Anupam Gupta Klaus Jansen
José Rolim Rocco Servedio (Eds.)

Approximation, Randomization, and Combinatorial Optimization

Algorithms and Techniques

LNCS 7408

15th International Workshop, APPROX 2012
and 16th International Workshop, RANDOM 2012
Cambridge, MA, USA, August 2012, Proceedings

Springer

Lecture Notes in Computer Science 7408

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

Anupam Gupta Klaus Jansen
José Rolim Rocco Servedio (Eds.)

Approximation, Randomization, and Combinatorial Optimization

Algorithms and Techniques

15th International Workshop, APPROX 2012
and 16th International Workshop, RANDOM 2012
Cambridge, MA, USA, August 15-17, 2012
Proceedings

Springer

Volume Editors

Anupam Gupta
Carnegie Mellon University
Department of Computer Science
7203 Gates Building, Pittsburgh, PA 15213, USA
E-mail: anupamg@cs.cmu.edu

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

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

Rocco Servedio
Columbia University
Department of Computer Science
Foundation School of Engineering and Applied Science
1214 Amsterdam Avenue, 10027-7003 New York, NY, USA
E-mail: rocco@cs.columbia.edu

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-32511-3 e-ISBN 978-3-642-32512-0
DOI 10.1007/978-3-642-32512-0
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012943609

CR Subject Classification (1998): F.2.2, G.2.2, G.2.1, F.1.2, G.1.0, G.1.2, G.1.6, G.3

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

© Springer-Verlag Berlin Heidelberg 2012
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 15th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems(APPROX 2012) and the 16th International Workshop on Randomization and Computation (RANDOM 2012), which took place concurrently at the Massachusetts Institute of Technology, USA, during August 15–17, 2012.

APPROX focuses on algorithmic and complexity issues surrounding the development of efficient approximate solutions to computationally difficult problems, and was the 15th in the series after Aalborg (1998), Berkeley (1999), Saarbrücken (2000), Berkeley (2001), Rome (2002), Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), and Princeton (2011). RANDOM is concerned with applications of randomness to computational and combinatorial problems, and was the 16th workshop in the series following Bologna (1997), Barcelona (1998), Berkeley (1999), Geneva (2000), Berkeley (2001), Harvard (2002), Princeton (2003), Cambridge (2004), Berkeley (2005), Barcelona (2006), Princeton (2007), Boston (2008), Berkeley (2009), Barcelona (2010), Princeton (2011).

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 28 contributed papers, selected by the APPROX Program Committee out of 70 submissions, and 28 contributed papers, selected by the RANDOM Program Committee out of 67 submissions.

We would like to thank all of the authors who submitted papers, the invited speakers, the members of the Program Committees, and the external reviewers.

We gratefully acknowledge the support from the Department of Computer Science at the Carnegie Mellon University, USA, the Department of Computer Science at the Columbia University, the Institute of Computer Science of the Christian-Albrechts-Universität zu Kiel, and the Department of Computer Science of the University of Geneva.

We warmly thank Joanne Talbot and Ronitt Rubenfeld from the MIT Computer Science and Artificial Intelligence Laboratory for all the help and support. We also would like to thank Microsoft Research - New England for the partial sponsorship of this conference.

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

August 2012

Anupam Gupta
Rocco Servedio
Klaus Jansen
José D.P. Rolim

Organization

Program Committees

APPROX 2012

Alexandr Andoni	Microsoft Research Silicon Valley, USA
Yossi Azar	Blavatnik School of Computer Science, Israel
Shuchi Chawla	University of Wisconsin, Madison
Anupam Gupta	Carnegie Mellon University (Chair), USA
Sariel Har-Peled	University of Illinois, USA
Jochen Koenemann	University of Waterloo, Canada
Amit Kumar	Indian Institute of Technology, India
Lap Chi Lau	Chinese University of Hong Kong, Hong Kong
Konstantin Makarychev	Microsoft Research, USA
Monaldo Mastrolilli	IDSIA, Switzerland
Dana Moshkovitz	Massachusetts Institute of Technology, USA
Rene Sitters	Vrije Universiteit, The Netherlands
David Steurer	Microsoft Research, USA
Kunal Talwar	Microsoft Research Silicon Valley, USA
Jan Vondrak	IBM Almadan Research Center, USA
Lisa Zhang	Bell Laboratories, USA

RANDOM 2012

Eli Ben-Sasson	Technion-Israel Institute of Technology, Israel
Andrej Bogdanov	Chinese University of Hong Kong, Hong Kong
Mark Braverman	University of Toronto, Canada
Colin Cooper	King's College London, UK
Tobias Friedrich	Saarland University, Germany
Tali Kaufman	Bar Ilan University, Israel
Raghu Meka	Institute for Advanced Study, Princeton, USA
Jelani Nelson	Princeton University, USA
Ilan Newman	University of Haifa, Israel
Ryan O'Donnell	Carnegie Mellon University,USA
Konstantinos Panagiotou	Max-Planck-Institut für informatik, Germany
Prasad Raghavendra	Georgia Institute of Technology, USA
Atri Rudra	University of Buffalo, USA
Rocco Servedio (Chair)	Columbia University, USA
Alistair Sinclair	University of California, USA
Emanuele Viola	Northeastern University, USA

