interfering binary mixtures.

The key tools in our proofs are careful *Peierls arguments*, used in statistical physics to study uniqueness of the Gibbs state and phase transitions (see, e.g., [4], [6]), and in computer science to study slow mixing of Markov chains (see, e.g., [2], [10], [15]). Peierls arguments allow you to add and remove contours by complementing the interiors of those contours. The main challenge here is maintaining the number of A -tiles, making the arguments considerably more difficult. We introduce the concept of bridge systems, to handle multiple contours by connecting components and make it possible to efficiently encode the boundaries of all contours removed. The encoding is necessary to account for the entropy/energy tradeoffs in these maps.

2 Binary Mixtures and the Clustering Property

We begin by formalizing the model, defining clustering and stating our main theorems.

2.1 Interfering Binary Mixtures

Recall A -tiles lie on faces of $L_A = \Lambda_A \cap L$ and $|L_A|$ is the total number of faces of L_A . Given constants $\lambda > 1$, and $0 < b < 1/2$, where $b|L_A| \in \mathbb{Z}$, define $\Omega = \Omega(b, \lambda)$ as the set of non-overlapping packings of L with $b|L_A|$ A -tiles and any number of B -tiles (where a tile can only be placed on a face of its type). We wish to study the distribution $\pi(\rho) = \lambda^{d(\rho)}/Z$, where $d(\rho)$ is the number of B -tiles in ρ and $Z = \sum_{\rho \in \Omega} \lambda^{d(\rho)}$ is a normalizing constant. Our goal is to determine whether a configuration chosen according to π is likely to have clusters of A -tiles.

In Sections 2-4, we study **Model 1**, and in Section 5, we generalize the techniques to other models of interfering binary mixtures. We start by defining the Ising model on the $n \times n$ grid L_n and explaining the equivalence with **Model 1**. Let $\overline{G} = (\overline{V}, \overline{E})$ be the dual lattice region to L_n and let $\rho \in \{+, -\}^{\overline{V}}$ be an assignment of spins to each of the vertices in \overline{V} (i.e., the faces in V). The weight of a configuration is $\overline{\pi}(\rho) = e^{\beta|\overline{E}_d(\rho)|}/\overline{Z}$, where $\overline{E}_d(\rho) \subseteq \overline{E}$ is the set of edges in \overline{G} whose endpoints have different spins in ρ , β is inverse temperature and \overline{Z} is the normalizing constant.

For **Model 1**, given a configuration ρ in Ω , let the *square structure* $\Gamma(\rho)$ be the configuration σ obtained from ρ by removing all of its B -tiles (diamonds). We consider the set $\widehat{\Omega}$ of all such square structures with bn^2 A -tiles (squares). Let $\widehat{\pi}$ be the induced distribution on $\widehat{\Omega}$; that is, for $\sigma \in \widehat{\Omega}$, let $\widehat{\pi}(\sigma) = \sum_{\rho \in \Gamma^{-1}(\sigma)} \overline{\pi}(\rho)$. For σ in Ω or $\widehat{\Omega}$, define the *perimeter* of σ to be the edges that belong to exactly one A -tile in σ , and define $\kappa(\sigma)$ as the length of the perimeter of σ . Let $e(\sigma)$ be the number of edges that are not incident to any A -tile in σ . We find that

$$\widehat{\pi}(\sigma) = \sum_{k=0}^{e(\sigma)} \frac{\lambda^k}{Z} \binom{e(\sigma)}{k} = \frac{1}{Z} (1 + \lambda)^{e(\sigma)} = (1 + \lambda)^{2n^2 - 2bn^2} \frac{\mu^{\kappa(\sigma)}}{Z}, \tag{1}$$

where $\mu = (1 + \lambda)^{-\frac{1}{2}}$. Thus, the total perimeter of the square structure completely determines the probability that it will show up in Ω . This directly implies the equivalence with the Ising Model: give a face f of L_n a positive spin if there is an A -tile on f and a negative spin otherwise. Since the weight of a configuration is determined exactly by the number of edges with opposite spins in L_n , this is the Ising model with a fixed number of positive spins for some λ that is a function of β , known as *fixed magnetization*.

2.2 The Clustering Property

The goal of this paper is to show that when the density of B -tiles is high, interfering binary mixtures cluster, while at low density they do not. First, we characterize clustering in this context. Intuitively, a configuration has the clustering property if there is a large region densely filled with A -tiles. More precisely, let a *region* R be any set of faces in L_n . Its perimeter, $\kappa(R)$ is the number of edges adjacent to a face in R and a face in $\overline{R} = L_n \setminus R$. Let $c = \min \left\{ \frac{b}{2}, \frac{1}{100} \right\}$.

Definition 1. We say that a configuration $\sigma \in \Omega$ (or $\Gamma(\sigma) \in \widehat{\Omega}$) has the clustering property if it contains a region R which satisfies the following properties:

1. R contains at least $(b - c)n^2$ A -tiles,
2. the perimeter of R is at most $8\sqrt{b}n$, and
3. the density of A -tiles in R is at least $1 - c$ and in \overline{R} is at most c .

If a configuration has the clustering property, we show that it contains an $n^{1/3} \times n^{1/3}$ window with high density and one with low density, demonstrating the heterogeneity of the configuration. In Section 5.2 we contrast this with **Model 4**, related to bond percolation, which remains homogeneous at all densities.

2.3 Main Results

We show that at high density interfering binary mixtures have the clustering property while at low densities they do not. Specifically, we prove the following theorems in the context of **Model 1** on the $n \times n$ region L_n with bn^2 A -tiles and the density of B -tiles determined by λ . In Section 5, we show they also hold for other interfering binary mixtures.

Theorem 1. *For $0 < b \leq 1/2$, there exist constants $\lambda^* = \lambda^*(b) > 1, \gamma_1 < 1$ and $n_1 = n_1(b)$ such that for all $n > n_1, \lambda \geq \lambda^*$ a random sample from Ω will have the clustering property with probability at least $(1 - \gamma_1^n)$.*

Theorem 2. *For $0 < b < 1/2$, there exist constants $\lambda_* = \lambda_*(b) > 0, \gamma_2 < 1$ and $n_2 = n_2(b)$ such that for all $n > n_2, \lambda \leq \lambda_*$ a random sample from Ω will not have the clustering property with probability at least $(1 - \gamma_2^n)$.*

Furthermore, it follows from the proofs that at low density if a dense region R' has area $\Omega(n^2)$ then it must have perimeter $\Omega(n^2)$. Notice that in the case $b > 1/2$ we can obtain comparable results by the symmetry of the A -tiles to the empty space. Indeed, in this case if λ is sufficiently high we will see empty cells clustering within a sea of A -tiles and for low density the empty cells will be well-distributed.

Note that since clustering is just a property of the A -tiles, it suffices to prove Theorems 1 and 2 for weighted square structures $\widehat{\Omega}$, involving just the A -tiles. From this point we focus on $\widehat{\Omega}$, and we refer to A -tiles just as *tiles*.

3 High Density of B -tiles

We concentrate first on interfering binary mixtures at high density to prove Theorem 1. Define $\Psi \subset \widehat{\Omega}$ to be the set of configurations that have the clustering property; then we show that $\widehat{\pi}(\widehat{\Omega} \setminus \Psi) \leq \gamma_1^n \widehat{\pi}(\Psi)$ for some constant $\gamma_1 < 1$. To achieve this, we apply a Peierls argument, in which we define a map $f : \widehat{\Omega} \setminus \Psi \rightarrow \Psi$ and show that for all $\tau \in \Psi$,

$$\sum_{\sigma \in f^{-1}(\tau)} \widehat{\pi}(\sigma) \leq \gamma_1^n \widehat{\pi}(\tau). \tag{2}$$

Given a configuration $\sigma \in \widehat{\Omega} \setminus \Psi$, the map f removes a large set T of tiles in σ and reassembles them in a single large component in $f(\sigma)$. This decreases the total perimeter of the configuration significantly, and therefore $\widehat{\pi}(f(\sigma))$ is exponentially larger than $\widehat{\pi}(\sigma)$. The challenge is to bound the number of configurations that map to a given $\tau \in \Psi$ by carefully encoding the preimages of τ .

Some definitions will be helpful. We say two tiles are *adjacent* if their borders share an edge. A *component* is a maximal connected set of tiles, and maximal connected segments of the perimeter of σ are *contours*. The set T of tiles we remove will be a union of components, which we identify using a system of

“bridges” connecting these components (Fig. 2). The key is that the number of edges in the bridges is at most a constant times the total perimeter of the components bridged. Then if E is the set of all edges in bridges or along contours bridged, we can bound $|f^{-1}(\tau)|$ by the number of ways that those E edges could be distributed in σ . Finally, we show that there is a sparse, roughly square region in the resulting configuration where we can add the T tiles. We complement that region to obtain $f(\sigma)$.

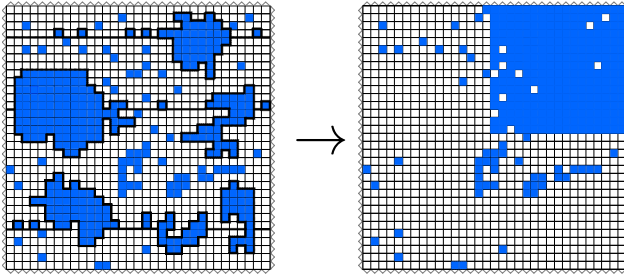


Fig. 2. A configuration $\sigma \in \widehat{\Omega} \setminus \Psi$ and the image $f(\sigma)$ of σ in Ψ

Building Bridges. Given a region R , let $C(R)$ be the set of contours fully contained within the interior of R and define the *outer contours* to be those in $C(R)$ that are not contained in the interior of other contours in $C(R)$. The interior of the outer contours of components are called *holes* and the interior of the outer contours of holes are called *islands*.

Consider first the case in which there are no components with holes. Suppose B is a set of edges of L_n connecting some subset S of the contours to the boundary of L_n . We call B a set of *bridges* and S a set of *bridged contours*. A cell in L_n or a tile is called *unbridged* if it is not bounded by a bridged contour. Then (B, S) is a **c -bridge system** for $\sigma \in \widehat{\Omega}$ if the number of unbridged tiles is at most c times the number of unbridged cells, and $|B| \leq (1 - c)/(2c)\kappa(S)$. If σ has components with holes, then first construct a c -bridge system (B, S) for σ' , obtained from σ by filling all the holes. Next for each bridged contour X in σ , construct a c -bridge system for the region bounded by X (treating tiles as empty cells and empty cells as tiles). Recurse until you obtain c -bridge systems for each bridged contour at every level of the recursion. We call this a c -bridge system of σ . We defer the details to the full version of the paper.

Lemma 1. *There exists a c -bridge system for any configuration $\sigma \in \widehat{\Omega}$.*

Proof. We may assume that σ has no holes, since otherwise we recurse as described above. Now we use induction on the number of contours in R . If there are no contours, then clearly (\emptyset, \emptyset) is a c -bridge system for R . Otherwise, define $t(R)$ to be the tiles in R , and $x(R)$ is the number of empty cells in R . Let \mathbb{H} be the set of horizontal lines through R . For every $H \in \mathbb{H}$, if $|t(R) \cap H| < c|R \cap H|$ then we are done, since then (\emptyset, \emptyset) is a c -bridge system for R . Otherwise there exists

a horizontal line H such that $|t(R) \cap H| \geq c|R \cap H|$. Then let B be the set of bottom edges of every outer cell in $H \cap R$. See Fig. 3, where the dark black edges along the line H are the new bridges. Let S be the set of contours connected in this step. We know that $\kappa(S) \geq 2|t(R) \cap H| \geq 2c|R \cap H| \geq 2c/(1-c)|x(R) \cap H|$, so $|B| \leq (1-c)/(2c)\kappa(S)$. We obtain R' from R by removing the cells bounded by a contour in S , as in Fig. 3. Then by induction, there exists a c -bridge system (B', S') of R' . Then $\widehat{B} := B \cup B'$ is a set of bridges connecting the contours in $\widehat{S} = S \cup S'$ to each other and to the boundary of R . Moreover, $|\widehat{B}| \leq \frac{1-c}{2c} \kappa(\widehat{S})$ and the number of unbridged tiles is at most c times the number of unbridged cells. Hence $(\widehat{B}, \widehat{S})$ is a c -bridge system for R . \square

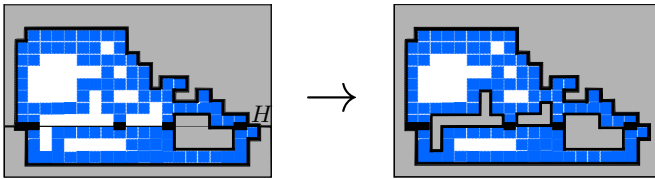


Fig. 3. Before and after one step of the construction of a c -bridge system for a region R ; the solid lighter grey area is exterior to R

Once we have a c -bridge system, we can apply a map in which we *complement* an entire region of cells, making tiled cells empty and vice versa. This map significantly reduces the perimeter, but can dramatically change the total number of tiles. Recall we must maintain the total number of tiles, so we may need to supplement by adding extra tiles from another region or we may have extra tiles, which we will put in our “bank” for later. At the end of the process we will find a roughly square region that we can again complement using the bank of extra tiles so that the total number of tiles is restored to bn^2 at minimal cost.

Finding a Sparse Box. We now show that after removing all but cn^2 tiles, there exists a region of low density where we can place the tiles in our bank.

Lemma 2. *For $(b - c)n^2 \leq a < bn^2$, there exists a constant $n_3 = n_3(b)$ such that for all $n \geq n_3$, if ρ is a configuration with at most cn^2 tiles then ρ contains a roughly square region R' such that complementing R' requires a additional tiles and the change in total perimeter is at most $5\sqrt{a}$.*

Proof. Given a region R , let $d(R)$ denote the number of tiles needed to complement R ; this is exactly the area of R minus twice the number of tiles in R . Let $l = \lceil \sqrt{8a/7} \rceil$. First we show that there exists a square $l \times l$ region R such that $d(R) \geq a$. Assume that such a square does not exist. Divide the grid into $\lfloor \frac{n}{l} \rfloor^2$ disjoint squares with side length l and consider any square. Let t be the number of tiles in the square. The empty volume is at least $l^2 - t$. By assumption each square satisfies $l^2 - t < t + a$, and so $t > \frac{l^2 - a}{2}$. In particular, $8a/7 \leq l^2 < a + 2cn^2$,

so we may assume that $a < 14cn^2$. This implies that $l \leq \sqrt{8a/7} + 1 \leq 1 + 4\sqrt{cn}$. However, if T is the total number of tiles,

$$cn^2 \geq T > \left\lfloor \frac{n}{l} \right\rfloor^2 \frac{l^2 - a}{2} \geq \frac{n^2}{2} \left(1 - \frac{l}{n}\right)^2 \left(1 - \frac{a}{l^2}\right) > \frac{n^2 \left(1 - \frac{1}{n} - 4\sqrt{c}\right)^2}{16} \geq cn^2,$$

since $c \leq \frac{1}{65}$ and $n \geq n_3$, a contradiction. Therefore there exists an $l \times l$ square R such that $d(R) \geq a$. Remove cells from R one at a time, starting with the bottom row of R and moving across, until we obtain a region $R' \subseteq R$ with $d(R') = a$. This can be done because removing one cell at a time changes d by at most 1. This region R' is roughly square and has perimeter at most $4\sqrt{8a/7} < 5\sqrt{a}$. \square

The Proof of Theorem 11. Finally we can prove Theorem 11, showing that for large λ a typical configuration will have the clustering property.

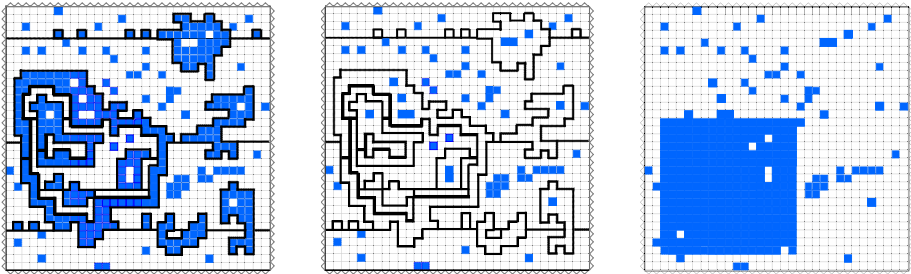


Fig. 4. A c -bridge system for $\sigma \in \widehat{\Omega} \setminus \Psi$; the image $f_1(\sigma)$; and $f(\sigma) = f_2 \circ f_1(\sigma)$

Proof of Theorem 11. Let $\sigma \in \widehat{\Omega} \setminus \Psi$. Construct a c -bridge system (B, S) for L_n as described in Lemma 11. That is, (B, S) is a set of bridges in L_n connecting some of the components, some of the holes within those components, some of the islands within those holes, etc. For any bridged contour X , let $r(X)$ be the region bounded by X . If $r(X)$ is a component with holes, then we remove all outer tiles of $r(X)$ and complement all unbridged holes in X , using a subset of the tiles removed to fill in the holes. If $r(X)$ is a hole with islands, then we leave all of the unbridged islands alone. At this point, after complementing some number of regions, we have a bank of extra tiles; let a be the number of tiles in the bank. Notice that by the definition of a c -bridge system, the density of tiles remaining is at most c , so $a \geq (b - c)n^2$.

Let $f_1(\sigma)$ be obtained from σ by removing the bridged components and complementing as described above. Let \mathcal{F}_1 be the image of f_1 on $\widehat{\Omega} \setminus \Psi$; note that $\mathcal{F}_1 \not\subseteq \widehat{\Omega}$ since the configurations in \mathcal{F}_1 have too few tiles. Let κ be the total perimeter of all contours bridged. Then for any $\rho \in \mathcal{F}_1$, we claim that the number of preimages of ρ whose bridged contours have total perimeter κ is at most $5c^3$ for $c_3 = \left(1 + \frac{1-c}{2c} + \frac{1}{c_2}\right)\kappa$. Consider the c -bridge system obtained above for L_n .

Let V denote the leftmost vertical edges of the region. Let $S' = S \cup V$. We perform what is essentially a depth-first-search traversal of the bridge system on S' , starting at the top left corner of L_n . As we traverse an edge we record what type of edge it was using five bits that represent forward, left, right, bridge east, or bridge west (see full version for details). Given the encoded information, there is a unique way to distribute the contours. Hence for all perimeters $\kappa \geq c_2 n$ the number of preimages of ρ whose bridged contours have total perimeter κ is at most $5^{|B|+\kappa+n} \leq 5^{c_3}$. Therefore $|f_1^{-1}(\rho)| \leq \sum_{\kappa \geq c_2 n} 5^{c_3}$.