External Reviewers

Mohammed Abdullah
Dimitris Achlioptas
Noga Alon
Hyung-Chan An
Per Austrin
Nikhil Bansal
Siddharth Barman
Avi Ben-Aroya
Arnab Bhattacharyya
Eric Blais
Andrej Bogdanov
Graham Brightwell
Karl Bringmann
Andrei Bulatov
K. Chandrasekaran
Deeparnab Chakrabarty
Tanmoy Chakraborty
Parinya Chalermosook
Ho-Leung Chan
Chandra Chekuri
Joseph Cheriyan
Eden Chlamtac
Sung-Soon Choi
Giorgos Christodoulou
Edith Cohen
Gil Cohen
Ilan Cohen
Marek Cygan
Artur Czumaj
Anindya De
Ilias Diakonikolas
Irit Dinur
Yevgeniy Dodis
Benjamin Doerr
Zeev Dvir
Jeff Edmonds
Amir Epstein
Dan Feldman
Jon Feldman
Cristina Fernandes
David Fernandez-Baca
Eldar Fischer
Fedor Fomin
Zachary Friggstad
Stanley P. Y. Fung
Ariel Gabizon
Iftah Gamzu
Naveen Garg
Luisa Gargano
William Gasarch
Konstantinos Georgiou
Shayan Oveis Gharan
Parikshit Gopalan
Venkat Guruswami
Iftach Haitner
Tobias Harks
Prahladh Harsha
Frank Hellweg
Sungjin Im
Piotr Indyk
Rahul Jain
Klaus Jansen
Mark Jerrum
Lukasz Jez
Michael Kapralov
Mark Keil
Rohit Khandekar
Shiva Kintali
Hartmut Klauck
Lasse Kliemann
Ranganath Kondapally
Michael Krivelevich
Amit Kumar
Tsz Chiu Kwok
Oded Lachish
Huy Le Nguyen
Troy Lee
Johannes Lengler
Stefano Leonardi
Reut Levi
Xin Li
Henry Lin
Shachar Lovett
Mohammad Mahdian
Yury Makarychev
Tal Malkin
Rajsekar Manokaran
Bodo Manthey
Arie Matsliah
Pierre McKenzie
Moti Medina
Or Meir
Julian Mestre
Joel Miller
Carsten Moldenhauer
Ben Moseley
Viswanath Nagarajan
Meghana Nasre
Amir Nayyeri
Chrystopher Nehaniv
Zeev Nutov
Krzysztof Onak
Shayan Oveis Gharan
Renato Paes-Leme
Denis Pankratov
Britta Peis
Eric Price
Prasad Raghavendra
Rajmohan Rajaraman
R. Ravi
Ben Recht
Oded Regev
Dana Ron
Benjamin Rossman
Guy Rothblum
Barna Saha
Rishi Saket
Alex Samorodnitsky
Shubhangi Saraf
Guido Schaefer
Gil Segev
Sandeep Sen
Ronen Shaltiel
Yaoyun Shi
Amir Shpilka
Brad Shutters
Yiannis Siantos
Mohit Singh
Allan Sly

Reto Spohel
Aravind Srinivasan
Rob van Stee
Daniel Stefankovic
Martin Strauss
He Sun
Tami Tamir
Li-Yang Tan
Madhur Tulsiani
Jonathan Ullman
Seeun Umboh
Sergei Vassilvitskii
Laszlo Vegh
Jose' Verschae
A. Vijayaraghavan
Pascal Vontobel
Andrew Wan
Carol Wang
David Williamson
Karl Wimmer
Carola Winzen
Ning Xie
Dachuan Xu
Li Yan
Qiqi Yan
Grigory Yaroslavtsev
Yuichi Yoshida
Qin Zhang
Yuan Zhou
David Zuckerman

Table of Contents

Contributed Talks of APPROX

Contributed Talks of RANDOM

A New Point of NP-Hardness for 2-to-1 Label Cover

Per Austrin[1,⋆], Ryan O'Donnell[2,⋆⋆], and John Wright[2]

[1] Department of Computer Science, University of Toronto
[2] Department of Computer Science, Carnegie Mellon University

Abstract. We show that given a satisfiable instance of the 2-to-1 Label Cover problem, it is NP-hard to find a $(\frac{23}{24} + \epsilon)$-satisfying assignment.

1 Introduction

Over the past decade, a significant amount of progress has been made in the field of hardness of approximation via results based on the conjectured hardness of certain forms of the Label Cover problem. The *Unique Games Conjecture* (UGC) of Khot [15] states that it is NP-hard to distinguish between nearly satisfiable and almost completely unsatisfiable instances of *Unique*, or *1-to-1*, Label Cover. Using the UGC as a starting point, we now have optimal inapproximability results for Vertex Cover [17], Max-Cut [16], and many other basic constraint satisfaction problems (CSP). Indeed, assuming the UGC we have essentially optimal inapproximability results for *all* CSPs [21]. In short, modulo the understanding of Unique Label Cover itself, we have an excellent understanding of the (in-)approximability of a wide range of problems.