Let $\rho \in \mathcal{F}_1$ with $bn^2 - a$ tiles. Lemma 2 shows how to find a region S' in ρ to complement using the a tiles from the bank to obtain τ in such a way that $\kappa(\tau) - \kappa(\rho) \leq 5\sqrt{a}$. Let $f_2(\rho) = \tau$ and $f = f_2 \circ f_1$. We can encode the boundary of S' with $n^2 3^{\kappa(S')}$ information. Hence for any $\tau \in \Psi$,

$$|f^{-1}(\tau)| \leq n^2 3^{5\sqrt{a}} \max_{\rho \in f_2^{-1}(\tau)} |f_1^{-1}(\rho)|.$$

Let $\sigma \in \widehat{\Omega} \setminus \Psi$, and as above let κ be the total perimeter of components bridged in σ (recall $\kappa(\sigma)$ is the total perimeter of all contours in σ). If $\kappa \leq 8\sqrt{a}$, then $\sigma \in \Psi$, a contradiction. To see this, define the parity of a cell to be 1 if it is contained within an odd number of bridged contours and 0 otherwise, and let R be the set of cells with parity 1. Then R has density at least $1 - c$, perimeter at most $8\sqrt{a}$ and $a \geq (b - c)n^2$ tiles. Moreover, \overline{R} has density at most c . Thus R is the region we require, and so $\sigma \in \Psi$. This implies $\kappa > 8\sqrt{a}$. We have shown that $\kappa(\sigma) - \kappa(f(\sigma)) > \kappa - 5\sqrt{a} > \kappa/4$. Let $\tau \in \Psi$ and define $f_\kappa^{-1}(\tau) \leq n^2 \left(3^{\frac{1}{2\sqrt{7}}} 5^{1+\frac{1-c}{2c}+\frac{1}{16\sqrt{b}}} \right)^\kappa$ to be the set of configurations with perimeter κ that map to τ . Then

$$\pi(\tau)^{-1} \sum_{\sigma \in f^{-1}(\tau)} \pi(\sigma) \leq \sum_{\sigma \in f^{-1}(\tau)} \mu^{\kappa(\sigma)-\kappa(f(\sigma))} \leq \sum_{\kappa=8\sqrt{a}}^{2n^2} \mu^{\kappa/4} |f_\kappa^{-1}(\tau)| \leq \gamma_1^n,$$

for some $\gamma_1 < 1$, if $\mu \leq \mu^* < \left(3^{\frac{1}{2\sqrt{7}}} 5^{1+\frac{1-c}{2c}+\frac{1}{16\sqrt{b}}} \right)^{-4}$. Thus the theorem holds if $\lambda \geq \lambda^* = \mu^{*-2} - 1$. □

As a corollary, we find that if a configuration has the clustering property then there exists an $n^{1/3} \times n^{1/3}$ window with high density and one with low density. We defer this straightforward proof to the full version of the paper.

Corollary 1. *For $0 < b \leq 1/2$ there exists a constant $n_4 = n_4(b)$ such that for all $n > n_4$, if σ satisfies the clustering property then σ contains square $n^{1/3} \times n^{1/3}$ windows W_1 and W_2 such that the density of tiles in W_1 is at least $.99(1 - c)$ and the density of tiles in W_2 is at most $2.1c$.*

4 Low Density of B-tiles

We now examine the low density case and prove Theorem 2, stating that typical configurations will not have the clustering property. For small enough λ , the

A -tiles will be well-distributed throughout L_n , in the following sense. Any large dense region must have perimeter on the order of n^2 .

Proof of Theorem 2. Define $\delta = ((1 - c)/(b - c))^{b-c}$. Let $\Psi' \subset \widehat{\Omega}$ be the set of configurations with a region R that have density at least $1 - c$, at least $(b - c)n^2$ tiles, and perimeter less than αn^2 , where α satisfies $0 < \alpha < (\ln(\delta) - b \ln 2)/((1 + 1/c) \ln 5)$. We will show $\widehat{\pi}(\Psi')$ is exponentially small. Clearly $\Psi \subset \Psi'$, so this implies that the clustering property is exponentially unlikely to occur.

For each $\sigma \in \Psi'$, construct a c -bridge system for σ . As in the proof of Theorem 1, we complement all bridged components and all non-bridged holes within those components. We obtain $f_1(\sigma)$, which has $t_\sigma \leq cn^2$ tiles, and a bank of $a_\sigma \geq (b - c)n^2$ tiles. Next we define $N(\sigma)$ to be the set of all configurations obtained from $f_1(\sigma)$ by adding a_σ tiles back at any empty location; then $|N(\sigma)| = \binom{n^2 - t_\sigma}{a_\sigma}$. For each $\tau \in \widehat{\Omega}$, we need to bound the number of configurations σ such that $\tau \in N(\sigma)$. As before, we can reconstruct the bridge system for σ with $5^{(1 + \frac{1-c}{2c})\kappa + n}$ information and we can recover the original with 2^{bn^2} information by recording whether each tile moved. Hence the number of σ that map to τ is at most $5^{(1 + \frac{1-c}{2c})\alpha n^2 + n} 2^{bn^2} \leq (2^b \delta)^{n^2/2}$ for large enough n .

Finally, we define a weighted bipartite graph $G(\Psi', \widehat{\Omega}, E)$ with an edge of weight $\pi(\sigma)$ between $\sigma \in \Psi'$ and $\tau \in \widehat{\Omega}$ if $\tau \in N(\sigma)$. The total weight of edges is

$$\sum_{\sigma \in \Psi'} \pi(\sigma) |N(\sigma)| \geq \sum_{\sigma \in \Psi'} \pi(\sigma) \binom{n^2 - (bn^2 - a_\sigma)}{a_\sigma} \geq \pi(\Psi') \delta^{-n^2}.$$

However, the weight of the edges is at most $\sum_{\tau \in \widehat{\Omega}} \pi(\tau) \mu^{-4(b-c)n^2} (2^b \delta)^{n^2/2}$. Let $\mu^* = (2^b/\delta)^{1/(8(b-c))}$ and $\lambda^* = (\mu^*)^{-2} - 1$. Thus for all $\mu < \mu^*$,

$$\pi(\Psi') < \mu^{-4(b-c)n^2} (2^b \delta)^{n^2/2} \delta^{-n^2} < \gamma_2^n,$$

for some $\gamma_2 < 1$, completing the proof. □

5 Other Models

We conclude by considering other natural models of binary mixtures and showing that Theorems 1 and 2 still hold for the interfering models.

5.1 Interfering Binary Mixtures

Model 2: A -tiles are squares on L_n and B -tiles are unit squares centered on vertices of L_n , (see Fig. 5(a)). It is not hard to see that this model corresponds exactly to an independent set model on the rotated grid where vertices correspond to the centers of A -tiles and B -tiles, and the number of even vertices is fixed. The number of odd vertices varies according to λ . Again the A -tiles will cluster together at high enough λ , leaving large regions to fill with B -tiles.

The weight of a configuration σ is proportional to λ^v , where v is the number of vertices in σ not intersecting any A -tiles (we call these *open vertices*). Hence we must argue that by removing several components and putting them together into a single large component, the number of open vertices increases. Indeed, the number of open vertices is proportional to the length of the perimeter, so this can be carried out. One must be careful, however, to define a component so that two tiles are adjacent if they share a vertex (not an edge). Otherwise, if a region looks like a checkerboard of tiles to empty space, and we remove every other row to create a new component, we decrease the perimeter but increase the number of occupied vertices. This cannot happen as long as we choose maximal connected subsets of tiles according to this definition of adjacency.

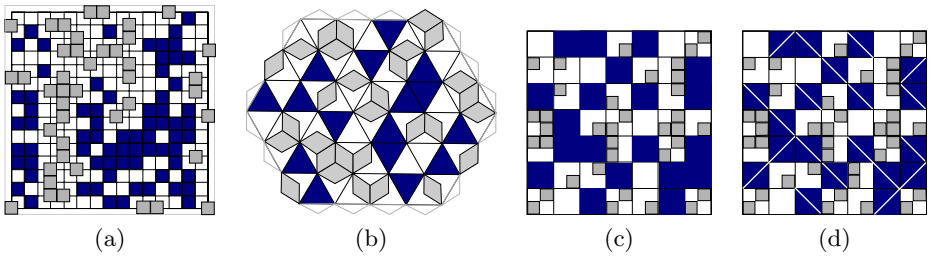


Fig. 5. (a) Model 2 (b) Model 3 (c) Model 4 (d) Model 4 and bond percolation

Model 3: A -tiles are triangles on the triangular lattice Λ_A and B -tiles are lozenges bisected by edges of Λ_A , (see Fig. 5(b)). **Model 3** maps bijectively onto an Ising Model with fixed magnetization on Λ_A . In models like this, where the A -tiles are not square, the large component we create for Theorem 1 might not be square, but some other shape with large area to perimeter ratio, such as a hexagon in this context. The remaining details are similar.

5.2 Noninterfering Binary Mixtures

Model 4: A -tiles are unit squares on L_n and B -tiles are squares of side length $1/2$ on the half-integer lattice, (see Fig. 5(c)). This model is qualitatively different from the previous models since *the placement* of the A -tiles does not influence *the number of places* in which we can put the B -tiles. In fact, this model is just bond percolation on a rotated grid with a fixed number of edges, where we do not expect clustering at any density. To see the bijection, label a unit square with a Northwest-Southeast diagonal if it lies on an even face and label it with a Northeast-Southwest diagonal otherwise, as in Fig. 5(d). Notice that these lines form a subset of the edges of a rotated grid. If we have bn^2 A -tiles then each edge in the rotated grid is present with probability b . To illustrate the difference between the behavior of **Model 4** and the interfering binary mixtures, consider an $n^{1/3} \times n^{1/3}$ window in each. In **Model 4**, the

probability that any $n^{1/3} \times n^{1/3}$ box has density d such that $d > 1.5b$ or $d < 0.5b$ is less than γ_3^n for some constant $\gamma_3 < 1$ (see the full version for details). In contrast by Corollary [11](#), a configuration with the clustering property has a window with density $d \geq .99(1-c)$ and a window with density $d \leq 2.1c$. Hence we see markedly different behavior between interfering and non-interfering binary mixtures.

References

1. Birdi, K.S.: Handbook of Surface and Colloid Chemistry. CRC Press, Boca Raton (2008)
2. Borgs, C., Chayes, J.T., Frieze, A., Kim, J.H., Tetali, P., Vigoda, E., Vu, V.H.: Torpid Mixing of Some MCMC Algorithms in Statistical Physics. In: 40th IEEE Symp. on Foundations of Computer Science (FOCS), pp. 218–229 (1999)
3. Buhot, A., Krauth, W.: Phase Separation in Two-dimensional Additive Mixtures. Phys. Rev. E 59, 2939–2941 (1999)
4. Dobrushin, R.L.: The Problem of Uniqueness of a Gibbsian Random Field and the Problem of Phase Transitions. Funct. Analysis and its App. 2, 302–312 (1968)
5. Dobrushin, R.L., Kotecký, R., Shlosman, S.B.: The Wulff Construction: A Global Shape from Local Interactions. American Mathematical Society, Providence (1992)
6. Dobrushin, R.L., Shlosman, S.: Constructive Criterion for the Uniqueness of Gibbs Fields. In: Statistical Physics and Dynamical Systems, pp. 347–370, Birkhäuser, Boston (1985)
7. Dress, C., Krath, W.: Cluster Algorithm for Hard Spheres and Related Systems. J. Phys. A: Math. Gen. 28, L597 (1995)
8. Frenkel, D.: Order Through Disorder: Entropy-driven Phase Transitions. Lecture Notes in Physics, vol. 415, pp. 137–148 (1993)
9. Frenkel, D., Louis, A.: Phase Separation in Binary Hard-core Mixtures: An Exact Result. Physical Review Letters 68, 3363–3365 (1992)
10. Gore, V., Jerrum, M.: The Swendsen-Wang Algorithm Does Not Always Mix Rapidly. J. Statistical Phys. 97, 67–86 (1999)
11. Hiemenz, P., Rajagopalan, R.: Principles of Colloid and Surface Chemistry. CRC Press, Boca Raton (1997)
12. Krauth, W.: Cluster Monte Carlo algorithms. In: Hartmann, A.K., Rieger, H. (eds.) New Optimization Algorithms in Physics, Wiley-VCh, Weinheim (2005)
13. Martinelli, F.: Lectures on Glauber dynamics for Discrete Spin Models. Lectures Notes in Math. vol. 1717, pp. 93–191 (1999)
14. Miracle, S., Randall, D., Streib, A.P.: Cluster Algorithms for Discrete Models of Colloids with Bars. In: 8th Workshop on Analytic Algorithmics and Combinatorics (ANALCO), pp. 135–150 (2011)
15. Randall, D.: Slow Mixing of Glauber Dynamics Via Topological Obstructions. In: 17th ACM-SIAM Symp. on Discrete Alg. (SODA), pp. 870–879 (2006)
16. Rodríguez-Guadarrama, L., Talsania, S., Mohanty, K., Rajagopalan, R.: Mixing Properties of Two-Dimensional Lattice Solutions of Amphiphiles. J. of Colloid and Interface Science 224, 188–197 (2001)

Approximating the Influence of Monotone Boolean Functions in $O(\sqrt{n})$ Query Complexity

Dana Ron*, Ronitt Rubinfeld**, Muli Safra***, and Omri Weinstein†

Abstract. The *Total Influence (Average Sensitivity)* of a discrete function is one of its fundamental measures. We study the problem of approximating the total influence of a monotone Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, which we denote by $I[f]$. We present a randomized algorithm that approximates the influence of such functions to within a multiplicative factor of $(1 \pm \epsilon)$ by performing $O\left(\frac{\sqrt{n} \log n}{I[f]} \text{poly}(1/\epsilon)\right)$ queries. We also prove a lower bound of $\Omega\left(\frac{\sqrt{n}}{\log n \cdot I[f]}\right)$ on the query complexity of any constant-factor approximation algorithm for this problem (which holds for $I[f] = \Omega(1)$), hence showing that our algorithm is almost optimal in terms of its dependence on n . For general functions we give a lower bound of $\Omega\left(\frac{n}{I[f]}\right)$, which matches the complexity of a simple sampling algorithm.

1 Introduction

The influence of a function, first introduced by Ben-Or and Linial [2] in the context of “collective coin-flipping”, captures the notion of the sensitivity of a multivariate function. More precisely, for a Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, the *individual influence* of coordinate i on f is defined as $I_i[f] \stackrel{\text{def}}{=} \Pr_{x \in \{0, 1\}^n} [f(x) \neq f(x^{(\oplus i)})]$, where x is selected uniformly in $\{0, 1\}^n$ and $x^{(\oplus i)}$ denotes x with the i^{th} bit flipped. The *total influence* of a Boolean function f (which we simply refer to as *the influence* of f) is $I[f] = \sum_i I_i[f]$.

The study of the influence of a function and its individual influences (distribution) has been the focus of many papers – for a survey see [12]. The influence of functions has played a central role in several areas of computer science. In particular, this is true for distributed computing (e.g., [2, 17]), hardness of approximation (e.g., [9, 18]), learning

* School of Electrical Engineering at Tel Aviv University, danar@eng.tau.ac.il. This work was supported by the Israel Science Foundation (grant number 246/08).

** CSAIL at MIT, and the Blavatnik School of Computer Science at Tel Aviv University, ronitt@csail.mit.edu. This work was supported by NSF grants 0732334 and 0728645, Marie Curie Reintegration grant PIRG03-GA-2008-231077 and the Israel Science Foundation grant nos. 1147/09 and 1675/09.

*** Blavatnik School of Computer Science at Tel Aviv University, safra@post.tau.ac.il

† Computer Science Department, Princeton University, oweinste@cs.princeton.edu

¹ The influence can be defined with respect to other probability spaces (as well as for non-Boolean functions), but we focus on the above definition.

theory (e.g., [13,6,24,8])² and property testing (e.g., [10,4,5,22,26]). The notion of influence also arises naturally in the context of probability theory (e.g., [27,28,3]), game theory (e.g., [20]), reliability theory (e.g., [19]), as well as theoretical economics and political science (e.g., [11,5,16]).

Given that the influence is such a basic measure of functions and it plays an important role in many areas, we believe it is of interest to study the algorithmic question of approximating the influence of a function as efficiently as possible, that is by querying the function on as few inputs as possible. Specifically, the need for an efficient approximation for a function’s influence might arise in the design of sublinear algorithms, and in particular property testing algorithms.

As we show, one cannot improve on a standard sampling argument for the problem of estimating the influence of a general Boolean function, which requires $\Omega(\frac{n}{I[f]})$ queries to the function, for any constant multiplicative estimation factor.³ This fact justifies the study of subclasses of Boolean functions, among which the family of monotone functions is a very natural and central one. Indeed, we show that the special structure of monotone functions implies a useful behavior of their influence, making the computational problem of approximating the influence of such functions significantly easier.

Our Results and Techniques. We present a randomized algorithm that approximates the influence of a monotone Boolean function to within any multiplicative factor of $(1 \pm \epsilon)$ in $O\left(\frac{\sqrt{n} \log n}{I[f]} \text{poly}(1/\epsilon)\right)$ expected query complexity. We also prove a nearly matching lower bound of $\Omega\left(\frac{\sqrt{n}}{\log n \cdot I[f]}\right)$ on the query complexity of any constant-factor approximation algorithm for this problem (which holds for $I[f] = \Omega(1)$).

As noted above, the influence of a function can be approximated by sampling random edges (i.e., pairs $(x, x^{(\oplus i)})$ that differ on a single coordinate) from the $\{0, 1\}^n$ lattice. A random edge has probability $\frac{I[f]}{n}$ to be influential (i.e. satisfy $f(x) \neq f(x^{(\oplus i)})$), so a standard sampling argument implies that it suffices to ask $O(\frac{n}{I[f]} \text{poly}(1/\epsilon))$ queries in order to approximate this probability to within $(1 \pm \epsilon)$.⁴

In order to achieve better query complexity, we would like to increase the probability of hitting an influential edge in a single trial. The algorithm we present captures this intuition, by taking random walks down the $\{0, 1\}^n$ lattice, and then averaging the total number of influential edges encountered in all walks over the number of walks taken. The crucial observation on which the algorithm relies, is that a monotone function can have at most one influential edge in a single path, and thus it is sufficient to query

² Here we referenced several works in which the influence appears explicitly. The influence of variables plays an implicit role in many learning algorithms, and in particular those that build on Fourier analysis, beginning with [21].

³ If one wants an additive error of ϵ , then $\Omega((n/\epsilon)^2)$ queries are necessary (when the influence is large) [23].

⁴ We also note that in the case of monotone functions, the total influence equals twice the sum of the Fourier coefficients that correspond to singleton sets $\{i\}$, $i \in \{1, \dots, n\}$. Therefore, it is possible to approximate the influence of a function by approximating this sum, which equals $\frac{1}{2^n} \cdot \sum_{i=1}^n \left(\sum_{x \in \{0,1\}^n: x_i=1} f(x) - \sum_{x \in \{0,1\}^n: x_i=0} f(x)\right)$. However, the direct sampling approach for such an approximation again requires $\Omega(n/I[f])$ samples.

only the start and end points of the walk to determine whether any influential edge was traversed.

Before continuing the technical discussion concerning the algorithm and its analysis, we make the following more conceptual note. Random walks have numerous applications in Computer Science as they are an important tool for mixing and sampling almost uniformly. In our context, where the walk is performed on the domain of an unknown function, it is used for a different purpose. Namely, by querying only the two endpoints of a random walk (starting from a uniformly sampled element) we (roughly) simulate the process of taking a much larger sample of elements⁵

The main issue that remains is determining the length of the walk, which we denote by w . Let $p_w(f)$ denote the probability that a walk of length w (down the lattice and from a uniformly selected starting point) passes through some influential edge⁶. We are interested in analyzing how $p_w(f)$ increases as a function of w . We show that for w that is $O(\epsilon\sqrt{n/\log n})$, the value of $p_w(f)$ increases almost linearly with w . Namely, it is $(1 \pm \epsilon) \cdot \frac{w}{n} \cdot I[f]$. Thus, by taking w to be $\Theta(\epsilon\sqrt{n/\log n})$ we get an improvement by a factor of roughly \sqrt{n} on the basic sampling algorithm. We note though that by taking w to be larger we cannot ensure in general the same behavior of $p_w(f)$ as a function of w and $I[f]$, since the behavior might vary significantly depending on f .

The way we prove the aforementioned dependence of $p_w(f)$ on w is roughly as follows. For any edge e in the Boolean lattice, let $p_w(e)$ denote the probability that a walk of length w (as defined above) passes through e . By the observation made previously, that a monotone function can have at most one influential edge in a given path, $p_w(f)$ is the sum of $p_w(e)$, taken over all edges e that are influential with respect to f . For our purposes it is important that $p_w(e)$ be roughly the same for almost all edges. Otherwise, different functions that have the same number of influential edges, and hence the same influence $I[f]$, but whose influential edges are distributed differently in the Boolean lattice, would give different values for $p_w(f)$. We show that for $w = O(\epsilon\sqrt{n/\log n})$, the value of $p_w(e)$ increases almost linearly with w for all but a negligible fraction of the influential edges (where ‘negligible’ is with respect to $I[f]$). This implies that $p_w(f)$ grows roughly linearly in w for $w = O(\epsilon\sqrt{n/\log n})$.

To demonstrate the benefit of taking walks of length $O(\sqrt{n})$, let us consider the classic example of the *Majority* function on n variables. Here, all influential edges are concentrated in the exact middle levels of the lattice (i.e, all of them are of the form $(x, x^{(\oplus i)})$ where the Hamming weight of x is $\lfloor n/2 \rfloor$ or $\lceil n/2 \rceil$). The probability, $p_w(e)$, of a walk of length w passing through an influential edge e is simply the probability of starting the walk at distance at most w above the threshold $n/2$. Thus, taking longer walks allows us, so to speak, to start our walk from a higher point in the lattice, and still hit an influential edge. Since the probability of a uniformly chosen point to fall in each one of the the first \sqrt{n} levels above the middle is roughly the same, the probability of hitting an influential edge in that case indeed grows roughly linearly in the size

⁵ We also note that the relation between edges and certain paths that pass through them arises in the context of Markov Chains when using the *canonical paths* method (see e.g. [14, Chap. 5]).

⁶ For technical reasons we actually consider a slightly different measure than $p_w(f)$, but we ignore this technicality in the introduction.

of the walk. Nevertheless, taking walks of length which significantly exceeds $O(\sqrt{n})$ (say, even $\Omega(\sqrt{n \cdot \log(n)})$) would add negligible contribution to that probability (as this contribution is equivalent to the probability of a uniformly chosen point to deviate $\Omega(\sqrt{n \cdot \log(n)})$ levels from the middle level) and thus the linear dependence on the length of the walk is no longer preserved.

2 Preliminaries

In the introduction we defined the influence of a function as the sum, over all its variables, of their individual influence. An equivalent definition is that the influence of a function f is the expected number of sensitive coordinates for a random input $x \in \{0, 1\}^n$ (that is, those coordinates i for which $f(x) \neq f(x^{\oplus i})$).

It will occasionally be convenient to view f as a 2-coloring of the Boolean lattice. Under this setting, any “bi-chromatic” edge, i.e. an edge $(x, x^{\oplus i})$ such that $f(x) \neq f(x^{\oplus i})$, will be called an *influential edge*. The number of influential edges of a Boolean function f is $2^{n-1} \cdot I[f]$.

We consider the standard partial order ‘ \prec ’ over the (n -dimensional) Boolean lattice. Namely, for $x = (x_1, \dots, x_n), y = (y_1, \dots, y_n) \in \{0, 1\}^n$, we use the notation $x \prec y$ to mean that $x_i \leq y_i$ for every $1 \leq i \leq n$, and $x_i < y_i$ for some $1 \leq i \leq n$. A Boolean function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ is said to be *monotone* if $f(x) \leq f(y)$ for all $x \prec y$. A well known isoperimetric inequality implies that any monotone Boolean function satisfies $I[f] = O(\sqrt{n})$ (see [11] for a proof). This bound is tight for the notable *Majority* function.

In this paper we deal mainly with monotone Boolean functions that have at least constant Influence (i.e. $I[f] \geq c$, for some $c \geq 0$), since the computational problem we study arises more naturally when the function has some significant sensitivity. As shown in [17], the influence of a function is lower bounded by $4 \cdot \Pr[f(x) = 1] \cdot \Pr[f(x) = 0]$, and so our analysis holds in particular for functions that are not too biased (relatively balanced, i.e., $\Pr[f(x) = 1]$ and $\Pr[f(x) = 0]$ do not differ by much).

Notations. We use the notation $f(n) = \tilde{O}(g(n))$ if $f(n) = O(g(n)\text{polylog}(g(n)))$. Similarly, $f(n) = \tilde{\Omega}(g(n))$ if $f(n) = \Omega(g(n)/\text{polylog}(g(n)))$.

3 The Algorithm

As noted in the introduction, we can easily get a $(1 \pm \epsilon)$ -factor estimate of the influence with high constant probability by uniformly sampling $\Theta\left(\frac{n}{I[f]} \cdot \epsilon^{-2}\right)$ pairs $(x, x^{\oplus i})$ (edges in the Boolean lattice), querying the function on these pairs, and considering the fraction of influential edges observed in the sample. We refer to this as the *direct sampling approach*. However, since we are interested in an algorithm whose complexity

⁷ To verify this, observe that when partitioning the Boolean lattice into two sets with respect to a coordinate i , we end up with 2^{n-1} vertices in each set. The individual influence of variable i , $I_i[f]$, is the fraction of the “bi-chromatic” edges among all edges crossing the cut. Since $I[f] = \sum_{i=1}^n I_i[f]$ we get that the total number of influential edges is $2^{n-1} \cdot I[f]$.

Algorithm 1: Approximating the Influence (given ϵ, δ and oracle access to f)

1. Set $\tilde{\epsilon} = \epsilon/4$, $w = \frac{\tilde{\epsilon}\sqrt{n}}{16\sqrt{2\log(\frac{2n}{\tilde{\epsilon}})}}$, $s^* = \frac{1}{2}\sqrt{2n\log(\frac{2n}{\tilde{\epsilon}})}$, and $t = \frac{96\ln(\frac{2}{\delta})}{\epsilon^2}$.
2. Initialize $\alpha \leftarrow 0$, $m \leftarrow 0$, and $\hat{I} \leftarrow 0$.
3. Repeat the following until $\alpha = t$:
 - (a) Perform a random walk of length w down the $\{0, 1\}^n$ lattice from a uniformly chosen point v with a cut-off at $n/2 - s^* - 1$, and let u denote the endpoint of the walk.
 - (b) If $f(u) \neq f(v)$ then $\alpha \leftarrow \alpha + 1$.
 - (c) $m \leftarrow m + 1$
4. $\hat{I} \leftarrow \frac{n}{w} \cdot \frac{t}{m}$
5. Return \hat{I} .

Fig. 1. The algorithm for approximating the influence of a monotone function f

is $\frac{\sqrt{n}}{I[f]} \cdot \text{poly}(1/\epsilon)$ we take a different approach. To be precise, the algorithm we describe works for ϵ that is above a certain threshold (of the order of $\sqrt{\log n/n}$). However, if ϵ is smaller, then $\frac{n}{I[f]} \cdot \epsilon^{-2}$ is upper bounded by $\frac{\sqrt{n}}{I[f]} \cdot \text{poly}(1/\epsilon)$, and we can take the direct sampling approach. Thus we assume from this point on that $\epsilon \geq c\sqrt{\log n/n}$, for some sufficiently large constant c .

As discussed in the introduction, instead of considering neighboring pairs, $(x, x^{(\oplus i)})$, we consider pairs (v, u) such that $v \succ u$ and there is a path down the lattice of length roughly $\epsilon\sqrt{n}$ between v and u . Observe that since the function f is monotone, if the path (down the lattice) from v to u contains an influential edge, then $f(v) \neq f(u)$, and furthermore, any such path can contain at most one influential edge. The intuition is that since we “can’t afford” to detect influential edges directly, we raise our probability of detecting edges by considering longer paths.

In our analysis we show that this intuition can be formalized so as to establish the correctness of the algorithm. We stress that when considering a path, the algorithm only queries its endpoints, so that it “doesn’t pay” for the length of the path. The precise details of the algorithm are given in Figure 1. When we say that we take a walk of a certain length w down the Boolean lattice *with a cut-off* at a certain level ℓ , we mean that we stop the walk (before taking all w steps) if we reach a point in level ℓ (i.e., with Hamming weight ℓ).

Note that m , the number of walks taken, is a random variable. Namely, the algorithm continues taking new walks until the number of “successful” walks (that is, walks that pass through an influential edge) reaches a certain threshold, which is denoted by t . The reason for doing this, rather than deterministically setting the number of walks and considering the random variable which is the number of successful walks, is that the latter approach requires to know a lower bound on the influence of f . While it is possible to search for such a lower bound (by working iteratively in phases and decreasing the lower bound on the influence between phases) our approach yields a somewhat simpler algorithm.

In what follows we assume for simplicity that $I[f] \geq 1$. As we discuss subsequently, this assumption can be easily replaced with $I[f] \geq c$ for any constant $c > 0$, or even $I[f] \geq n^{-c}$, by slightly modifying the setting of the parameters of the algorithm.

Theorem 1. For every monotone function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ such that $I[f] \geq 1$, and for every $\delta > 0$ and $\epsilon = \omega(\sqrt{\log n/n})$, with probability at least $1 - \delta$, the output, \hat{I} , of Algorithm [1](#) satisfies: $(1 - \epsilon) \cdot I[f] \leq \hat{I} \leq (1 + \epsilon) \cdot I[f]$. Furthermore, with probability at least $1 - \delta$, the number of queries performed by the algorithm is $O\left(\frac{\log(1/\delta)}{\epsilon^3} \cdot \frac{\sqrt{n} \log(n/\epsilon)}{I[f]}\right)$.

We note that the bound on the number of queries performed by the algorithm implies that the expected query complexity of the algorithm is $O\left(\frac{\log(1/\delta)}{\epsilon^3} \cdot \frac{\sqrt{n} \log(n/\epsilon)}{I[f]}\right)$.

Furthermore, the probability that the algorithm performs a number of queries that is more than k times the expected value decreases exponentially with k .

The next definition is central to our analysis.

Definition 1. For a (monotone) Boolean function f and integers w and s^* , let $p_{w,s^*}(f)$ denote the probability that a random walk of length w down the Boolean lattice, from a uniformly selected point and with a cut-off at $n/2 - s^* - 1$, starts from $f(v) = 1$ and reaches $f(u) = 0$.

Given the definition of $p_{w,s^*}(f)$, we next state and prove the main lemma on which the proof of Theorem [1](#) is based.

Lemma 1. Let f satisfy $I[f] \geq 1$, let $\epsilon > 0$ satisfy $\epsilon > \frac{8\sqrt{2 \log(\frac{8n}{\epsilon})}}{\sqrt{n}}$, and denote $\tilde{\epsilon} = \epsilon/4$. For any $w \leq \frac{\tilde{\epsilon}\sqrt{n}}{16\sqrt{2 \log(\frac{2n}{\tilde{\epsilon}I[f]})}}$ and for $s^* = \frac{1}{2}\sqrt{n} \cdot \sqrt{2 \log(\frac{2n}{\tilde{\epsilon}})}$ we have that

$$(1 - \epsilon/2) \cdot \frac{w}{n} \cdot I[f] \leq p_{w,s^*}(f) \leq (1 + \epsilon/2) \cdot \frac{w}{n} \cdot I[f].$$

Proof: For a point $y \in \{0, 1\}^n$, let $h(y)$ denote its Hamming weight (which we also refer to as the *level* in the Boolean lattice that it belongs to). By the choice of $s^* = \frac{1}{2}\sqrt{n}\sqrt{2 \log(\frac{2n}{\tilde{\epsilon}})}$, and since $I[f] \geq 1$, the number of points y for which $h(y) \geq n/2 + s^*$ or $h(y) \leq n/2 - s^*$, is upper bounded by $2^n \cdot \frac{\tilde{\epsilon}I[f]}{n}$. It follows that there are at most $2^{n-1} \cdot \tilde{\epsilon}I[f]$ edges (y, x) for which $h(y) \geq n/2 + s^*$ or $h(y) \leq n/2 - s^*$. Recall that an influential edge (y, x) for $h(y) = h(x) + 1$, is an edge that satisfies $f(y) = 1$ and $f(x) = 0$. Let $e_{s^*}(f)$ denote the number of influential edges (y, x) such that $n/2 - s^* \leq h(x), h(y) \leq n/2 + s^*$. Since the total number of influential edges is $2^{n-1}I[f]$, we have that

$$(1 - \tilde{\epsilon}) \cdot 2^{n-1}I[f] \leq e_{s^*}(f) \leq 2^{n-1}I[f]. \tag{1}$$

Consider any influential edge (y, x) where $h(y) = \ell$ and $\ell \geq n/2 - s^*$. We are interested in obtaining bounds on the probability that a random walk of length w (where $w \leq \frac{\tilde{\epsilon}\sqrt{n}}{16\sqrt{2 \log(\frac{2n}{\tilde{\epsilon}I[f]})}}$) down the lattice, starting from a uniformly selected point $v \in \{0, 1\}^n$, and with a cut-off at $n/2 - s^* - 1$, passes through (y, x) . First, there is the event that $v = y$ and the edge (y, x) was selected in the first step of the walk. This event occurs with probability $2^{-n} \cdot \frac{1}{\ell}$. Next there is the event that v is at distance 1 from y (and

above it, that is, $h(v) = h(y) + 1 = \ell + 1$, and the edges (v, y) and (y, x) are selected. This occurs with probability $2^{-n} \cdot (n - \ell) \cdot \frac{1}{\ell+1} \cdot \frac{1}{\ell}$. In general, for every $1 \leq i \leq w - 1$ we have $(n - \ell) \cdots (n - \ell - i + 1)$ pairs (v, P) where $v \succ y$ and $w(v) = \ell + i$, and where P is a path down the lattice from v to y . The probability of selecting v as the starting vertex is 2^{-n} and the probability of taking the path P from v is $\frac{1}{(\ell+i)\cdots(\ell+1)}$. Therefore, the probability that the random walk passes through (y, x) is:

$$2^{-n} \cdot \frac{1}{\ell} \cdot \left(1 + \sum_{i=1}^{w-1} \frac{(n - \ell) \cdots (n - \ell - i + 1)}{(\ell + i) \cdots (\ell + 1)} \right) = 2^{-n} \cdot \frac{1}{\ell} \left(1 + \sum_{i=1}^{w-1} \prod_{j=0}^{i-1} \frac{n - \ell - j}{\ell + i - j} \right). \tag{2}$$

Let $\ell = n/2 + s$ (where s may be negative), and denote $\tau(\ell, i, j) \stackrel{\text{def}}{=} \frac{n - \ell - j}{\ell + i - j}$. Then

$$\tau(\ell, i, j) = \frac{n/2 - s - j}{n/2 + s + i - j} = 1 - \frac{2s + i}{n/2 + s + i - j}. \tag{3}$$

Consider first the case that $\ell \geq n/2$, i.e $\ell = n/2 + s$ ($s \geq 0$). In that case it is clear that $\tau(\ell, i, j) \leq 1$ (since $j \leq i$), so $\prod_{j=0}^{i-1} \tau(\ell, i, j)$ is upper bounded by 1. In order to lower bound $\prod_{j=0}^{i-1} \tau(\ell, i, j)$, we note that

$$\tau(\ell, i, j) \geq 1 - \frac{2s + w}{n/2} = 1 - \frac{2(2s + w)}{n}. \tag{4}$$

Thus, for $s \leq s^*$ we have

$$\begin{aligned} \prod_{j=0}^{i-1} \tau(\ell, i, j) &\geq \prod_{j=0}^{i-1} \left(1 - \frac{2(2s + w)}{n} \right) \geq \left(1 - \frac{2(2s + w)}{n} \right)^w \geq 1 - \frac{2(2s + w)w}{n} \\ &\geq 1 - \frac{6s^*w}{n} = 1 - \frac{3\tilde{\epsilon}}{16} \geq 1 - \tilde{\epsilon}/2 \end{aligned}$$

where the second inequality uses $i \leq w$, the fourth inequality uses $s \leq s^*$ and $w \leq s^*$, and the fifth inequality uses the definitions of s^* and w . Therefore, we have that for $n/2 \leq \ell \leq n/2 + s^*$,

$$1 - \tilde{\epsilon}/2 \leq \prod_{j=0}^{i-1} \frac{n - \ell - j}{\ell + i - j} \leq 1, \tag{5}$$

and for $\ell > n/2 + s^*$ it holds that

$$\prod_{j=0}^{i-1} \frac{n - \ell - j}{\ell + i - j} \leq 1. \tag{6}$$

We turn to the case where $\ell = n/2 - s$ for $1 \leq s \leq s^*$. Here we have

$$\tau(\ell, i, j) = 1 + \frac{2s - i}{n/2 - s + i - j} \geq 1 - \frac{2w}{n - 2w} \geq 1 - \frac{4w}{n} \tag{7}$$

where the last inequality follows from the fact that $w < n/4$. Thus,

$$\prod_{j=0}^{i-1} \tau(\ell, i, j) \geq \left(1 - \frac{4w}{n}\right)^w \geq 1 - \frac{4w^2}{n} = 1 - \frac{4}{n} \cdot \left(\frac{\tilde{\epsilon}\sqrt{n}}{16\sqrt{2\log(\frac{2n}{\tilde{\epsilon}I[f]})}}\right)^2 > 1 - \tilde{\epsilon}^2/2 > 1 - \tilde{\epsilon}/2. \tag{8}$$

On the other hand,

$$\tau(\ell, i, j) = 1 + \frac{2s - i}{n/2 - s + i - j} \leq 1 + \frac{2s}{n/2 - s} \leq 1 + \frac{8s^*}{n}, \tag{9}$$

where the last inequality holds since $n \geq 2s$. Thus, we have

$$\prod_{j=0}^{i-1} \tau(\ell, i, j) \leq \left(1 + \frac{8s^*}{n}\right)^w \leq 1 + \frac{16s^*w}{n} = 1 + \tilde{\epsilon}/2. \tag{10}$$

where the second inequality follows from the inequality $(1 + \alpha)^k \leq 1 + 2\alpha k$ which holds for $\alpha < 1/(2k)$; Indeed, in our case $8s^*/n \leq 1/(2w)$ (this is equivalent to $w \leq n/16s^*$ which holds given our setting of s^* and the upper bound on w).