Where the UGC's explanatory powers falter is in pinning down the approximability of *satisfiable* CSPs. This means the task of finding a good assignment to a CSP when guaranteed that the CSP is fully satisfiable. For example, we know from the work of Håstad [13] that given a fully satisfiable 3Sat instance, it is NP-hard to satisfy $\frac{7}{8} + \epsilon$ of the clauses for any $\epsilon > 0$. However given a fully satisfiable 1-to-1 Label Cover instance, it is completely trivial to find a fully satisfying assignment. Thus the UGC can not be used as the starting point for hardness results for satisfiable CSPs. Because of this, Khot additionally posed his d-to-1 Conjectures:

Conjecture 1.1 ([15]). For every integer $d \geq 2$ and $\epsilon > 0$, there is a label set size q such that it is NP-hard to $(1, \epsilon)$-decide the d-to-1 Label Cover problem.

Here by (c, s)-deciding a CSP we mean the task of determining whether an instance is at least c-satisfiable or less than s-satisfiable. It is well known (from the Parallel Repetition Theorem [6, 22]) that the conjecture is true if d is allowed to depend on ϵ. The strength of this conjecture, therefore, is that it is stated for each fixed d greater than 1.

⋆ Funded by NSERC.
⋆⋆ Supported by NSF grants CCF-0747250 and CCF-0915893, and by a Sloan fellowship.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 1–12, 2012.
© Springer-Verlag Berlin Heidelberg 2012

The d-to-1 Conjectures have been used to resolve the approximability of several basic "satisfiable CSP" problems. The first result along these lines was due to Dinur, Mossel, and Regev [5] who showed that the 2-to-1 Conjecture implies that it is NP-hard to C-color a 4-colorable graph for any constant C. (They also showed hardness for 3-colorable graphs via another Unique Games variant.) O'Donnell and Wu [20] showed that assuming the d-to-1 Conjecture for any fixed d implies that it is NP-hard to $(1, \frac{5}{8} + \epsilon)$-approximate instances a certain 3-bit predicate — the "Not Two" predicate. This is an optimal result among all 3-bit predicates, since Zwick [25] showed that every satisfiable 3-bit CSP instance can be efficiently $\frac{5}{8}$-approximated. In another example, Guruswami and Sinop [12] have shown that the 2-to-1 Conjecture implies that given a q-colorable graph, it is NP-hard to find a q-coloring in which less than a $(\frac{1}{q} - O(\frac{\ln q}{q^2}))$ fraction of the edges are monochromatic. This result would be tight up to the $O(\cdot)$ by an algorithm of Frieze and Jerrum [7]. It is therefore clear that settling the d-to-1 Conjectures, especially in the most basic case of $d = 2$, is an important open problem.

Regarding the hardness of the 2-to-1 Label Cover problem, the only evidence we have is a family of integrality gaps for the canonical SDP relaxation of the problem, in [9]. Regarding algorithms for the problem, an important recent line of work beginning in [1] (see also [4, 11, 23]) has sought subexponential-time algorithms for Unique Label Cover and related problems. In particular, Steurer [23] has shown that for any constant $\beta > 0$ and label set size, there is an $\exp(O(n^\beta))$-time algorithm which, given a satisfiable 2-to-1 Label Cover instance, finds an assignment satisfying an $\exp(-O(1/\beta^2))$-fraction of the constraints. E.g., there is a $2^{O(n^{.001})}$-time algorithm which $(1, s_0)$-approximates 2-to-1 Label Cover, where $s_0 > 0$ is a certain universal constant.

In light of this, it is interesting not only to seek NP-hardness results for certain approximation thresholds, but to additionally seek evidence that *nearly full exponential time* is required for these thresholds. This can done by assuming the Exponential Time Hypothesis (ETH) [14] and by reducing from the Moshkovitz–Raz Theorem [18], which shows a near-linear size reduction from 3Sat to the standard Label Cover problem with subconstant soundness. In this work, we show reductions from 3Sat to the problem of $(1, s + \epsilon)$-approximating several CSPs, for certain values of s and for all $\epsilon > 0$. In fact, though we omit it in our theorem statements, it can be checked that all of the reductions in this paper are quasilinear in size for $\epsilon = \epsilon(n) = \Theta\left(\frac{1}{(\log \log n)^\beta}\right)$, for some $\beta > 0$.

1.1 Our Results

In this paper, we focus on proving NP-hardness for the 2-to-1 Label Cover problem. To the best of our knowledge, no explicit NP-hardness factor has previously been stated in the literature. However it is "folklore" that one can obtain an explicit one for label set sizes 3 & 6 by performing the "constraint-variable" reduction on an NP-hardness result for 3-coloring (more precisely, Max-3-Colorable-Subgraph). The best known hardness for 3-coloring is due to Guruswami and Sinop

[12], who showed a factor $\frac{32}{33}$-hardness via a somewhat involved gadget reduction from the 3-query adaptive PCP result of [10]. This yields NP-hardness of $(1, \frac{65}{66} + \epsilon)$-approximating 2-to-1 Label Cover with label set sizes 3 & 6. It is not known how to take advantage of larger label set sizes. On the other hand, for label set sizes 2 & 4 it is known that satisfying 2-to-1 Label Cover instances can be found in polynomial time.

The main result of our paper gives an improved hardness result:

Theorem 1.2. *For all $\epsilon > 0$, $(1, \frac{23}{24}+\epsilon)$-deciding the 2-to-1 Label Cover problem with label set sizes 3 & 6 is NP-hard.*

By duplicating labels, this result also holds for label set sizes $3k$ & $6k$ for any $k \in \mathbb{N}^+$.

Let us describe the high-level idea behind our result. The folklore constraint-variable reduction from 3-coloring to 2-to-1 Label Cover would work just as well if we started from "3-coloring with literals" instead. By this we mean the CSP with domain $\mathbb{Z}_3$ and constraints of the form "$v_i - v_j \neq c \pmod 3$". Starting from this CSP — which we call $\mathsf{2NLin}(\mathbb{Z}_3)$ — has two benefits: first, it is at least as hard as 3-coloring and hence could yield a stronger hardness result; second, it is a bit more "symmetrical" for the purposes of designing reductions. We obtain the following hardness result for $\mathsf{2NLin}(\mathbb{Z}_3)$.

Theorem 1.3. *For all $\epsilon > 0$, it is NP-hard to $(1, \frac{11}{12} + \epsilon)$-decide the 2NLin problem.*

As 3-coloring is a special case of $\mathsf{2NLin}(\mathbb{Z}_3)$, [12] also shows that $(1, \frac{32}{33} + \epsilon)$-deciding 2NLin is NP-hard for all $\epsilon > 0$, and to our knowledge this was previously the only hardness known for $\mathsf{2NLin}(\mathbb{Z}_3)$. The best current algorithm achieves an approximation ratio of 0.836 (and does not need the instance to be satisfiable) [8]. To prove Theorem 1.3, we proceed by designing an appropriate "function-in-the-middle" dictator test, as in the recent framework of [19]. Although the [19] framework gives a direct translation of certain types of function-in-the-middle tests into hardness results, we cannot employ it in a black-box fashion. Among other reasons, [19] assumes that the test has "built-in noise", but we cannot afford this as we need our test to have perfect completeness.

Thus, we need a different proof to derive a hardness result from this function-in-the-middle test. We first were able to accomplish this by an analysis similar to the Fourier-based proof of $\mathsf{2Lin}(\mathbb{Z}_2)$ hardness given in Appendix F of [19]. Just as that proof "reveals" that the function-in-the-middle $\mathsf{2Lin}(\mathbb{Z}_2)$ test can be equivalently thought of as Håstad's $\mathsf{3Lin}(\mathbb{Z}_2)$ test composed with the $\mathsf{3Lin}(\mathbb{Z}_2)$-to-$\mathsf{2Lin}(\mathbb{Z}_2)$ gadget of [24], our proof for the $\mathsf{2NLin}(\mathbb{Z}_3)$ function-in-the-middle test revealed it to be the composition of a function test for a certain four-variable CSP with a gadget. We have called the particular four-variable CSP 4-Not-All-There, or 4NAT for short. Because it is a 4-CSP, we are able to prove the following NP-hardness of approximation result for it using a classic, Håstad-style Fourier-analytic proof.

Theorem 1.4. *For all $\epsilon > 0$, it is* NP*-hard to* $(1, \frac{2}{3}+\epsilon)$*-decide the* 4NAT *problem.*

Thus, the final form in which we present our Theorem 1.2 is as a reduction from Label-Cover to 4NAT using a function test (yielding Theorem 1.4), followed by a 4NAT-to-2NLin($\mathbb{Z}_3$) gadget (yielding Theorem 1.3), followed by the constraint-variable reduction to 2-to-1 Label Cover. Indeed, all of the technology needed to carry out this proof was in place for over a decade, but without the function-in-the-middle framework of [19] it seems that pinpointing the 4NAT predicate as a good starting point would have been unlikely.

1.2 Organization

We leave to Section 2 most of the definitions, including those of the CSPs we use. The heart of the paper is in Section 3, where we give both the 2NLin($\mathbb{Z}_3$) and 4NAT function tests, explain how one is derived from the other, and then perform the Fourier analysis for the 4NAT test. In Section 4 we discuss the NP-hardness result for 4NAT. Due to space considerations, several proofs are omitted but can be found in the full version of the paper [3].

2 Preliminaries

We primarily work with strings $x \in \mathbb{Z}_3^K$ for some integer K. We write x_i to denote the ith coordinate of x. Oftentimes, our strings $y \in \mathbb{Z}_3^{dK}$ are "blocked" into K "blocks" of size d. In this case, we write $y[i] \in \mathbb{Z}_3^d$ for the ith block of y, and $(y[i])_j \in \mathbb{Z}_3$ for the jth coordinate of this block. Define the function $\pi : [dK] \to [K]$ such that $\pi(k) = i$ if k falls in the ith block of size d (e.g., $\pi(k) = 1$ for $1 \le k \le d$, $\pi(k) = 2$ for $d+1 \le k \le 2d$, and so on).