We therefore have that for $n/2 - s^* \leq \ell < n/2$,

$$1 - \tilde{\epsilon}/2 \leq \prod_{j=0}^{i-1} \frac{n - \ell - j}{\ell + i - j} \leq 1 + \tilde{\epsilon}/2. \tag{11}$$

Combining Equations (5) and (11), we have that for $n/2 - s^* \leq \ell \leq n/2 + s^*$,

$$1 - \tilde{\epsilon}/2 \leq \prod_{j=0}^{i-1} \frac{n - \ell - j}{\ell + i - j} \leq 1 + \tilde{\epsilon}/2. \tag{12}$$

Now, we are interested in summing up the probability, over all random walks, that the walk passes through an influential edge. Since the function is monotone, every random walk passes through at most one influential edge, so the sets of random walks that correspond to different influential edges are disjoint (that is, the event that a walk passes through an influential edge (y, x) is disjoint from the event that it passes through another influential edge (y', x')). Since the edges that contribute to $p_{w, s^*}(f)$ are all from levels $\ell \geq n/2 - s^*$ (and since there are $2^{n-1}I[f]$ influential edges in total), by Equations (2), (6) and (12) we have

$$p_{w, s^*}(f) \leq 2^{n-1}I[f]2^{-n} \cdot \frac{1}{n/2 - s^*} \left(1 + \sum_{i=1}^{w-1} (1 + \tilde{\epsilon}/2)\right) \tag{13}$$

$$\leq \frac{I[f] \cdot w}{n} (1 + \epsilon/2), \tag{14}$$

For lower bounding $p_{w, s^*}(f)$, we will consider only the contribution of the influential edges that belong to levels $\ell \leq n/2 + s^*$. Consequently, Equations (1), (2) and (12) give in total

$$p_{w, s^*}(f) \geq \frac{I[f] \cdot w}{n} (1 - \epsilon/2), \tag{15}$$

(by applying similar manipulations to those used for deriving Equations (13)–(14)).

Equations (14) and (15) give

$$(1 - \epsilon/2) \cdot \frac{w}{n} \cdot I[f] \leq p_{w,s^*}(f) \leq (1 + \epsilon/2) \cdot \frac{w}{n} \cdot I[f], \tag{16}$$

as claimed in the Lemma. ■

Proof of Theorem 1: For w and s^* as set by the algorithm, let $p_{w,s^*}(f)$ be as in Definition 1 where we shall use the shorthand $p(f)$. Recall that m is a random variable denoting the number of iterations performed by the algorithm until it stops (once $\alpha = t$). Let $\tilde{m} = \frac{t}{p(f)}$, $\tilde{m}_1 = \frac{\tilde{m}}{(1+\epsilon/4)}$, and $\tilde{m}_2 = \frac{\tilde{m}}{(1-\epsilon/4)}$. We say that an iteration of the algorithm is *successful* if the walk taken in that iteration passes through an influential edge (so that the value of α is increased by 1). Let $\hat{p}(f) = \frac{t}{m}$ denote the fraction of successful iterations.

Suppose that $\tilde{m}_1 \leq m \leq \tilde{m}_2$. In such a case,

$$(1 - \epsilon/4) \cdot p(f) \leq \hat{p}(f) \leq (1 + \epsilon/4)p(f) \tag{17}$$

since $\hat{p}(f) = \frac{t}{m} = \frac{p(f) \cdot \tilde{m}}{m}$. By the definition of the algorithm, $\hat{I} = \frac{n}{w} \cdot \frac{t}{m} = \frac{n}{w} \cdot \hat{p}(f)$ so by Lemma 1 (recall that by the premise of the theorem, $\epsilon = \omega(\sqrt{\log n/n})$) we have

$$(1 - \epsilon)I[f] \leq (1 - \epsilon/2)(1 - \epsilon/4)I[f] \leq \hat{I} \leq (1 + \epsilon/4)(1 + \epsilon/2)I[f] \leq (1 + \epsilon)I[f]$$

and thus (assuming $\tilde{m}_1 \leq m \leq \tilde{m}_2$), the output of the algorithm provides the estimation we are looking for.

It remains to prove that $\tilde{m}_1 \leq m \leq \tilde{m}_2$ with probability at least $1 - \delta$. Let X_i denote the indicator random variable whose value is 1 if and only if the i^{th} iteration of the algorithm was successful, and let $X = \sum_{i=1}^{\tilde{m}_1} X_i$. By the definition of X_i , we have that $E[X_i] = p(f)$, and so (by the definition of \tilde{m}_1 and \tilde{m}) we have that $E[X] = \tilde{m}_1 \cdot p(f) = \frac{t}{1+\epsilon/4}$. Hence, by applying the multiplicative Chernoff bound [7],

$$\Pr[m < \tilde{m}_1] \leq \Pr[X \geq t] = \Pr[X \geq (1 + \epsilon/4)E[X]] \leq \exp\left(-\frac{\epsilon^2 t}{100}\right) \tag{18}$$

Thus, for $t = \frac{100 \ln(\frac{2}{\delta})}{\epsilon^2}$ we have that $\Pr[m < \tilde{m}_1] \leq \frac{\delta}{2}$. By an analogous argument we get that $\Pr[m > \tilde{m}_2] \leq \frac{\delta}{2}$, and so $\tilde{m}_1 \leq m \leq \tilde{m}_2$ with probability at least $1 - \delta$, as desired.

Since in particular $m \leq \tilde{m}_2$ with probability at least $1 - \delta$, and the query complexity of the algorithm is $O(m)$, we have that, with probability at least $1 - \delta$, the query complexity is upper bounded by

$$O(\tilde{m}_2) = O\left(\frac{t}{p(f)}\right) = O\left(\frac{t \cdot n}{w \cdot I[f]}\right) = O\left(\frac{\log(1/\delta)}{\epsilon^3} \cdot \frac{\sqrt{n} \log(n/\epsilon)}{I[f]}\right) \tag{19}$$

as required. ■

Remark. We assumed that $I[f] \geq 1$ only for the sake of technical simplicity. This assumption can be replaced with $I[f] \geq \frac{1}{n^c}$ for any constant $c \geq 0$, and the only modifications needed in the algorithm and its analysis are the following. The level of the cutoff s^* should be set to $s^* = \sqrt{n/2} \cdot \sqrt{\log(\frac{2n}{\epsilon n - c})} = \frac{1}{2}\sqrt{n} \sqrt{2c \log(2n) + \log(1/\epsilon)}$ (which is a constant factor larger than the current setting), and the length w of the walks in the algorithm should be set to $w = \frac{\epsilon \sqrt{n}}{16 \sqrt{2 \log(\frac{2n}{\epsilon n - c})}}$ (which is a constant factor smaller than the current setting). For details of the analysis see the full version of this paper [25].

We note that the lower bound we give in Section 4 applies only to functions with (at least) constant influence, and so in the above case where $I[f] = 1/\text{poly}(n)$, the tightness of the algorithm (in terms of query complexity) is not guaranteed.

4 A Lower Bound

In this section we prove a lower bound of $\Omega\left(\frac{\sqrt{n}}{I[f] \cdot \log n}\right)$ on the query complexity of approximating the influence of monotone functions. Following it we explain how a related construction gives a lower bound of $\Omega\left(\frac{n}{I[f]}\right)$ on approximating the influence of *general* functions. The idea for the first lower bound is the following. We show that any algorithm that performs $o\left(\frac{\sqrt{n}}{I[f] \cdot \log n}\right)$ queries cannot distinguish with constant success probability between as follows: (1) A certain threshold function (over a relatively small number of variables), and (2) A function selected uniformly at random from a certain family of functions that have significantly higher influence than the threshold function. The functions in this family can be viewed as “hiding their influence behind the threshold function”. More precise details follow.

We first introduce one more notation. For any integer $1 \leq k \leq n$ and $0 \leq t \leq k$, let $\tau_k^t : \{0, 1\}^n \rightarrow \{0, 1\}$ be the t -threshold function over x_1, \dots, x_k . That is, $\tau_k^t(x) = 1$ if and only if $\sum_{i=1}^k x_i \geq t$. Observe that (since for every $1 \leq i \leq k$ we have that $I_i[\tau_k^t] = 2^{-k} \cdot 2 \cdot \binom{k-1}{t-1}$ while for $i > k$ we have that $I_i[\tau_k^t] = 0$), $I[\tau_k^t] = k \cdot 2^{-(k-1)} \cdot \binom{k-1}{t-1}$.

The above observation implies that for every sufficiently large k ($k \geq 2 \log n$ suffices), there exists a setting of $t < k/2$, which we denote by $t(k, 1)$, such that $I[\tau_k^{t(k, 1)}] = 1 - o(1)$ (where the $o(1)$ is with respect to k). This setting satisfies $\binom{k-1}{t(k, 1)-1} = \Theta(2^k/k)$ (so that $t(k, 1) = k/2 - \Theta(\sqrt{k \log k})$).

Theorem 2. *For every I^* such that $2 \leq I^* \leq \sqrt{n}/\log n$, there exists a family of monotone functions F_{I^*} such that $I[f] \geq I^*$ for every $f \in F_{I^*}$, but any algorithm that distinguishes with probability at least $2/3$ between a uniformly selected function in F_{I^*} and $\tau_k^{t(k, 1)}$ for $k = 2 \log n$, must perform $\Omega\left(\frac{\sqrt{n}}{I^* \cdot \log n}\right)$ queries.*

In particular, considering $I^* = c$ for any constant $c \geq 2$, we get that every algorithm for approximating the influence to within a multiplicative factor of \sqrt{c} must perform $\tilde{\Omega}(\sqrt{n})$ queries. If we increase the lower bound Ω on the influence, then the lower bound on the complexity of the algorithm decreases, but the approximation factor (for which the lower bound holds), increases. We note that the functions for which the lower bound

construction hold are not balanced, but we can easily make them very close to balanced without any substantial change in the argument (by “ORing” $\tau_k^{t(k,1)}$ as well as every function in F_{I^*} with x_1). We also note that for $I^* = \Omega(\sqrt{\log n})$ we can slightly improve the lower bound on approximating the influence to $\Omega\left(\frac{\sqrt{n}}{I^* \cdot \sqrt{\log(\sqrt{n}/I^*)}}\right)$ (for a slightly smaller approximation factor). We address this issue in the full version of this paper [25].

Proof: For $k = 2 \log n$ and for any $0 \leq t \leq k$, let $L_k^t \stackrel{\text{def}}{=} \{x \in \{0, 1\}^k : \sum_{i=1}^k x_i = t\}$. We shall also use the shorthand \tilde{t} for $t(k, 1)$. Fixing a choice of I^* , each function in F_{I^*} is defined by a subset R of $L_k^{\tilde{t}}$ where $|R| = \beta(I^*) \cdot 2^k$ for $\beta(I^*)$ that is set subsequently. We denote the corresponding function by f_R and define it as follows: For every $x \in \{0, 1\}^n$, if $x_1 \dots x_k \notin R$, then $f_R(x) = \tau_k^{\tilde{t}}(x)$, and if $x_1 \dots x_k \in R$, then $f_R(x) = \text{maj}'_{n-k}(x)$, where $\text{maj}'_{n-k}(x) = 1$ if and only if $\sum_{i=k+1}^n x_i > (n - k)/2$. By this definition, each $f_R \in F_{I^*}$ is a monotone function, and $I[f_R] \geq \beta(I^*) \cdot I[\text{maj}'_{n-k}]$. If we take $\beta(I^*)$ to be $\beta(I^*) = I^*/I[\text{maj}'_{n-k}] = cI^*/\sqrt{n - k}$ (for c that is roughly $\sqrt{\pi/2}$), then in F_{I^*} every function has influence at least I^* . Since $\beta(I^*)$ is upper bounded by $|L_k^{\tilde{t}}|/2^k$, which, (by the definition of $\tilde{t} = t(k, 1)$), is of the order of $1/k = \Theta(1/\log n)$ this construction is applicable to $I^* = O(\sqrt{n}/\log n)$.

Consider an algorithm that needs to distinguish between $\tau_k^{\tilde{t}}$ and a uniformly selected $f_R \in F_{I^*}$. Clearly, as long as the algorithm doesn't perform a query on x such that $x_1 \dots x_k \in R$, the value returned by f_R is the same as that of $\tau_k^{\tilde{t}}$. But since R is selected uniformly in $L_k^{\tilde{t}}$, as long as the algorithm performs less than $\frac{|L_k^{\tilde{t}}|}{c' \cdot \beta(I^*) \cdot 2^k}$ queries (where c' is some sufficiently large constant), with high constant probability (over the choice of R), it won't “hit” a point in R . Since $\frac{|L_k^{\tilde{t}}|}{c' \cdot \beta(I^*) \cdot 2^k} = \Theta\left(\frac{\sqrt{n}}{\log n \cdot I^*}\right)$, the theorem follows. ■

A Lower Bound of $\Omega(n/I[f])$ for General Functions. A similar construction as in the proof of Theorem 2 can be used to establish a lower bound of $\Omega(n/I[f])$ queries for estimating the influence of general functions. The idea is the same, only now the lack of the monotonicity constraint allows us to “hide” the influential part of the function “behind” an arbitrary small set R of assignments to the first k variables. We then compensate for the small probability of hitting R with a function having very high influence (Parity) over the remaining $n - k$ variables. For further details see the full version of this paper [25].

Acknowledgements. We would like to thank several anonymous reviewers for their helpful comments.

References

1. Arrow, K.: A difficulty in the theory of social welfare. *Journal of Political Economics* 58, 328–346 (1950)
2. Ben-Or, M., Linial, N.: Collective coin flipping, robust voting schemes and minima of Banzhaf values. In: *Proceedings of FOCS*, pp. 408–416 (1985)
3. Benjamini, I., Kalai, G., Schramm, O.: Noise sensitivity of boolean functions and applications to percolation. *Inst. Hautes Etudes Sci. Publ. Math.* 90, 5–43 (1999)

4. Blais, E.: Improved bounds for testing juntas. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 317–330. Springer, Heidelberg (2008)
5. Blais, E.: Testing juntas nearly optimally. In: Proceedings of STOC, pp. 151–158 (2009)
6. Bshouty, N., Tamon, C.: On the Fourier spectrum of monotone functions. *Journal of the ACM* 43(4), 747–770 (1996)
7. Chernoff, H.: A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of the Mathematical Statistics* 23, 493–507 (1952)
8. Diakonikolas, I., Harsha, P., Klivans, A., Meka, R., Raghavendra, P., Servedio, R., Tan, L.-Y.: Bounding the average sensitivity and noise sensitivity of polynomial threshold functions. In: Proceedings of STOC, pp. 533–543 (2010)
9. Dinur, I., Safra, S.: The importance of being biased. In: Proceedings of STOC, pp. 33–42 (2002)
10. Fischer, E., Kindler, G., Ron, D., Safra, S., Samorodnitsky, S.: Testing juntas. *Journal of Computer and System Sciences* 68(4), 753–787 (2004)
11. Friedgut, E., Kalai, G.: Every monotone graph property has a sharp threshold. *Proc. Amer. Math. Soc.* 124, 2993–3002 (1996)
12. Kalai, G., Safra, S.: Threshold phenomena and influence. In: Computational Complexity and Statistical Physics, pp. 25–60 (2006)
13. Hancock, T., Mansour, Y.: Learning monotone k - μ DNF formulas on product distributions. In: Proceedings of COLT, pp. 179–183 (1991)
14. Jerrum, M.: Counting, Sampling and Integrating: algorithms and complexity. *Lectures in Mathematics*. ETH Zürich, Birkhäuser, Basel (2003)
15. Kalai, G.: A Fourier-theoretic perspective for the Condorcet Paradox and Arrow’s Theorem. *Advances in Applied Mathematics* 29, 412–426 (2002)
16. Kalai, G.: Social indeterminacy. *Econometrica* 72, 1565–1581 (2004)
17. Kalai, G., Kahn, J., Linial, N.: The influence of variables on Boolean functions. In: Proceedings of FOCS, pp. 68–80 (1988)
18. Khot, S.: On the power of unique 2-prover 1-round games. In: Proceedings of STOC, pp. 767–775 (2002)
19. Krivelevich, M., Sudakov, B., Vu, V.H.: A sharp threshold for network reliability. *Combinatorics, Probability and Computing* 11, 465–474 (2002)
20. Lehrer, E.: An axiomatization of the Banzhaf value. *International Journal of Game Theory* 17, 89–99 (1988)
21. Linial, N., Mansour, Y., Nisan, N.: Constant depth circuits, Fourier transform, and learnability. *Journal of the ACM* 40(3), 607–620 (1993)
22. Matulef, K., O’Donnell, R., Rubinfeld, R., Servedio, R.A.: Testing $\{-1, +1\}$ halfspaces. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX 2009. LNCS, vol. 5687, pp. 646–657. Springer, Heidelberg (2009)
23. Matulef, K., Servedio, R., Wimmer, K.: Personal communication (2009)
24. O’Donnell, R., Servedio, R.: Learning monotone decision trees in polynomial time. *SIAM Journal on Computing* 37(3), 827–844 (2007)
25. Ron, D., Rubinfeld, R., Safra, M., Weinstein, O.: Approximating the influence of a monotone boolean function in $o(\sqrt{n})$ query complexity. Technical report, arXiv:1101.5345 (2011)
26. Ron, D., Tsur, G.: Testing computability by width two oBDDs. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX 2009. LNCS, vol. 5687, pp. 686–699. Springer, Heidelberg (2009)
27. Russo, L.: On the critical percolation probabilities. *Z. Wahrsch. Verw. Geb.* 56, 229–237 (1981)
28. Talagrand, M.: On Russo’s approximate 0 – 1 law. *Annals of Probability* 22, 1576–1587 (1994)

On Approximating the Number of Relevant Variables in a Function

Dana Ron* and Gilad Tsur**

Abstract. In this work we consider the problem of approximating the number of relevant variables in a function given query access to the function. Since obtaining a multiplicative factor approximation is hard in general, we consider several relaxations of the problem. In particular, we consider a relaxation of the property testing variant of the problem and we consider relaxations in which we have a promise that the function belongs to a certain family of functions (e.g., linear functions). In the former relaxation the task is to distinguish between the case that the number of relevant variables is at most k , and the case in which it is far from any function in which the number of relevant variable is more than $(1 + \gamma)k$ for a parameter γ . We give both upper bounds and almost matching lower bounds for the relaxations we study.