2.1 Definitions of Problems

An instance $\mathcal{I}$ of a *constraint satisfaction problem* (CSP) is a set of variables V, a set of labels D, and a weighted list of constraints on these variables. We assume that the weights of the constraints are nonegative and sum to 1. The weights therefore induce a probability distribution on the constraints. Given an assignment to the variables $f : V \to D$, the *value* of f is the probability that f satisfies a constraint drawn from this probability distribution. The *optimum* of $\mathcal{I}$ is the highest value of any assignment. We say that an $\mathcal{I}$ is *s-satisfiable* if its optimum is at least s. If it is 1-satisfiable we simply call it satisfiable.

We define a CSP $\mathcal{P}$ to be a set of CSP instances. Typically, these instances will have similar constraints. We will study the problem of (c, s)*-deciding* $\mathcal{P}$. This is the problem of determining whether an instance of $\mathcal{P}$ is at least c-satisfiable or less than s-satisfiable. Related is the problem of (c, s)*-approximating* $\mathcal{P}$, in which one is given a c-satisfiable instance of $\mathcal{P}$ and asked to find an assignment of value at least s. It is easy to see that (c, s)-deciding $\mathcal{P}$ is at least as easy as (c, s)-approximating $\mathcal{P}$. Thus, as all our hardness results are for (c, s)-deciding CSPs, we also prove hardness for (c, s)-approximating these CSPs.

We now state the three CSPs that are the focus of our paper.

2-NLin($\mathbb{Z}_3$): In this CSP the label set is $\mathbb{Z}_3$ and the constraints are of the form

$$v_i - v_j \neq a \pmod 3, \quad a \in \mathbb{Z}_3.$$

The special case when each RHS is 0 is the 3-coloring problem. We often drop the ($\mathbb{Z}_3$) from this notation and simply write 2NLin. The reader may think of the 'N' in 2NLin($\mathbb{Z}_3$) as standing for 'N'on-linear, although we prefer to think of it as standing for 'N'early-linear. The reason is that when generalizing to moduli $q > 3$, the techniques in this paper generalize to constraints of the form "$v_i - v_j \pmod q \in \{a, a+1\}$" rather than "$v_i - v_j \neq a \pmod q$". For the ternary version of this constraint, "$v_i - v_j + v_k \pmod q \in \{a, a+1\}$", it is folklore[1] that a simple modification of Håstad's work [13] yields NP-hardness of $(1, \frac{2}{q})$-approximation.

4-Not-All-There: For the 4-Not-All-There problem, denoted 4NAT, we define 4NAT : $\mathbb{Z}_3^4 \to \{0,1\}$ to have output 1 if and only if at least one of the elements of $\mathbb{Z}_3$ is not present among the four inputs. The 4NAT CSP has label set $D = \mathbb{Z}_3$ and constraints of the form $\mathsf{4NAT}(v_1 + k_1, v_2 + k_2, v_3 + k_3, v_4 + k_4) = 1$, where the k_i's are constants in $\mathbb{Z}_3$.

We additionally define the "Two Pairs" predicate TwoPair : $\mathbb{Z}_3^4 \to \{0,1\}$, which has output 1 if and only if its input contains two distinct elements of $\mathbb{Z}_3$, each appearing twice. Note that an input which satisfies TwoPair also satisfies 4NAT.

d-*to-1 Label Cover:* An instance of the d-to-1 Label Cover problem is a bipartite graph $G = (U \cup V, E)$, a label set size K, and a d-to-1 map $\pi_e : [dK] \to [K]$ for each edge $e \in E$. The elements of U are labeled from the set $[K]$, and the elements of V are labeled from the set $[dK]$. A labeling $f : U \cup V \to [dK]$ satisfies an edge $e = (u, v)$ if $\pi_e(f(v)) = f(u)$. Of particular interest is the $d = 2$ case, i.e., 2-to-1 Label Cover.

Label Cover serves as the starting point for most NP-hardness of approximation results. We use the following theorem of Moshkovitz and Raz:

Theorem 2.1 ([18]). *For any $\epsilon = \epsilon(n) \geq n^{-o(1)}$ there exists $K, d \leq 2^{\mathrm{poly}(1/\epsilon)}$ such that the problem of deciding a 3Sat instance of size n can be Karp-reduced in $\mathrm{poly}(n)$ time to the problem of $(1, \epsilon)$-deciding d-to-1 Label Cover instance of size $n^{1+o(1)}$ with label set size K.*

2.2 Gadgets

A typical way of relating two separate CSPs is by constructing a *gadget reduction* which translates from one to the other. A gadget reduction from CSP_1 to CSP_2 is one which maps any CSP_1 constraint into a weighted set of CSP_2 constraints. The CSP_2 constraints are over the same set of variables as the CSP_1 constraint, plus some new, auxiliary variables (these auxiliary variables are not shared between

[1] Venkatesan Guruswami, Subhash Khot personal communications.