1 Introduction

In many scientific endeavors, an important challenge is making sense of huge datasets. In particular, when trying to make sense of functional relationships we would like to know or estimate the number of variables that a function depends upon. This can be useful both as a preliminary process for machine learning and statistical inference and independently, as a measure of the complexity of the relationship in question. For the sake of simplicity, in this extended abstract we focus on Boolean functions over the Boolean hypercube, which is endowed with the uniform distribution. We discuss extensions to other finite domains and ranges (as well as other product distributions) in the full version of this paper [13]

For a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$, we let $r(f)$ denote the number of variables that f depends on, which we shall also refer to as the number of *relevant* variables. A variable x_i is *relevant* to a function f if there exists an assignment to the input variables such that changing the value of just the variable x_i causes the value of f to change. Given query access to f , computing $r(f)$ *exactly* may require a number of queries that is exponential in n (linear in the size of the domain) [9]

* School of EE at Tel Aviv University, danar@eng.tau.ac.il. This work was supported by the Israel Science Foundation (grant number 246/08).

** School of EE at Tel Aviv University, gilad.tsur@gmail.com

¹ Consider for example the family of functions, where each function in the family takes the value 0 on all points in the domain but one. Such a function depends on all n variables, but a uniformly selected function in the family cannot be distinguished (with constant probability) from the all-0 function, which depends on 0 variables.

Thus, we would like to consider relaxed notions of this computational task. One natural relaxation is to compute $r(f)$ approximately. Namely, to output a value \hat{r} such that $r(f)/B \leq \hat{r} \leq B \cdot r(f)$ for some approximation factor B . Unfortunately, this relaxed task may still require an exponential number of queries (see the example in Footnote [1](#)).

A different type of relaxation that has been studied in the past, is the one defined by property testing [\[14,8\]](#). We shall say that f is a k -junta if $r(f) \leq k$. A property testing algorithm is given k and a distance parameter $0 < \epsilon < 1$. By performing queries to f , the algorithm should distinguish between the case that f is a k -junta and the case that it differs from every k -junta on at least an ϵ -fraction of the domain (in which case we shall say that it is ϵ -far from being a k -junta). This problem was studied in several papers [\[7,5,3,4\]](#). The best upper bound on the number of queries that the algorithm performs (in terms of the dependence on k) is $O(k \log k)$ [\[4\]](#), where this upper bound almost matches the lower bound of $\Omega(k)$ [\[5\]](#).

A natural question, which was raised in [\[7\]](#), is whether it is possible to reduce the complexity below $\tilde{O}(k)$ if we combine the above two relaxations. Namely, we consider the following problem: *Given parameters $k \geq 1$ and $0 < \epsilon, \gamma < 1$ and query access to a function f , distinguish (with high constant probability) between the case that f is a k -junta and the case that f is ϵ -far from any $(1 + \gamma)k$ -junta.*[2](#) This problem was recently considered by Blais et al. [\[2\]](#). They apply a general new technique that they develop for obtaining lower bounds on property testing problems via communication complexity lower bounds. Specifically, they give a lower bound of $\Omega(\min\{(\frac{k}{t})^2, k\} - \log k)$ on the number of queries necessary for distinguishing between functions that are k -juntas and functions that are ϵ -far from $(k + t)$ -juntas (for a constant ϵ). Using our formulation, this implies that we cannot go below a linear dependence on k for $\gamma = O(1/\sqrt{k})$.

OUR RESULTS. What if we allow γ to be a constant (i.e., independent of k), say, $\gamma = 1$? Our first main result is that even if we allow γ to be a constant, then the testing problem does not become much easier. Specifically, we prove:

Theorem 1. *Any algorithm that distinguishes between the case that f is a k -junta and the case that f is ϵ -far from any $(1 + \gamma)k$ -junta for constant ϵ and γ must perform $\Omega(k/\log(k))$ queries.*

While Theorem [1](#) does not leave much place for improvement of the query complexity as compared to the $O(k \log k)$ upper bound [\[4\]](#) for the standard property testing problem (i.e., when $\gamma = 0$), we show that a small improvement (in terms of the dependence on k) can be obtained:

Theorem 2. *There exists an algorithm that, given query access to $f : \{0, 1\}^n \rightarrow \{0, 1\}$ and parameters $k \geq 1$, and $0 < \epsilon, \gamma < 1$, distinguishes with high constant*

² We note that problems in the spirit of this problem (which allow a further relaxation to that defined by “standard” property testing) have been studied in the past (e.g., [\[11,10,11\]](#)).

probability between the case that f is a k -junta and the case that f is ϵ -far from any $(1 + \gamma)k$ -junta. The algorithm performs $O\left(\frac{k \log(1/\gamma)}{\epsilon \gamma^2}\right)$ queries.

Given that the relaxed property testing problem is not much easier than the standard one in general, we consider another possible relaxation: Computing (approximately) the number of relevant variables of *restricted* classes of functions. For example, suppose that we are given the promise that f is a *linear* function. Since it is possible to exactly learn f (with high constant probability) by performing $O(r(f) \log n)$ queries, it is also possible to exactly compute $r(f)$ in this case using this number of queries. On the other hand, Blais et al. [2] show that in order to distinguish (with constant success probability) between the case that a linear function has k relevant variables and the case that it has more than $k + 1$ relevant variables, requires $\Omega(\min\{k, n - k\})$ queries [3] (so that $\Omega(r(f))$ queries are necessary for exactly computing $r(f)$). However, if we allow a constant multiplicative gap, then we get the following result:

Theorem 3. *Given query access to a linear function f , it is possible to distinguish with high constant probability between the case that f has at most k relevant variables and the case that f has more than $(1 + \gamma)k$ relevant variables by performing $\Theta(\log(1/\gamma)/\gamma^2)$ queries.*

By standard techniques, Theorem 3 implies that we can obtain (with high constant probability) a multiplicative approximation of $(1 + \gamma)$ for $r(f)$ (when f is a linear function), by performing $\tilde{O}(\log(r(f))/\gamma^2)$ queries to f .

Theorem 3, which deals with linear functions, extends to polynomials:

Theorem 4. *There exists an algorithm that distinguishes between polynomials of degree at most d with at most k relevant variables and polynomials of degree at most d that have at least $(1 + \gamma)k$ relevant variables by performing $O\left(\frac{2^d \log(1/\gamma)}{\gamma^2}\right)$ queries.*

Compared to Theorem 2, Theorem 4 gives a better result for degree- d polynomials when $d < \log(k)$. A natural question is whether in this case we can do even better in terms of the dependence on d . We show that it is not possible to do much better (even if we also allow the property testing relaxation):

Theorem 5. *For fixed values of ϵ (for sufficiently small ϵ), and for $d < \log(k)$, any algorithm that distinguishes between polynomials of degree d with k relevant variables and those that are ϵ -far from all degree- d polynomials with $2k$ relevant variables must perform $\Omega(2^d/d)$ queries.*

Finally we show that a lower lower bound similar to the one stated in Theorem 1 holds when we have a promise that the function is monotone (except that it holds for $\epsilon = O(1/\log(k))$ rather than constant ϵ).

³ A slightly weaker bound of $\Omega(k/\text{polylog}(k))$ was proved independently by Chakraborty et al. [6] based on work by Goldreich [9].

TECHNIQUES. Our lower bounds build on reductions from the *Distinct Elements* problem: Given query access to a sequence of length n , the goal is approximate the number of *distinct* elements in the sequence. This problem is equivalent to approximating the support size of a distribution where every element in the support of the distribution has probability that is a multiple of $1/n$ [12]. Several works [12,15,17] gave close to linear lower bounds for distinguishing between support size at least n/d_1 and support size at most n/d_2 (for constant d_1 and d_2), where the best lower bound, due to Valiant and Valiant [17], is $\Omega(n/\log(n))$, and this bound is tight [17].

Turning to the upper bounds, assume first that we have a promise that the function f is a linear function, and we want to distinguish between the case that it depends on at most k variables and the case that it depends on more than $2k$ variables. Suppose we select a subset S of the variables by including each variable in the subset, independently, with probability $1/2k$. The first basic observation is that the probability that S contains at least one of the relevant variables of f when f depends on more than $2k$ variables, is some constant multiplicative factor (greater than 1) larger than the probability that this occurs when f depends on at most k relevant variables. The second observation is that given the promise that f is a linear function, using a small number of queries we can distinguish with high constant probability between the case that S contains at least one relevant variable of f , and the case that it contains no such variable. By quantifying the above more precisely, and repeating the aforementioned process a sufficient number of times, we can obtain Theorem 3.

The algorithm for degree- d polynomials is essentially the same, except that the sub-test for determining whether S contains any relevant variables is more costly. The same ideas are also the basis for the algorithm for general functions, only we need a more careful analysis since in a general function we may have relevant variables that have very small influence. Indeed, as in previous work on testing k -juntas [7,3,4], the influence of variables (and subsets of variables), plays a central role (and we use some of the claims presented in previous work).

ORGANIZATION. We start by introducing several definitions and basic claims in Section 2. In Section 3 we prove Theorems 1 and 2 (the lower and upper bounds for general functions). In Section 4 we describe our results for restricted function classes, where the algorithms for linear functions and more generally, for degree- d polynomials, are special cases of a slight variant of the algorithm for general functions. All missing details as well as extensions of our results can be found in the full version of this paper [13].

2 Preliminaries

For two functions $f, g : \{0, 1\}^n \rightarrow \{0, 1\}$, we define the *distance* between f and g as $\Pr_x[f(x) \neq g(x)]$ where x is selected uniformly in $\{0, 1\}^n$. For a family of functions \mathcal{F} and a function f , we define the distance between f and \mathcal{F} as the minimum distance over all $g \in \mathcal{F}$ of the distance between f and g . We say that f is ϵ -far from \mathcal{F} , if this distance is at least ϵ .

Our work refers to the influence of sets of variables on the output of a Boolean function (in a way that will be described presently). As such, we often consider the values that a function f attains conditioned on a certain fixed assignment to some of its variables, e.g., the values f may take when the variables x_1 and x_3 are set to 0. For an assignment σ to a set of variables S we will denote the resulting restricted function by $f_{S=\sigma}$. Thus, $f_{S=\sigma}$ is a function of $\{0, 1\}^{n-|S|}$ variables. When we wish to relate to the variables $\{x_1, \dots, x_n\} \setminus S$ we use the notation \bar{S} .

We now give a definition that is central for this work:

Definition 1. For a function $f : \{0, 1\}^n \rightarrow \{0, 1\}$ we define the influence of a set of variables $S \subseteq \{x_1, \dots, x_n\}$ as $\Pr_{\sigma, y, y'}[f_{\bar{S}=\sigma}(y) \neq f_{\bar{S}=\sigma}(y')]$ where σ is selected uniformly at random from $\{0, 1\}^{n-|S|}$ and y, y' are selected uniformly at random from $\{0, 1\}^{|S|}$. For a fixed function f we denote this value by $I(S)$. When the set S consists of a single variable x_i we may use the notation $I(x_i)$ instead of $I(\{x_i\})$.

Proofs of the following claims can be found in [7]. The first claim tells us that the influence of sets of variables is monotone and subadditive:

Claim 1. Let f be a function from $\{0, 1\}^n$ to $\{0, 1\}$, and let S and T be subsets of the variables x_1, \dots, x_n . It holds that $I(S) \leq I(S \cup T) \leq I(S) + I(T)$.

Definition 2. For a fixed function f the marginal influence of set of variables T with respect to a set of variables S is $I(S \cup T) - I(S)$. We denote this value by $I^S(T)$.

The marginal influence of a set of variables is diminishing:

Claim 2. Let $S, T,$ and W be disjoint sets of variables. For any fixed function f it holds that $I^S(T) \geq I^{S \cup W}(T)$.

The next claim relates between the distance to being a k -junta and the influence of sets of variables.

Claim 3. Let f be a function that is ϵ -far from being a k -junta. Then for every subset S of f 's variables of size at most k , the influence of $\{x_1, \dots, x_n\} \setminus S$ is at least ϵ .

The converse of Claim [3] follows from the definition of influence:

Claim 4. Let f be a function such that for every subset S of f 's variables of size at most k , the influence of $\{x_1, \dots, x_n\} \setminus S$ is at least ϵ . Then f is ϵ -far from being a k -junta.

3 Distinguishing between k -Juntas and Functions Far from Every $(1 + \gamma)k$ -Junta

In this section we prove Theorems [1] and [2] (stated in the introduction).

3.1 The Lower Bound

The lower bound stated in Theorem 1 is achieved by a reduction from the *Distinct Elements* problem. In the Distinct Elements problem an algorithm is given query access to a string s and must compute approximately and with high probability the number of distinct elements contained in s . For a string of length t , this problem is equivalent to approximating the support size of a distribution where the probability for every event is in multiples of $1/t$ [12]. Valiant and Valiant [17] give the following theorem (paraphrased here):

Theorem 6. *For any constant $\varphi > 0$, there exists a pair of distributions p^+, p^- for which each domain element occurs with probability at least $1/t$, satisfying:*

1. $|S(p^+) - S(p^-)| = \varphi \cdot t$, where $S(D) \stackrel{\text{def}}{=} |\{x : \Pr_D[x] > 0\}|$.
2. Any algorithm that distinguishes p^+ from p^- with probability at least $2/3$ must obtain $\Omega(\frac{t}{\log(t)})$ samples.

While the construction in the proof of this theorem relates to distributions where the probability of events is not necessarily a multiple of $1/t$, it carries to the Distinct Elements problem [16].

In our work we use the following implication of this theorem - $\Omega(t/\log(t))$ queries are required to distinguish between a string of length t with $\frac{t}{2}$ distinct elements and one with fewer than $\frac{t}{16}$ distinct elements (for a sufficiently large t)⁴

In what follows we assume $k = n/8$. It is possible to (easily) modify the argument for the case that $k \leq n/8$ by “padding”. We set $\gamma = 1$ (so that $1 + \gamma = 2$), which implies the bound holds for all $\gamma \leq 1$. Using terminology coined by Raskhodnikova et al. [12], we refer to each distinct element in the string as a “color”. We show a reduction that maps strings of length $t = \Theta(n)$ to functions from $\{0, 1\}^n$ to $\{0, 1\}$ such that the following holds: If there exists an algorithm that can distinguish (with high constant probability) between functions that are k -juntas and functions that are ϵ -far from any $2k$ -junta (for a constant ϵ) using q queries, then the algorithm can be used to distinguish between strings with at most $k - \Theta(\log(k))$ colors and strings with at least $8k - \Theta(\log(k))$ colors using q queries.

We begin by describing a parametrized family of functions, which we denote by F_m^n . Each function in F_m^n depends on the first $\log(n)$ variables and on an additional subset of m variables.⁵ The first $\log(n)$ variables are used to determine the identity of one of these m variables, and the value of the function is the assignment to this variable. More formally, for each subset $U \subset \{\log(n) + 1, \dots, n\}$

⁴ We note that allowing a bigger gap between the number of distinct elements (e.g., distinguishing between strings with at least t/d distinct elements for some constant d and strings with at most $t^{1-\alpha}$ distinct elements for a (small) constant α), does not make the distinguishing task much easier: $\Omega(t^{1-\alpha})$ queries are still necessary [12].

⁵ In fact, it depends on an integer number of variables, and thus depends, e.g., on the $\lceil \log n \rceil$ first variables. We ignore this rounding issue throughout the paper, as it makes no difference asymptotically.

of size m and each surjective function $\psi : \{0, 1\}^{\log(n)} \rightarrow U$, we have a function $f^{U,\psi}$ in F_m^n where $f^{U,\psi}(y_1, \dots, y_n) = y_{\psi(y_1, \dots, y_{\log(n)})}$. For a given function $f^{U,\psi}$ we call the variables $\{x_i\}_{i \in U}$ *active variables*.

Claim 5. *For any constant value c and for $t > n/c$, every function in $F_{t/2}^n$ is ϵ -far from all $t/4$ -juntas, for a constant value ϵ .*

Proof: From Claim 4 we know that it suffices to show that for every function $f \in F_{t/2}^n$, and for every set of variables $S \subset \{x_1, \dots, x_n\}$ having size at most $t/4$, the set of variables $\bar{S} = \{x_1, \dots, x_n\} \setminus S$ has influence at least ϵ for a constant ϵ .