constraints of CSP_1). We require that for every assignment which satisfies the CSP_1 constraint, there is a way to label the auxiliary variables to fully satisfy the CSP_2 constraints. Furthermore, there is some parameter $0 < \gamma < 1$ such that for every assignment which does not satisfy the CSP_1 constraint, the optimum labeling to the auxiliary variables will satisfy exactly γ fraction of the CSP_2 constraints. Such a gadget reduction we call a *γ-gadget-reduction* from CSP_1 to CSP_2. The following proposition is well-known:

Proposition 2.2. *Suppose it is* NP*-hard to (c, s)-decide CSP_1. If there exists a γ-gadget-reduction from CSP_1 to CSP_2, then it is* NP*-hard to $(c + (1 - c)\gamma, s + (1 - s)\gamma)$-decide CSP_2.*

We note that the notation γ-gadget-reduction is similar to a piece of notation employed by [24], but the two have different (though related) definitions.

2.3 Fourier Analysis on $\mathbb{Z}_3$

Let $\omega = e^{2\pi i/3}$ and set $U_3 = \{\omega^0, \omega^1, \omega^2\}$. For $\alpha \in \mathbb{Z}_3^n$, consider the Fourier character $\chi_\alpha : \mathbb{Z}_3^n \to U_3$ defined as $\chi_\alpha(x) = \omega^{\alpha \cdot x}$. Then it is easy to see that $\mathbf{E}[\chi_\alpha(\boldsymbol{x})\overline{\chi_\beta(\boldsymbol{x})}] = \mathbf{1}[\alpha = \beta]$, where here and throughout $\boldsymbol{x}$ has the uniform probability distribution on $\mathbb{Z}_3^n$ unless otherwise specified.. As a result, the Fourier characters form an orthonormal basis for the set of functions $f : \mathbb{Z}_3^n \to U_3$ under the inner product $\langle f, g \rangle = \mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{x})]$; i.e.,

$$f = \sum_{\alpha \in \mathbb{Z}_3^n} \hat{f}(\alpha)\chi_\alpha,$$

where the $\hat{f}(\alpha)$'s are complex numbers defined as $\hat{f}(\alpha) = \mathbf{E}[f(\boldsymbol{x})\overline{\chi_\alpha(\boldsymbol{x})}]$. For $\alpha \in \mathbb{Z}_3^n$, we use the notation $|\alpha|$ to denote $\sum \alpha_i$ and $\#\alpha$ to denote the number of nonzero coordinates in α. When d is clear from context and $\alpha \in \mathbb{Z}_3^{dK}$, define $\pi_3(\alpha) \in \mathbb{Z}_3^K$ so that $(\pi_3(\alpha))_i \equiv |\alpha[i]| \pmod 3$ (recall the notation $\alpha[i]$ from the beginning of this section).

We have Parseval's identity: for every $f : \mathbb{Z}_3^n \to U_3$ it holds that $\sum_{\alpha \in \mathbb{Z}_3^n} |\hat{f}(\alpha)|^2 = 1$. Note that this implies that $|\hat{f}(\alpha)| \leq 1$ for all α, as otherwise $\hat{f}(\alpha)^2$ would be greater than 1. A function $f : \mathbb{Z}_3^n \to \mathbb{Z}_3$ is said to be *folded* if for every $x \in \mathbb{Z}_3^n$ and $c \in \mathbb{Z}_3$, it holds that $f(x + c) = f(x) + c$, where $(x + c)_i = x_i + c$.

Proposition 2.3. *Let $f : \mathbb{Z}_3^n \to U_3$ be folded. Then $\hat{f}(\alpha) \neq 0 \Rightarrow |\alpha| \equiv 1 \pmod 3$.*

3 2-to-1 Hardness

In this section, we give our hardness result for 2-to-1 Label Cover, following the proof outline described at the end of Section 1.1.

Theorem 3.1 (Theorem 1.2 (restated)). *For all $\epsilon > 0$, it is* NP*-hard to $(1, \frac{23}{24} + \epsilon)$-decide the 2-to-1 Label Cover problem.*

First, we state a pair of simple gadget reductions:

Lemma 3.2. *There is a 3/4-gadget-reduction from 4NAT to 2NLin.*

Lemma 3.3. *There is a 1/2-gadget-reduction from 2NLin to 2-to-1.*

Together with Proposition 2.2, these imply the following corollary:

Corollary 3.4. *There is a 7/8-gadget-reduction from 4NAT to 2-to-1. Thus, if it is* NP*-hard to (c, s)-decide the 4NAT problem, then it is* NP*-hard to $((7 + c)/8, (7 + s)/8)$-decide the 2-to-1 Label Cover problem.*

The gadget reduction from 4NAT to 2NLin relies on the simple fact that if $a, b, c, d \in \mathbb{Z}_3$ satisfy the 4NAT predicate, then there is some element of $\mathbb{Z}_3$ that none of them equal.

The reduction from 2NLin to 2-to-1 Label Cover is the well-known constraint-variable reduction, and uses the fact that in the equation $v_i - v_j \neq a \pmod 3$, for any assignment to v_j there are two valid assignments to v_i, and vice versa.