Consider a particular function $f \in F_{t/2}^n$. For any set S having size at most $t/4$, the set \bar{S} contains at least $t/4$ active variables. We next show that the influence of a set T of $t/4$ active variables is at least $1/8c$, and by the monotonicity of the influence (Claim 1) we are done. The influence of T is defined as $\Pr_{\sigma,y,y'}(f_{\bar{T}=\sigma}(y) \neq f_{\bar{T}=\sigma}(y'))$ where σ is selected uniformly at random from $\{0, 1\}^{n-|T|}$ and y, y' are selected uniformly at random from $\{0, 1\}^{|T|}$. The probability of $x_{\psi(\sigma_1, \dots, \sigma_{\log(n)})}$ belonging to T is at least $|T|/n = t/4 \geq n/4c$. The probability of this coordinate having different values in y and y' is $1/2$, and the claim follows. ■

We now introduce the reduction $R(s)$, which maps a string of colors s (a potential input to the distinct elements problem) to a function from $\{0, 1\}^n$ to $\{0, 1\}$ (a potential input to the “ k -junta vs. far from $(1 + \gamma)k$ -junta” problem):

Let s be a string of length n , where every element i in s gets a color from the set $\{1, \dots, n - \log(n)\}$, which we will denote by $s[i]$. The mapping $R(s) = f$ maps a string with m colors to a function in F_m^n . Informally, we map each color to one of the variables $x_{\log(n)+1}, \dots, x_n$ in f 's input, and compute $f(y_1, \dots, y_n)$ by returning the value of the variable that corresponds to the color of the element in s indexed by the values $y_1, \dots, y_{\log(n)}$. More precisely, let $b : \{0, 1\}^{\log(n)} \rightarrow \{0, \dots, n - 1\}$ be the function that maps the binary representation of a number to that number, e.g., $b(010) = 2$. We define the function f (that corresponds to a string s) as follows: $f(y_1, \dots, y_n) = y_{s[b(y_1, \dots, y_{\log(n)})] + \log(n)}$ (recall that the colors of s range from 1 to $n - \log(n)$).

The next claim follows directly from the definition of the reduction.

Claim 6. *The reduction $R(s)$ has the following properties:*

1. *For a string s , each query to the function $f = R(s)$ of the form $f(y_1, \dots, y_n)$ can be answered by performing a single query to s .*
2. *For a string s with $n/2$ colors the function $f = R(s)$ belongs to $F_{n/2}^n$.*
3. *For a string s with $n/16$ colors the function $f = R(s)$ belongs to $F_{n/16}^n$.*

By Claims 5 and 6, any algorithm that can distinguish (with high constant probability) between functions that are $n/8$ -juntas and functions that are ϵ -far from all $n/4$ -juntas can be used to distinguish (with high constant probability) between strings with $n/2$ distinct elements and strings with $n/16$ distinct elements. Given the lower bound from [17], we have that any algorithm that distinguishes

(with high constant probability) between functions with at most $n/8$ relevant variables and functions that are ϵ -far from all functions with at most $n/4$ relevant variables must perform $\Omega(n/\log(n))$ queries.

3.2 The Algorithm

In this subsection we present the algorithm referred to in Theorem 2. This algorithm uses the procedure **Test-for-relevant-variables** (given in Figure 1), which performs repetitions of the *independence test* defined in 7. The number of repetitions depends on the parameters η and δ , which the algorithm receives as input.

Test-for-relevant-variables
 Input: Oracle access to a function f , a set S of variables to examine, an influence parameter η and a confidence parameter δ .

1. Repeat the following $m = \Theta(\log(1/\delta)/\eta)$ times:
 - (a) Select $\sigma \in \{0, 1\}^{n-|S|}$ uniformly at random.
 - (b) Select two values $y, y' \in \{0, 1\}^{|S|}$ uniformly at random. If $f_{\bar{\sigma}=\sigma}(y) \neq f_{\bar{\sigma}=\sigma}(y')$ return **true**.
2. Return **false**.

Fig. 1. Test-for-relevant-variables

Claim 7. *When given access to a function f , a set S , and parameters η and δ , where S has influence of at least η , **Test-for-relevant-variables** returns **true** with probability at least $1 - \delta$. When S contains no relevant variables, **Test-for-relevant-variables** returns **false** with probability 1. It performs $\Theta(\log(1/\delta)/\eta)$ queries.*

Claim 7 follows directly from the definition of influence and a standard amplification argument.

Proof of Theorem 2: We prove that the statement in the theorem holds for Algorithm **Separate- k -from- $(1 + \gamma)k$** , given in Figure 2. For a function f that has at most k relevant variables (i.e., is a k -junta), the probability that S (created in Step 1a of **Separate- k -from- $(1 + \gamma)k$**) contains at least one such relevant variable is (at most) $p_k = 1 - (1 - \frac{1}{2k})^k$ (note that $1/4 < p_k \leq 1/2$). It follows from the one-sided error of **Test-for-relevant-variables** that the probability that it will return **true** in Step 1b is at most this p_k . We will show that if f is ϵ -far from every $(1 + \gamma)k$ -junta, then the probability of **Test-for-relevant-variables** returning **true** in Step 1b is at least $p'_k = p_k + \Omega(\gamma)$. Having established this, the correctness of **Separate- k -from- $(1 + \gamma)k$** follows by setting the threshold τ to $\tau = (p_k + p'_k)/2$.

In the following we assume that when applied to a subset of the variables with influence at least η , **Test-for-relevant-variables** executed with the influence

Separate- k -from- $(1 + \gamma)k$

Input: Oracle access to a function f , an approximation parameter $\gamma < 1$ and a distance parameter ϵ .

1. Repeat the following $m = \Theta(1/\gamma^2)$ times:
 - (a) Select a subset S of the variables, including each variable in S independently with probability $1/2k$.
 - (b) Run **Test-for-relevant-variables** on f and S , with influence parameter $\eta = \Theta(\epsilon/k)$ and with confidence parameter $\delta = 1/8m$.
2. If the fraction of times **Test-for-relevant-variables** returned true passes a threshold τ , return **more-than- $(1 + \gamma)k$** . Otherwise return **up-to- k** . We determine τ in the analysis.

Fig. 2. Separate- k -from- $(1 + \gamma)k$

parameter η , returns **true**. We will later factor the probability of this not happening in even one iteration of the algorithm into our analysis of the algorithm’s probability of success.

Consider a function f that is ϵ -far from every $(1 + \gamma)k$ -junta. For such a function, and for any constant $c > 1$, by Claim 3 one of the following must hold.

1. There are at least $(1 + \gamma)k$ variables in f each with influence at least $\epsilon/c(1 + \gamma)k$.
2. There are (more than $c(1 + \gamma)k$) variables each with influence less than $\epsilon/c(1 + \gamma)k$ that have, as a set, an influence of at least ϵ .

To verify this, note that if Case 1 does not hold, then there are fewer than $(1 + \gamma)k$ variables in f with influence at least $\epsilon/c(1 + \gamma)k$. Recall that by Claim 3, the variables of f except for the $(1 + \gamma)k$ most influential variables have a total influence of at least ϵ , giving us Case 2.

We first deal with Case 1 (which is the simpler case). We wish to show that the probability that S (as selected in Step 1a) contains at least one variable with influence $\Omega(\epsilon/(1 + \gamma)k)$ is $p_k + \Omega(\gamma)$. As there are at least $(1 + \gamma)k$ variables with influence $\Omega(\epsilon/(1 + \gamma)k)$, it suffices to consider the influence attributed to these variables, and to bound from below the probability that at least one of them appears in S . If we consider these $(1 + \gamma)k$ variables one after the other (in an arbitrary order), for the first k variables, the probability that (at least) one of them is assigned to S is p_k (as defined above). If none of these were assigned to S , an event that occurs with probability at least $1 - p_k \geq 1/2$, we consider the additional γk variables. The probability of at least one of them being selected is at least γp_k , and so we have that the total probability of S containing at least one variable with influence $\Omega(\epsilon/(1 + \gamma)k)$ is at least $p_k(1 + \gamma/2)$. Given that $p_k > 1/4$ we have that the probability is at least $p_k + \gamma/8$, as required.

For our analysis of Case 2 we will focus on the set of variables described in the case. Recall that this set has influence of at least ϵ while every variable in the set has influence of less than $\epsilon/c(1 + \gamma)k$. We denote this set of variables by $Y = \{y_1, \dots, y_\ell\}$. We wish to bound from below the influence of

subsets of Y . To this end we assign to each variable from the set Y a value that bounds from below the marginal influence it has when added to any subset of Y . By the premise of the claim we have that $I(Y) \geq \epsilon$. We consider the values $I(y_1), I^{\{y_1\}}(y_2), \dots, I^{\{y_1, \dots, y_{\ell-1}\}}(y_\ell)$. The sum of these must be at least ϵ by the definition of marginal influence (Definition 2). Let us denote by $I'(y_i)$ the value $I^{\{y_1, \dots, y_{i-1}\}}(y_i)$. We refer to this as *the* marginal influence of y_i . If we consider adding (with probability $1/2k$) each element in Y to S in the order y_1, \dots, y_ℓ , we get by Claim 2 that the total influence of S is no less than the total of the marginal influences of those variables added to S . It now suffices to show that the sum of marginal influences in S is likely to be at least $\epsilon/4k$, and we are done. This can be established using a multiplicative Chernoff bound.

We now turn to lower bounding the algorithm’s probability of success. By the choice of $\delta = 1/8m$, the probability that *any* of the m runs of **Test-for-relevant-variables** fails to detect a set with influence $\Omega(\epsilon/(1 + \gamma)k)$ is at most $1/8$. Conversely, when the set S contains no variables with influence, **Test-for-relevant-variables** never accepts. Thus, for a function with at most k relevant variables, **Test-for-relevant-variables** accepts with probability at most p_k . On the other hand, for a function that is ϵ -far from all functions with at most $(1 + \gamma)k$ relevant variables, **Test-for-relevant-variables** accepts with probability at least $p_k + \gamma/8$. We therefore set the threshold τ to $p_k + \gamma/16$. Recall that the number of iterations performed by the algorithm is $m = \Theta(1/\gamma^2)$. By an additive Chernoff bounds (for a sufficiently large constant in the Θ notation), conditioned on **Test-for-relevant-variables** returning a correct answer in each iteration, the probability that we “fall on the wrong side of the threshold” is at most $1/8$. Thus, with probability at least $3/4$ our algorithm returns a correct answer.

Finally, we bound the query complexity of the algorithm. The algorithm perform $m = \Theta(1/\gamma^2)$ iterations. In each iteration it runs **Test-for-relevant-variables** with influence parameter $\eta = \Theta(\epsilon/k)$ and with confidence parameter $\delta = 1/8m$. The query complexity of the procedure **Test-for-relevant-variables** is $\Theta(\log(1/\delta)/\eta)$, giving a total of $\Theta(\frac{k \log(1/\gamma^2)}{\gamma^2 \epsilon})$ queries. ■

4 Restricting the Problem to Classes of Functions

Given that in general, distinguishing between functions that are k -juntas and functions that are ϵ -far from $(1 + \gamma)k$ juntas requires an almost linear dependence on k , we ask whether this task can be performed more efficiently for restricted function classes (and possibly without the introduction of the distance parameter ϵ). In particular, let \mathcal{C}_η be the class of functions where every variable has influence at least η . As we shall see later, there are natural families of functions that are subclasses of \mathcal{C}_η .

Theorem 7. *Given query access to a function $f \in \mathcal{C}_\eta$, it is possible to distinguish with high constant probability between the case that f has at most k relevant variables and the case that f has more than $(1 + \gamma)k$ relevant variables by performing $\Theta(\frac{\log(1/\gamma)}{\gamma^2 \eta})$ queries.*

Proof: We use the exact same algorithm as we use in the general case (that is, **Separate- k -from-** $(1 + \gamma)k$ given in Figure 2) with the following exception. In Step 11, instead of setting the influence parameter to $\Theta(\epsilon/k)$, we set it to $\Theta(\eta)$. The proof of correctness follows Case 1 in the general proof of correctness. ■

4.1 Linear Functions

A well studied class of functions for which we can test whether a function in the class has k relevant variables or more than $(1 + \gamma)k$ relevant variables, by performing a number of queries that depends only on γ , is the class of linear functions. For each function in the class, every influential variable has influence $1/2$. As a corollary of Theorem 7 we get Theorem 3 (stated in the introduction).

4.2 Polynomials over $GF(2)$

It is well known that every Boolean function can be represented by a polynomial over $GF(2)$. A common measure of the simplicity of such a polynomial is its maximum degree d . The upper bound for determining whether a degree- d polynomial has at most k relevant variables or more than $(1 + \gamma)k$ relevant variables, stated in Theorem 4, follows from Theorem 7 and from the fact that every relevant variable in a polynomial of degree d has influence $\Omega(2^{-d})$.

The lower bound for polynomials of degree d , stated in Theorem 5, uses a reduction from the distinct elements problem just as in the general case. Here, however, the families of functions that strings are mapped to must be realizable by degree- d polynomials where the number of relevant variables may be greater than 2^d . We describe a parametrized family of functions, which we denote $F_{m,d}^n$. Each function in $F_{m,d}^n : \{0, 1\}^n \rightarrow \{0, 1\}$ is a polynomial of degree d that depends on the first $d-1$ variables and on an additional subset of m variables. The setting of the first $d-1$ variables determines a particular subset of the m variables, of size $(n-d+1)/2^{d-1}$, and the value of f is the parity of the variables in this subset. For the reduction we map strings of length 2^{d-1} to functions with an arbitrarily large input size n .

References

1. Alon, N., Dar, S., Parnas, M., Ron, D.: Testing of clustering. *SIAM Journal on Discrete Math.* 16(3), 393–417 (2003)
2. Blais, E., Brody, J., Matulef, K.: Property testing lower bounds via communication complexity. In: To appear in the 26th Conference on Computational Complexity, CCC (2011)
3. Blais, E.: Improved bounds for testing juntas. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 317–330. Springer, Heidelberg (2008)
4. Blais, E.: Testing juntas nearly optimally. In: Proceedings of the Forty-First Annual ACM Symposium on the Theory of Computing, pp. 151–158 (2009)

5. Chockler, H., Gutfreund, D.: A lower bound for testing juntas. *Information Processing Letters* 90(6), 301–305 (2004)
6. Chakradorty, S., García-Soriano, D., Matsliah, A.: Private communication (2010)
7. Fischer, E., Kindler, G., Ron, D., Safra, S., Samorodnitsky, S.: Testing juntas. *Journal of Computer and System Sciences* 68(4), 753–787 (2004)
8. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. *Journal of the ACM* 45(4), 653–750 (1998)
9. Goldreich, O.: On testing computability by small width OBDDs. In: *Proceedings of the Fourteenth International Workshop on Randomization and Computation (RANDOM)*, pp. 574–587 (2010)
10. Kearns, M., Ron, D.: Testing problems with sub-learning sample complexity. *Journal of Computer and System Sciences* 61(3), 428–456 (2000)
11. Parnas, M., Ron, D.: Testing the diameter of graphs. *Random Structures and Algorithms* 20(2), 165–183 (2002)
12. Raskhodnikova, S., Ron, D., Shpilka, A., Smith, A.: Strong lower bounds for approximating distributions support size and the distinct elements problem. *SIAM Journal on Computing* 39(3), 813–842 (2009)
13. Ron, D., Tsur, G.: On approximating the number of relevant variables in a function. Technical Report TR11-041, *Electronic Colloquium on Computational Complexity, ECCC* (2011)
14. Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing* 25(2), 252–271 (1996)
15. Valiant, P.: Testing symmetric properties of distributions. In: *Proceedings of the Fourtieth Annual ACM Symposium on the Theory of Computing*, pp. 383–392 (2008)
16. Valiant, P.: Private communications (2011)
17. Valiant, G., Valiant, P.: Estimating the unseen: an $n/\log(n)$ -sample estimator for entropy and support size, shown optimal via new CLTs. In: *Proceedings of the Forty-Third Annual ACM Symposium on the Theory of Computing*, pp. 685–694 (2011); See also ECCC TR10-179 and TR10-180

Query Complexity in Errorless Hardness Amplification

Thomas Watson*

Computer Science Division
University of California, Berkeley, CA, USA
tom@cs.berkeley.edu

Abstract. An errorless circuit for a boolean function is one that outputs the correct answer or “don’t know” on each input (and never outputs the wrong answer). The goal of errorless hardness amplification is to show that if f has no size s errorless circuit that outputs “don’t know” on at most a δ fraction of inputs, then some f' related to f has no size s' errorless circuit that outputs “don’t know” on at most a $1 - \epsilon$ fraction of inputs. Thus the hardness is “amplified” from δ to $1 - \epsilon$. Unfortunately, this amplification comes at the cost of a loss in circuit size. This is because such results are proven by reductions which show that any size s' errorless circuit for f' that outputs “don’t know” on at most a $1 - \epsilon$ fraction of inputs could be used to construct a size s errorless circuit for f that outputs “don’t know” on at most a δ fraction of inputs. If the reduction makes q queries to the hypothesized errorless circuit for f' , then plugging in a size s' circuit yields a circuit of size $\geq qs'$, and thus we must have $s' \leq s/q$. Hence it is desirable to keep the query complexity to a minimum. The first results on errorless hardness amplification were obtained by Bogdanov and Safra (FOCS 2007). They achieved query complexity $O((\frac{1}{\delta} \log \frac{1}{\epsilon})^2 \cdot \frac{1}{\epsilon} \log \frac{1}{\delta})$ when f' is the XOR of several independent copies of f . We improve the query complexity (and hence the loss in circuit size) to $O(\frac{1}{\epsilon} \log \frac{1}{\delta})$, which is optimal up to constant factors for nonadaptive black-box errorless hardness amplification. Bogdanov and Safra also proved a result that allows for errorless hardness amplification within NP. They achieved query complexity $O(k^3 \cdot \frac{1}{\epsilon^2} \log \frac{1}{\delta})$ when f' consists of any monotone function applied to the outputs of k independent copies of f , provided the monotone function satisfies a certain combinatorial property parameterized by δ and ϵ . We improve the query complexity to $O(\frac{k}{t} \cdot \frac{1}{\epsilon} \log \frac{1}{\delta})$, where $t \geq 1$ is a certain parameter of the monotone function.

Keywords: hardness amplification, query complexity.

* This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-0946797 and by the National Science Foundation under Grant No. CCF-1017403.

1 Introduction

Traditionally, an algorithm for solving a computational problem is required to be correct on all inputs and is judged in terms of its efficiency (the amount of computational resources it uses). One criticism of this model is that it is too strict: In practice, an algorithm only needs to be correct on “real-world” inputs and not on contrived worst-case inputs. To address this issue within the framework of complexity theory, researchers developed the theory of average-case complexity (starting with the work of Levin [7]). In this theory, an algorithm is judged in terms of both its efficiency and the fraction of inputs on which it fails to solve the problem correctly. The topic of this paper is the relationship between these two measures of the quality of an algorithm.

There are two standard settings for average-case complexity. In the original setting proposed by Levin [7], one only considers *errorless algorithms*, which are required to output the correct answer or “don’t know” on each input. An errorless algorithm is judged in terms of both its efficiency and the fraction of inputs on which it outputs “don’t know”. We refer to this setting as *errorless average-case complexity*. In the other setting, one considers arbitrary algorithms which may output the wrong answer rather than just “don’t know” on an input. We refer to this setting as *non-errorless average-case complexity*. Errorless average-case complexity is an intermediate setting between worst-case complexity and non-errorless average-case complexity.

We first discuss non-errorless average-case complexity. A boolean function is said to be *mildly average-case hard* if no efficient algorithm can compute it on almost all inputs. Applications such as derandomization and cryptography require functions that are *strongly average-case hard*, meaning that no efficient algorithm can compute the function on noticeably more than half the inputs. This motivates hardness amplification, which is the problem of transforming a mildly average-case hard function into a strongly average-case hard function. A classic result in this area is the XOR Lemma [8,5,13], which states that the XOR of sufficiently many independent copies of a mildly average-case hard function is strongly average-case hard, provided the model of efficient algorithms is small circuits.

However, the XOR Lemma (as well as the numerous subsequent results on hardness amplification) incurs an unfortunate loss in circuit size. Suppose the original function f is mildly average-case hard in the sense that no size s circuit succeeds on at least a $1 - \delta$ fraction of inputs, and we wish for the new function f' to be strongly average-case hard in the sense that no size s' circuit succeeds on at least a $1/2 + \epsilon$ fraction of inputs. Then we would like s' to be as large as possible, but the XOR Lemma requires that s' is actually smaller than s . This is because such results are proven by reductions which show that if f' is not strongly average-case hard, then a circuit witnessing this could be used to construct a circuit witnessing that f is not mildly average-case hard. If the reduction makes q queries to the hypothesized circuit, then plugging in a size s' circuit yields a circuit of size $\geq qs'$, and thus we must have $s' \leq s/q$. Hence the query complexity q governs the loss in circuit size. For the XOR Lemma,

the query complexity is well-understood. The proof due to Impagliazzo [5] and Klivans and Servedio [6] shows that $q = O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ queries are sufficient, and Shaltiel and Viola [10] showed that in a certain sense, $q = \Omega(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ queries are necessary.

Bogdanov and Safra [2] initiated the study of hardness amplification in Levin’s original setting of errorless average-case complexity. A boolean function is said to be *mildly errorless average-case hard* if no efficient errorless algorithm (say, size s circuit) can compute it on almost all inputs (say, a $1 - \delta$ fraction). A function is said to be *strongly errorless average-case hard* if no efficient errorless algorithm (say, size s' circuit) can compute it on a noticeable fraction of inputs (say, an ϵ fraction). Note that in the non-errorless setting, computing a boolean function on half the inputs is trivial (using constant 0 or constant 1), but in the errorless setting, computing a boolean function on even a small fraction of inputs is nontrivial. The goal of errorless hardness amplification is to transform a mildly errorless average-case hard function f into a strongly errorless average-case hard function f' . Such results suffer from a loss in circuit size for the same reason as in the non-errorless setting. Bogdanov and Safra [2] showed that $q = O((\frac{1}{\delta} \log \frac{1}{\epsilon})^2 \cdot \frac{1}{\epsilon} \log \frac{1}{\delta})$ queries are sufficient when f' is the XOR of several independent copies of f . The result of Shaltiel and Viola [10] can be modified without difficulty to show that in a certain sense, $q = \Omega(\frac{1}{\epsilon} \log \frac{1}{\delta})$ queries are necessary. We close the gap by showing that $q = O(\frac{1}{\epsilon} \log \frac{1}{\delta})$ queries are sufficient.

Another natural goal for hardness amplification is to guarantee that if f represents an NP language at some input length, then f' also represents an NP language at some input length. In the non-errorless setting this goal has been studied, for example, in [9,4], and in the errorless setting this goal has been studied by Bogdanov and Safra [2]. We significantly improve the query complexity of the Bogdanov-Safra result.

1.1 The Errorless XOR Lemma

Given $f : \{0, 1\}^n \rightarrow \{0, 1\}$ we define $f^{\oplus k} : \{0, 1\}^{n \times k} \rightarrow \{0, 1\}$ as follows: $f^{\oplus k}(x_1, \dots, x_k) = f(x_1) \oplus \dots \oplus f(x_k)$.

Definition 1 (Errorless Average-Case Hardness). We say a circuit $A : \{0, 1\}^n \rightarrow \{0, 1, \perp\}$ is a δ -errorless circuit for $f : \{0, 1\}^n \rightarrow \{0, 1\}$ if

- (i) $A(x) \in \{f(x), \perp\}$ for all $x \in \{0, 1\}^n$, and
- (ii) $\Pr_x[A(x) = \perp] \leq \delta$ where $x \in \{0, 1\}^n$ is chosen uniformly at random.

We say f is (s, δ) -hard if it has no δ -errorless circuit of size $\leq s$.

Theorem 1 (Query-Optimal Errorless XOR Lemma). If f is (s, δ) -hard then $f' = f^{\oplus k}$ is $(s', 1 - \epsilon)$ -hard where $s' = s / (\frac{4}{\epsilon} \ln \frac{2}{\delta})$, provided $k \geq \frac{16}{\delta} \ln \frac{2}{\epsilon}$.

We prove Theorem 1 in Section 2. Bogdanov and Safra [2] proved a version of Theorem 1 where $s' = s / (k^2 \cdot \frac{2}{\epsilon} \ln \frac{2}{\delta})$, provided $k \geq \frac{2}{\delta} \ln \frac{2}{\epsilon}$. Even for the best value of k , they only achieve $O((\frac{1}{\delta} \log \frac{1}{\epsilon})^2 \cdot \frac{1}{\epsilon} \log \frac{1}{\delta})$ query complexity. Also, our bound on the query complexity does not depend on k .

We prove Theorem [1](#) by a reduction similar to the one used in [\[2\]](#). Our contribution is a new, tight analysis of the reduction. The crux of the reduction is a randomized procedure that solves f errorlessly (meaning that for each input x it may output $\overline{f(x)}$ with some probability and \perp with some probability, but it never outputs $f(x)$) while making one query to a hypothesized $(1 - \epsilon)$ -errorless circuit A' for f' . Suppose for some $\beta > 0$ we knew that $\leq \delta/2$ fraction of inputs x are bad in the sense that the probability the procedure outputs $f(x)$ is $< \beta$. Then by amplifying the success probability on the good inputs and hard-wiring the randomness appropriately, we obtain a δ -errorless circuit A for f , via a reduction with query complexity $O(\frac{1}{\beta} \log \frac{1}{\delta})$. The heart of our improvement over the Bogdanov-Safra proof is in arguing that we can take $\beta = \epsilon/4$. To prove this, we suppose the fraction of bad inputs is $> \delta/2$ and prove that then A' must compute f' on $< \epsilon$ fraction of inputs. The procedure outputs $f(x)$ if and only if the query is an input on which A' computes f' ; furthermore the distribution of this query (x_1, \dots, x_k) is obtained by setting $x_i = x$ for a uniformly random i and picking $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k$ uniformly at random. Consider the following two distributions on queries to A' : the uniform distribution, and the distribution obtained by picking a random bad x and running the procedure on input x . We know A' computes f' with probability $< \beta = \epsilon/4$ under the latter distribution, and we wish to show that A' computes f' with probability $< \epsilon$ under the former. For this, we show the two distributions are “close” in the sense that the probability of any event under the former is less than twice the probability under the latter plus $\epsilon/2$. The argument involves a dichotomy: Since we assume a large fraction of x 's are bad, a uniform query is unlikely to have few bad coordinates. Assuming there are many bad coordinates, we can essentially pretend there is one bad coordinate and then argue that we have overcounted the probability by a lot. This is the intuition behind the proof of Theorem [1](#).

It can be shown that $\Omega(\frac{1}{\epsilon} \log \frac{1}{\delta})$ queries are needed by any nonadaptive black-box reduction achieving errorless hardness amplification, regardless of how f' is constructed from f (see the full version of this paper for the precise statement). Since our proof of Theorem [1](#) (and the Bogdanov-Safra proof) is by a nonadaptive black-box reduction, this shows that Theorem [1](#) is optimal in a sense. Shaltiel and Viola [\[10\]](#) gave a general technique for lower bounding the query complexity of nonadaptive black-box reductions in various settings, and we observe that their technique applies to errorless hardness amplification. Artemenko and Shaltiel [\[1\]](#) have proven a $\Omega(\frac{1}{\epsilon})$ query lower bound even for *adaptive* black-box reductions. The optimal $\Omega(\frac{1}{\epsilon} \log \frac{1}{\delta})$ lower bound for adaptive reductions remains open.

1.2 Monotone Errorless Amplification

Consider the problem of errorless hardness amplification within NP. That is, if f is computable in nondeterministic polynomial time, then we want f' to also be computable in nondeterministic polynomial time. Taking $f' = f^{\oplus k}$ does not guarantee this. We instead consider more general constructions of the form $f' = C \circ f^k$ where $C : \{0, 1\}^k \rightarrow \{0, 1\}$, and $f^k : \{0, 1\}^{n \times k} \rightarrow \{0, 1\}^k$ is defined as $f^k(x_1, \dots, x_k) = (f(x_1), \dots, f(x_k))$. In the setting of the XOR Lemma, the

combiner function C is the k -bit parity function. If C is monotone (that is, $C(y_1, \dots, y_k) \leq C(z_1, \dots, z_k)$ whenever $y_i \leq z_i$ for all $i \in [k]$) and f and C are both computable in nondeterministic polynomial time, then f' is computable in nondeterministic polynomial time. This approach dates back to [9,4].

Bogdanov and Safra [2] showed that this construction yields errorless hardness amplification provided the monotone combiner function C satisfies a certain combinatorial property. To describe this property, we need some definitions from [2] (though we use somewhat different notation). Fix $b \in \{0, 1\}$. Given a monotone function $C : \{0, 1\}^k \rightarrow \{0, 1\}$ and a string $y \in \{0, 1\}^k$, we say that coordinate $i \in [k]$ is b -sensitive if flipping the i th bit of y causes the value of C to flip from b to \bar{b} , and we let $\sigma(C, y, b)$ denote the set of b -sensitive coordinates. That is,

$$\sigma(C, y, b) = \{i \in [k] : C(y) = b \text{ and } C(y \oplus e_i) = \bar{b}\}.$$

Note that if $C(y) = \bar{b}$ then $\sigma(C, y, b) = \emptyset$ and if $C(y) = b$ then by the monotonicity of C , $\sigma(C, y, b)$ only contains coordinates i such that $y_i = b$. For $p \in [0, 1]$, we use $y \sim_p \{0, 1\}^k$ to denote that y is sampled from the p -biased distribution, that is, each bit is independently set to 1 with probability p .

Definition 2. For $b \in \{0, 1\}$, a function $C : \{0, 1\}^k \rightarrow \{0, 1\}$ is a (t, ρ, p, b) -amplifier if C is monotone and

$$\Pr_{y \sim_p \{0, 1\}^k} [|\sigma(C, y, b)| \geq t] \geq 1 - \rho.$$

Note that a monotone function $C : \{0, 1\}^k \rightarrow \{0, 1\}$ is a (t, ρ, p, b) -amplifier if and only if its monotone complement $C^\dagger : \{0, 1\}^k \rightarrow \{0, 1\}$ is a $(t, \rho, 1 - p, \bar{b})$ -amplifier, where C^\dagger is defined as $C^\dagger(y_1, \dots, y_k) = C(\bar{y}_1, \dots, \bar{y}_k)$.

For reasons discussed in [2], it is necessary to consider the following one-sided version of Definition [2].

Definition 3 (One-Sided Errorless Average-Case Hardness). For $b \in \{0, 1\}$, we say a circuit $A : \{0, 1\}^n \rightarrow \{0, 1, \perp\}$ is a (δ, b) -errorless circuit for $f : \{0, 1\}^n \rightarrow \{0, 1\}$ if

- (i) $A(x) \in \{f(x), \perp\}$ for all $x \in \{0, 1\}^n$, and
- (ii) $\Pr_x[A(x) = \perp] \leq \delta$ where $x \in f^{-1}(b)$ is chosen uniformly at random.

We say f is (s, δ, b) -hard if it has no (δ, b) -errorless circuit of size $\leq s$.

Note that if f is (s, δ) -hard then f is either $(s/2, \delta, 0)$ -hard or $(s/2, \delta, 1)$ -hard.

Theorem 2 (Monotone Errorless Amplification Lemma). For $b \in \{0, 1\}$, if f is (s, δ, b) -hard and $C : \{0, 1\}^k \rightarrow \{0, 1\}$ is a (t, ρ, p, b) -amplifier then $f' = C \circ f^k$ is $(s', 1 - \epsilon)$ -hard where $s' = s / (\frac{k}{t} \cdot \frac{4}{\epsilon} \ln \frac{2}{\delta})$, provided $t \geq \frac{16}{\delta} \ln \frac{4}{\epsilon}$, $\rho \leq \epsilon/4$, and $p = \Pr_x[f(x) = 1]$.

We prove Theorem [2] in Section [3]. Bogdanov and Safra [2] proved a version of Theorem [2] where $s' = s / (k^3 \cdot \frac{64}{\epsilon^2} \ln \frac{2}{\delta})$, provided $t \geq \frac{4}{\delta} \ln \frac{8}{\epsilon}$ and $\rho \leq \epsilon/2$. Their argument involves considering the subcubes of $\{0, 1\}^{n \times k}$ given by $f^k(x_1, \dots, x_k) =$

y for each y individually and then combining the results for the different subcubes using a nontrivial probabilistic argument. We show how to give a direct argument that handles all the subcubes simultaneously. This idea alone actually simplifies the proof and reduces the query complexity to $O(k^2 \cdot \frac{1}{\epsilon} \log \frac{1}{\delta})$. Combining this idea with the ideas from our analysis in the proof of Theorem 1 allows us to further reduce the query complexity to $O(\frac{k}{t} \cdot \frac{1}{\epsilon} \log \frac{1}{\delta})$. We believe this bound on the query complexity cannot be improved without exploiting some non-obvious structural property of (t, ρ, p, b) -amplifiers; however, we could not come up with a compelling way to formalize this.

Bogdanov and Safra [2] showed how to construct good amplifiers (with large t and small ρ) and how to use Theorem 2 with these amplifiers to do uniform and nonuniform errorless hardness amplification within NP.

1.3 Preliminaries

We use the following standard Chernoff bound several times.

Theorem 3. *If X_1, \dots, X_τ are fully independent indicator random variables each with expectation π , then $\Pr [\sum_{j=1}^\tau X_j < \pi\tau/2] < e^{-\pi\tau/8}$.*

2 Proof of Theorem 1

We prove the contrapositive. Suppose f' is not $(s', 1 - \epsilon)$ -hard and thus there is a circuit A' of size $\leq s'$ such that

- (i) $A'(x_1, \dots, x_k) \in \{f'(x_1, \dots, x_k), \perp\}$ for all x_1, \dots, x_k , and
- (ii) $\Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) = \perp] \leq 1 - \epsilon$.

We give a nonuniform reduction that makes $\frac{4}{\epsilon} \ln \frac{2}{\delta}$ nonadaptive queries to A' and combines the results with some trivial computation, yielding a circuit A that witnesses that f is not (s, δ) -hard. To start out, we give a randomized algorithm (Algorithm 1) that solves f errorlessly using oracle access to A' and oracle access to f . The oracle queries to f only depend on the randomness (and not on the input), and later we will hard-wire a particular choice of randomness to get a circuit without oracle access to f .

Define the good set

$$G = \left\{ x \in \{0, 1\}^n : \Pr_{i, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k} [A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp] \geq \epsilon/4 \right\}$$

and define the bad set $B = \{0, 1\}^n \setminus G$. That is, G is the set of inputs for which each iteration of the loop has at least an $\epsilon/4$ probability of producing output.

Input: $x \in \{0, 1\}^n$
Output: $f(x)$ or \perp

- 1 repeat $\frac{4}{\epsilon} \ln \frac{2}{\delta}$ times
- 2 pick $i \in [k]$ and $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k \in \{0, 1\}^n$ uniformly at random
- 3 if $A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp$ then halt and output
 $f(x_1) \oplus \dots \oplus f(x_{i-1}) \oplus A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \oplus f(x_{i+1}) \oplus \dots \oplus f(x_k)$
- 4 end
- 5 halt and output \perp

Algorithm 1. Reduction for Theorem □

Proposition 1. $|B| \leq (\delta/2) \cdot 2^n$.

Proof. Suppose for contradiction that $|B| > (\delta/2) \cdot 2^n$, and define $\gamma = |B|/2^{n+1}$. We define the event

$$W = \left\{ (x_1, \dots, x_k) \in \{0, 1\}^{n \times k} : |\{i : x_i \in B\}| \geq \gamma k \right\}.$$

That is, W is the event that at least a γ fraction of coordinates are bad. We have

$$\Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp] \leq \Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W] + \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W]$$

and we show that both terms on the right side are $< \epsilon/2$, thus contradicting property (ii) of A' .

Bounding the first term. Applying Theorem □ with X_i as the indicator variable for $x_i \in B$, and with $\tau = k$ and $\pi = |B|/2^n$, we have

$$\Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W] < e^{-k \cdot |B|/2^{n+3}} < e^{-k\delta/16} \leq \epsilon/2$$

where the middle inequality follows by our assumption on $|B|$ and the last inequality follows by $k \geq \frac{16}{\delta} \ln \frac{2}{\epsilon}$.

Bounding the second term. For each $S \subseteq [k]$ we define the event

$$W_S = \left\{ (x_1, \dots, x_k) \in \{0, 1\}^{n \times k} : \forall i \ x_i \in B \Leftrightarrow i \in S \right\}.$$

Note that the W_S 's are disjoint and $W = \bigcup_{S : |S| \geq \gamma k} W_S$. We have

$$\begin{aligned} & \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W] \\ &= \sum_{S : |S| \geq \gamma k} \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W_S] \end{aligned}$$

$$\begin{aligned}
 &\leq \frac{1}{\gamma k} \sum_{S \subseteq [k]} |S| \cdot \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W_S] \\
 &= \frac{1}{\gamma k} \sum_{i \in [k]} \sum_{S \ni i} \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W_S] \\
 &= \frac{1}{\gamma k} \sum_{i \in [k]} \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } x_i \in B] \\
 &= \frac{1}{\gamma k} \sum_{i \in [k]} \sum_{x \in B} \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \mid x_i = x] \cdot \Pr_{x_1, \dots, x_k} [x_i = x] \\
 &= \frac{1}{\gamma k 2^n} \sum_{x \in B} \sum_{i \in [k]} \Pr_{i, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k} [A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp] \\
 &= \frac{1}{\gamma k 2^n} \sum_{x \in B} k \cdot \Pr_{i, x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k} [A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp] \\
 &< \frac{1}{\gamma k 2^n} \sum_{x \in B} k \cdot \epsilon/4 \\
 &= \frac{\epsilon/4}{\gamma 2^n} \cdot |B| \\
 &= \epsilon/2
 \end{aligned}$$

where the second and fifth lines follow by the disjointness of the W_S 's, and the remaining lines follow by simple rearrangements. \square

The rest of the proof of Theorem 1 is similar to the argument from [2]. First we note that for all $x \in \{0, 1\}^n$ and all choices of randomness, Algorithm 1 does indeed output either $f(x)$ or \perp . This follows trivially from the fact that if $A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp$ then $A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) = f(x_1) \oplus \dots \oplus f(x_{i-1}) \oplus f(x) \oplus f(x_{i+1}) \oplus \dots \oplus f(x_k)$ by property (i) of A' . Next we observe that for each $x \in G$, we have

$$\Pr_{\text{randomness}} [\text{Algorithm 1 outputs } \perp] \leq (1 - \epsilon/4)^{\frac{4}{\epsilon} \ln \frac{2}{\delta}} \leq \delta/2.$$

Therefore

$$\begin{aligned}
 &\Pr_{x, \text{randomness}} [\text{Algorithm 1 outputs } \perp] \\
 &\leq \Pr_x [x \in B] + \mathbb{E}_x \left[\Pr_{\text{randomness}} [\text{Algorithm 1 outputs } \perp] \mid x \in G \right] \\
 &\leq \delta/2 + \mathbb{E}_x [\delta/2 \mid x \in G] \\
 &= \delta
 \end{aligned}$$

where the second inequality follows by Proposition 1 and by the above observation. It follows that there exists a setting of the randomness such that

- (i) Algorithm 1 outputs $f(x)$ or \perp for all x , and
- (ii) $\Pr_x [\text{Algorithm 1 outputs } \perp] \leq \delta$.

To get a circuit A that witnesses that f is not (s, δ) -hard, just hard-wire the randomness and the values of $f(x_1) \oplus \dots \oplus f(x_{i-1}) \oplus f(x_{i+1}) \oplus \dots \oplus f(x_k)$ needed for this choice of randomness, and plug in the hypothesized circuit A' . Since A' has size $\leq s'$ and Algorithm 1 makes $\frac{4}{\epsilon} \ln \frac{2}{\delta}$ queries to A' , A has size $\leq s' \cdot \frac{4}{\epsilon} \ln \frac{2}{\delta} = s$. Note that Algorithm 1 can trivially be implemented with *nonadaptive* access to A' .

3 Proof of Theorem 2

We prove the contrapositive. Suppose f' is not $(s', 1 - \epsilon)$ -hard and thus there is a circuit A' of size $\leq s'$ such that

- (i) $A'(x_1, \dots, x_k) \in \{f'(x_1, \dots, x_k), \perp\}$ for all x_1, \dots, x_k , and
- (ii) $\Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) = \perp] \leq 1 - \epsilon$.

We give a nonuniform reduction that makes $\frac{k}{t} \cdot \frac{4}{\epsilon} \ln \frac{2}{\delta}$ nonadaptive queries to A' and combines the results with some trivial computation, yielding a circuit A that witnesses that f is not (s, δ, b) -hard. To start out, we give a randomized algorithm (Algorithm 2) that solves f errorlessly using oracle access to A' and oracle access to f and $\sigma(C, \cdot, b)$. The oracle queries to f and $\sigma(C, \cdot, b)$ only depend on the randomness (and not on the input), and later we will hard-wire a particular choice of randomness to get a circuit without oracle access to f or $\sigma(C, \cdot, b)$.

Input: $x \in \{0, 1\}^n$
Output: $f(x)$ or \perp

- 1 repeat $\frac{k}{t} \cdot \frac{4}{\epsilon} \ln \frac{2}{\delta}$ times
- 2 pick $i \in [k]$ and $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k \in \{0, 1\}^n$ uniformly at random
- 3 if $A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp$ and $i \in \sigma(C, y, b)$ where

$y = (f(x_1), \dots, f(x_{i-1}), b, f(x_{i+1}), \dots, f(x_k))$
- then halt and output $A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k)$
- 4 end
- 5 halt and output \perp

Algorithm 2. Reduction for Theorem 2

Define the good set

$$G = \left\{ x \in f^{-1}(b) : \Pr_{i, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k} [A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp \text{ and } i \in \sigma(C, y, b)] \geq \epsilon t / 4k \right\}$$

where y is as in line 3 of Algorithm [2](#) and define the bad set $B = f^{-1}(b) \setminus G$. That is, G is the set of inputs in $f^{-1}(b)$ for which each iteration of the loop has at least an $\epsilon t/4k$ probability of producing output.

Proposition 2. $|B| \leq (\delta/2) \cdot |f^{-1}(b)|$.

Proof. Suppose for contradiction that $|B| > (\delta/2) \cdot |f^{-1}(b)|$, and define $\gamma = |B|/2|f^{-1}(b)|$. We define the event

$$W = \left\{ (x_1, \dots, x_k) \in \{0, 1\}^{n \times k} : \left| \left\{ i : x_i \in B \text{ and } i \in \sigma(C, f^k(x_1, \dots, x_k), b) \right\} \right| \geq \gamma t \right\}.$$

That is, W is the event that at least a $\gamma t/k$ fraction of coordinates are both bad and b -sensitive. We have

$$\Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp] \leq \Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W] + \Pr_{x_1, \dots, x_k} [A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W]$$

and we show that both terms on the right side are $< \epsilon/2$, thus contradicting property (ii) of A' .

Bounding the first term. We have

$$\begin{aligned} & \Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W] \\ & \leq \Pr_{x_1, \dots, x_k} [|\sigma(C, f^k(x_1, \dots, x_k), b)| < t] + \\ & \quad \Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W \mid |\sigma(C, f^k(x_1, \dots, x_k), b)| \geq t]. \end{aligned}$$

To show that this is $< \epsilon/2$, we show that the first of the two terms on the right side is $\leq \epsilon/4$ and the second is $< \epsilon/4$. Since C is a (t, ρ, p, b) -amplifier, we have

$$\Pr_{x_1, \dots, x_k} [|\sigma(C, f^k(x_1, \dots, x_k), b)| < t] = \Pr_{y \sim_p \{0, 1\}^k} [|\sigma(C, y, b)| < t] \leq \rho$$

which is at most $\epsilon/4$ by assumption. We have

$$\begin{aligned} & \Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W \mid |\sigma(C, f^k(x_1, \dots, x_k), b)| \geq t] = \\ & \mathbb{E}_{y \sim_p \{0, 1\}^k} \left[\Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W \mid f^k(x_1, \dots, x_k) = y] \mid |\sigma(C, y, b)| \geq t \right]. \end{aligned}$$

Fix any $y \in \{0, 1\}^k$ such that $|\sigma(C, y, b)| \geq t$, and for now let us abbreviate $\sigma(C, y, b)$ as σ . Then we have

$$\begin{aligned} & \Pr_{x_1, \dots, x_k} [(x_1, \dots, x_k) \notin W \mid f^k(x_1, \dots, x_k) = y] \\ & = \mathbb{E}_{(x_i)_{i \notin \sigma}} \left[\Pr_{(x_i)_{i \in \sigma}} [(x_1, \dots, x_k) \notin W \mid f(x_i) = b \ \forall i \in \sigma] \mid f(x_i) = y_i \ \forall i \notin \sigma \right] \end{aligned}$$

since $\sigma \subseteq \{i : y_i = b\}$. Now fix any $(x_i)_{i \notin \sigma}$ such that $f(x_i) = y_i$ for all $i \notin \sigma$. Then we have

$$\begin{aligned} & \Pr_{(x_i)_{i \in \sigma}} \left[(x_1, \dots, x_k) \notin W \mid f(x_i) = b \ \forall i \in \sigma \right] \\ &= \Pr_{(x_i)_{i \in \sigma}} \left[\left| \{i \in \sigma : x_i \in B\} \right| < \gamma t \mid f(x_i) = b \ \forall i \in \sigma \right] \\ &\leq \Pr_{(x_i)_{i \in \sigma}} \left[\left| \{i \in \sigma : x_i \in B\} \right| < \gamma \cdot |\sigma| \mid f(x_i) = b \ \forall i \in \sigma \right] \end{aligned}$$

where the inequality follows by $t \leq |\sigma|$. Applying Theorem 3 with X_j as the indicator variable for $x_i \in B$ where i is the j th value in σ and x_i is chosen uniformly from $f^{-1}(b)$, and with $\tau = |\sigma|$ and $\pi = |B|/|f^{-1}(b)|$, we have that the latter quantity is less than

$$e^{-|\sigma| \cdot |B|/8|f^{-1}(b)|} < e^{-|\sigma| \cdot \delta/16} \leq e^{-t\delta/16} \leq \epsilon/4$$

where the first inequality follows by our assumption on $|B|$, the middle inequality follows by $|\sigma| \geq t$, and the last inequality follows by $t \geq \frac{16}{\delta} \ln \frac{4}{\epsilon}$. This establishes

$$\Pr_{x_1, \dots, x_k} \left[(x_1, \dots, x_k) \notin W \mid \left| \sigma(C, f^k(x_1, \dots, x_k), b) \right| \geq t \right] < \epsilon/4.$$

Bounding the second term. This is similar to the corresponding part of the analysis in the proof of Theorem 1. For each $S \subseteq [k]$ we define the event

$$\begin{aligned} W_S &= \left\{ (x_1, \dots, x_k) \in \{0, 1\}^{n \times k} : \right. \\ &\quad \left. \forall i \left(x_i \in B \text{ and } i \in \sigma(C, f^k(x_1, \dots, x_k), b) \right) \Leftrightarrow i \in S \right\}. \end{aligned}$$

Note that the W_S 's are disjoint and $W = \bigcup_{S : |S| \geq \gamma t} W_S$. Using the shorthand y as in line 3 of Algorithm 2, we have

$$\begin{aligned} & \Pr_{x_1, \dots, x_k} \left[A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W \right] \\ &= \sum_{S : |S| \geq \gamma t} \Pr_{x_1, \dots, x_k} \left[A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W_S \right] \\ &\leq \frac{1}{\gamma t} \sum_{S \subseteq [k]} |S| \cdot \Pr_{x_1, \dots, x_k} \left[A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W_S \right] \\ &= \frac{1}{\gamma t} \sum_{i \in [k]} \sum_{S \ni i} \Pr_{x_1, \dots, x_k} \left[A'(x_1, \dots, x_k) \neq \perp \text{ and } (x_1, \dots, x_k) \in W_S \right] \\ &= \frac{1}{\gamma t} \sum_{i \in [k]} \Pr_{x_1, \dots, x_k} \left[A'(x_1, \dots, x_k) \neq \perp \text{ and } x_i \in B \right. \\ &\quad \left. \text{and } i \in \sigma(C, f^k(x_1, \dots, x_k), b) \right] \\ &= \frac{1}{\gamma t} \sum_{i \in [k]} \sum_{x \in B} \Pr_{x_1, \dots, x_k} \left[A'(x_1, \dots, x_k) \neq \perp \text{ and } \right. \\ &\quad \left. i \in \sigma(C, f^k(x_1, \dots, x_k), b) \mid x_i = x \right] \cdot \Pr_{x_1, \dots, x_k} [x_i = x] \end{aligned}$$

$$\begin{aligned}
 &= \frac{1}{\gamma t 2^n} \sum_{x \in B} \sum_{i \in [k]} \Pr_{x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k} [A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp \\
 &\quad \text{and } i \in \sigma(C, y, b)] \\
 &= \frac{1}{\gamma t 2^n} \sum_{x \in B} k \cdot \Pr_{i, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_k} [A'(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k) \neq \perp \\
 &\quad \text{and } i \in \sigma(C, y, b)] \\
 &< \frac{1}{\gamma t 2^n} \sum_{x \in B} k \cdot \epsilon t / 4k \\
 &= \frac{\epsilon / 4}{\gamma 2^n} \cdot |B| \\
 &\leq \epsilon / 2
 \end{aligned}$$

where the second and fifth lines follow by the disjointness of the W_S 's, the last line follows by $|f^{-1}(b)| \leq 2^n$, and the remaining lines follow by simple rearrangements. For the seventh line, we used the fact that $x \in B$ implies $f(x) = b$ and thus $y = f^k(x_1, \dots, x_{i-1}, x, x_{i+1}, \dots, x_k)$. \square

The rest of the proof of Theorem 2 follows similarly as in Theorem 1.

Acknowledgments. I thank anonymous reviewers for helpful comments.

References

1. Artemenko, S., Shaltiel, R.: Lower Bounds on the Query Complexity of Non-Uniform and Adaptive Reductions Showing Hardness Amplification. In: To appear in Proceedings of the 15th International Workshop on Randomization and Computation (2011)
2. Bogdanov, A., Safra, M.: Hardness Amplification for Errorless Heuristics. In: Proceedings of the 48th IEEE Symposium on Foundations of Computer Science, pp. 418–426 (2007)
3. Goldreich, O., Nisan, N., Wigderson, A.: On Yao's XOR Lemma. Electronic Colloquium on Computational Complexity, Technical Report TR95-050 (1995)
4. Healy, A., Vadhan, S., Viola, E.: Using Nondeterminism to Amplify Hardness. *SIAM Journal on Computing* 35(4), 903–931 (2006)
5. Impagliazzo, R.: Hard-Core Distributions for Somewhat Hard Problems. In: Proceedings of the 36th IEEE Symposium on Foundations of Computer Science, pp. 538–545 (1995)
6. Klivans, A., Servedio, R.: Boosting and Hard-Core Sets. *Machine Learning* 53(3), 217–238 (2003)
7. Levin, L.: Average Case Complete Problems. *SIAM Journal on Computing* 15(1), 285–286 (1986)
8. Levin, L.: One-Way Functions and Pseudorandom Generators. *Combinatorica* 7(4), 357–363 (1987)
9. O'Donnell, R.: Hardness Amplification Within NP. *Journal of Computer and System Sciences* 69(1), 68–94 (2004)
10. Shaltiel, R., Viola, E.: Hardness Amplification Proofs Require Majority. *SIAM Journal on Computing* 39(7), 3122–3154 (2010)

Author Index

- Abdullah, Mohammed 351
Ambainis, Andris 365
Arora, Sanjeev 1
Artemenko, Sergei 377
Austrin, Per 13
Avigad, Lidor 389
- Ba, Khanh Do 26
Bansal, Nikhil 38
Bartal, Yair 50
Ben-Sasson, Eli 400, 412
Berman, Piotr 62
Bhalgat, Anand 75, 87
Bhatnagar, Nayantara 424
Bogdanov, Andrej 424
Braverman, Mark 13
Brody, Joshua 436
- Carnes, Tim 99
Carroll, Douglas E. 50
Chakaravarthy, Venkatesan T. 111
Chakrabarti, Amit 448
Chakrabarty, Deeparnab 75, 87
Chalermsook, Parinya 123
Cheung, Maurice 135
Childs, Andrew M. 365
Chlamtác, Eden 13
Cohen, Nachshon 147
Cooper, Colin 351
Crouch, Michael S. 158
- Dachman-Soled, Dana 460
Dani, Varsha 472
De, Anindya 483
Dellamonica, Domingos 495
Demaine, Erik D. 62
Dinur, Irit 507
Dragan, Feodor F. 171
Draief, Moez 351
Drucker, Andrew 519
Dubey, Chandan 184
Dumitrescu, Adrian 194
- Edelman, Alan 530
- Fekete, Sándor P. 206
Feldman, Moran 218
Fischer, Eldar 542
Friedrich, Tobias 555
- Galanis, Andreas 567
Ge, Qi 567
Ge, Rong 1
Gørtz, Inge Li 230
Goldreich, Oded 389, 579
Grigorescu, Elena 400
Guruswami, Venkatesan 593
- Hassidim, Avinatan 530
Håstad, Johan 242
Hemenway, Brett 605
Holenstein, Thomas 184
Huang, Zhiyi 254, 616
- Indyk, Piotr 26
- Jiang, Minghui 194
- Kalyanasundaram, Subrahmanyam 495
Kamphans, Tom 206
Kane, Daniel 628
Kannan, Sampath 616
Kapralov, Michael 266
Karpinski, Marek 277
Kaufman, Tali 507, 579
Khandekar, Rohit 289
Khani, M. Reza 302
Khanna, Sanjeev 75, 87
Köhler, Ekkehard 171
Kondapally, Ranganath 448
Kortsarz, Guy 289
Krishnaswamy, Ravishankar 38
Kröller, Alexander 206
Kumar, Amit 111
- Levine, Lionel 555
Liu, Yi-Kai 365
Louis, Anand 315
Lovett, Shachar 640
- Maatouk, Ghid 400
Martin, Daniel 495

- McGregor, Andrew 158
 Meka, Raghu 628
 Meyerson, Adam 50
 Miracle, Sarah 652
 Mitchell, Joseph S.B. 206
 Moore, Cristopher 472
 Mossel, Elchanan 424

 Nagarajan, Viswanath 230
 Naor, Joseph (Seffi) 218
 Neiman, Ofer 50
 Nelson, Jelani 628
 Nguyen, Huy N. 530
 Nutov, Zeev 147, 289

 Onak, Krzysztof 530
 Ostrovsky, Rafail 605

 Pach, János 194
 Pandit, Vinayaka 111
 Panigrahy, Rina 266

 Raghavendra, Prasad 315
 Randall, Dana 652
 Rödl, Vojtěch 495
 Ron, Dana 664, 676
 Roy, Sambuddha 111
 Rozenberg, Eyal 542
 Rubinfeld, Ronitt 664

 Sabharwal, Yogish 111
 Sachdeva, Sushant 327
 Safra, Muli 664
 Saha, Barna 38
 Saket, Rishi 327

 Salavatipour, Mohammad R. 302
 Schmidt, Christiane 206
 Schudy, Warren 277
 Schwartz, Roy 218
 Servedio, Rocco A. 460
 Shaltiel, Ronen 377
 Shapira, Asaf 495
 Shmoys, David B. 99, 135
 Shpilka, Amir 400
 Snir, Sagi 339
 Srinivasan, Srikanth 640
 Štefankovič, Daniel 567
 Strauss, Martin J. 605
 Streib, Amanda Pascoe 652
 Sudan, Madhu 400, 412

 Tetali, Prasad 315
 Tsur, Gilad 676

 Vempala, Santosh 315
 Vigoda, Eric 567

 Wang, Carol 593
 Wang, Lei 254
 Watson, Thomas 483, 688
 Weinstein, Omri 664
 Woodruff, David P. 436
 Wootters, Mary 605

 Yang, Linji 567
 Yuster, Raphael 339

 Zadimoghaddam, Morteza 62
 Zhou, Yuan 254