3.1 A Pair of Tests

Now that we have shown that 2NLin hardness results translate into 2-to-1 Label Cover hardness results, we present our 2NLin function test. Even though we don't directly use it, it helps explain how we were led to consider the 4NAT CSP. Furthermore, the Fourier analysis that we eventually use for the 4NAT Test could instead be performed directly on the 2NLin Test without any direct reference to the 4NAT predicate. The test is:

2NLin Test

Given folded functions $f : \mathbb{Z}_3^K \to \mathbb{Z}_3$, $g, h : \mathbb{Z}_3^{dK} \to \mathbb{Z}_3$:

- Let $\boldsymbol{x} \in \mathbb{Z}_3^K$ and $\boldsymbol{y} \in \mathbb{Z}_3^{dK}$ be independent and uniformly random.
- For each $i \in [K], j \in [d]$, select $(\boldsymbol{z}[i])_j$ independently and uniformly from the elements of $\mathbb{Z}_3 \setminus \{\boldsymbol{x}_i, (\boldsymbol{y}[i])_j\}$.
- With probability $\frac{1}{4}$, test $f(\boldsymbol{x}) \neq h(\boldsymbol{z})$; with probability $\frac{3}{4}$, test $g(\boldsymbol{y}) \neq h(\boldsymbol{z})$.

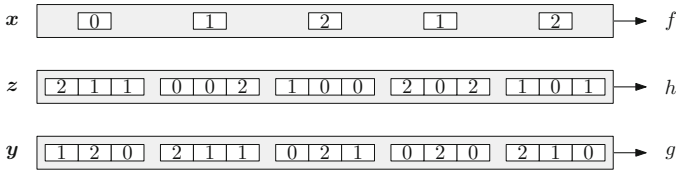

Fig. 1. An illustration of the 2NLin test distribution; $d = 3$, $K = 5$

Above is an illustration of the test. We remark that for any given block i, $z[i]$ determines x_i (with very high probability), because as soon as $z[i]$ contains two distinct elements of $\mathbb{Z}_3$, x_i must be the third element of $\mathbb{Z}_3$. Notice also that in every column of indices, the input to h always differs from the inputs to both f and g. Thus, "matching dictator" assignments pass the test with probability 1. (This is the case in which $f(x) = x_i$ and $g(y) = (y[i])_j$ for some $i \in [K]$, $j \in [d]$.) On the other hand, if f and g are "nonmatching dictators", then they succeed with only $\frac{11}{12}$ probability. This turns out to be essentially optimal among functions f and g without "matching influential coordinates/blocks". We will obtain the following theorem:

Theorem 3.5 (Theorem 1.3 restated). *For all $\epsilon > 0$, it is* NP*-hard to $(1, \frac{11}{12} + \epsilon)$-decide the 2NLin problem.*

Before proving this, let us further discuss the 2NLin test. Given $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{z}$ from the 2NLin test, consider the following method of generating two additional strings $\boldsymbol{y'}, \boldsymbol{y''} \in \mathbb{Z}_3^{dK}$ which represent h's "uncertainty" about $\boldsymbol{y}$. For $j \in [d]$, if $\boldsymbol{x}_i = (\boldsymbol{y}[i])_j$, then set both $(\boldsymbol{y'}[i])_j$ and $(\boldsymbol{y''}[i])_j$ to the lone element of $\mathbb{Z}_3 \setminus \{\boldsymbol{x}_i, (\boldsymbol{z}[i])_j\}$. Otherwise, set one of $(\boldsymbol{y'}[i])_j$ or $(\boldsymbol{y''}[i])_j$ to $\boldsymbol{x}_i$, and the other one to $(\boldsymbol{y}[i])_j$. It can be checked that $\mathsf{TwoPair}(\boldsymbol{x}_i, (\boldsymbol{y}[i])_j, (\boldsymbol{y'}[i])_j, (\boldsymbol{y''}[i])_j) = 1$, a more stringent requirement than satisfying 4NAT. In fact, the marginal distribution on these four variables is a uniformly random assignment that satisfies the TwoPair predicate. Conditioned on $\boldsymbol{x}$ and $\boldsymbol{z}$, the distribution on $\boldsymbol{y'}$ and $\boldsymbol{y''}$ is identical to the distribution on $\boldsymbol{y}$. To see this, first note that by construction, neither $(\boldsymbol{y'}[i])_j$ nor $(\boldsymbol{y''}[i])_j$ ever equals $(\boldsymbol{z}[i])_j$. Further, because these indices are distributed as uniformly random satisfying assignments to TwoPair, $\mathbf{Pr}[(\boldsymbol{y'}[i])_j = x_i] = \mathbf{Pr}[(\boldsymbol{y''}[i])_j = x_i] = \frac{1}{3}$, which matches the corresponding probability for $\boldsymbol{y}$. Thus, as $\boldsymbol{y}$, $\boldsymbol{y'}$, and $\boldsymbol{y''}$ are distributed identically, we may rewrite the test's success probability as:

$$\begin{aligned}\mathbf{Pr}[f, g, \text{ and } h \text{ pass the test}] &= \tfrac{1}{4}\mathbf{Pr}[f(\boldsymbol{x}) \neq h(\boldsymbol{z})] + \tfrac{3}{4}\mathbf{Pr}[g(\boldsymbol{y}) \neq h(\boldsymbol{z})] \\ &= \operatorname{avg}\left\{\begin{array}{r}\mathbf{Pr}[f(\boldsymbol{x}) \neq h(\boldsymbol{z})], \\ \mathbf{Pr}[g(\boldsymbol{y}) \neq h(\boldsymbol{z})], \\ \mathbf{Pr}[g(\boldsymbol{y'}) \neq h(\boldsymbol{z})], \\ \mathbf{Pr}[g(\boldsymbol{y''}) \neq h(\boldsymbol{z})]\end{array}\right\} \\ &\leq \frac{3}{4} + \frac{1}{4}\mathbf{E}[\mathsf{4NAT}(f(\boldsymbol{x}), g(\boldsymbol{y}), g(\boldsymbol{y'}), g(\boldsymbol{y''}))].\end{aligned}$$

This is because if 4NAT fails to hold on the tuple $(f(\boldsymbol{x}), g(\boldsymbol{y}), g(\boldsymbol{y'}), g(\boldsymbol{y''}))$, then $h(\boldsymbol{z})$ can disagree with at most 3 of them.

At this point, we have removed h from the test analysis and have uncovered what appears to be a hidden 4NAT test inside the 2NLin Test: simply generate four strings $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{y'}$, and $\boldsymbol{y''}$ as described earlier, and test whether $\mathsf{4NAT}(f(\boldsymbol{x}), g(\boldsymbol{y}), g(\boldsymbol{y'}), g(\boldsymbol{y''}))$. With some renaming of variables, this is exactly what our 4NAT Test does:

4NAT Test

Given folded functions $f : \mathbb{Z}_3^K \to \mathbb{Z}_3$, $g : \mathbb{Z}_3^{dK} \to \mathbb{Z}_3$:

- Let $\boldsymbol{x} \in \mathbb{Z}_3^K$ be uniformly random.
- Select $\boldsymbol{y}, \boldsymbol{z}, \boldsymbol{w}$ as follows: for each $i \in [K], j \in [d]$, select $((\boldsymbol{y}[i])_j, (\boldsymbol{z}[i])_j, (\boldsymbol{w}[i])_j)$ uniformly at random subjec to $\mathsf{TwoPair}(\boldsymbol{x}_i, (\boldsymbol{y}[i])_j, (\boldsymbol{z}[i])_j, (\boldsymbol{w}[i])_j)$ being satisfied.
- Test $\mathsf{4NAT}(f(\boldsymbol{x}), g(\boldsymbol{y}), g(\boldsymbol{z}), g(\boldsymbol{w}))$.

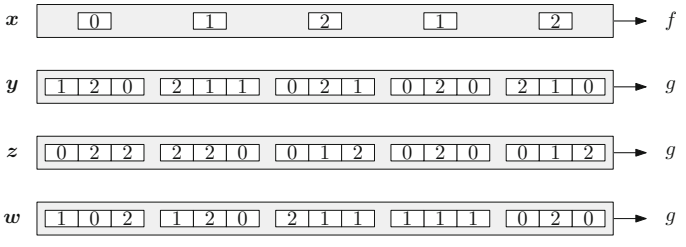

Fig. 2. An illustration of the 4NAT test distribution; $d = 3$, $K = 5$

Above is an illustration of this test. In this illustration, the strings $\boldsymbol{z}$ and $\boldsymbol{w}$ were derived from the strings in Figure 1 using the process detailed above for generating $\boldsymbol{y}'$ and $\boldsymbol{y}''$. Note that each column is missing one of the elements of $\mathbb{Z}_3$, and that each column satisfies the TwoPair predicate. Because satisfying TwoPair implies satisfying 4NAT, matching dictators pass this test with probability 1. On the other hand, it can be seen that nonmatching dictators pass the test with probability $\frac{2}{3}$. In the next section we show that this is optimal among functions f and g without "matching influential coordinates/blocks".

(As one additional remark, our 2NLin Test is basically the composition of the 4NAT Test with the gadget from Lemma 3.2. In this test, if we instead performed the $f(\boldsymbol{x}) \neq h(\boldsymbol{z})$ test with probability $\frac{1}{3}$ and the $g(\boldsymbol{y}) \neq h(\boldsymbol{z})$ test with probability $\frac{2}{3}$, then the resulting test would basically be the composition of a 3NLin test with a suitable 3NLin-to-2NLin gadget.)

3.2 Analysis of 4NAT Test

Let $\omega = e^{2\pi i/3}$, and set $U_3 = \{\omega^0, \omega^1, \omega^2\}$. In what follows, we identify f and g with the functions ω^f and ω^g, respectively, whose range is U_3 rather than $\mathbb{Z}_3$. Set $L = dK$. The remainder of this section is devoted to the proof of the following lemma:

Lemma 3.6. *Let $f : \mathbb{Z}_3^K \to U_3$ and $g : \mathbb{Z}_3^{dK} \to U_3$. Then*

$$\mathbf{E}[\mathit{4NAT}(f(\boldsymbol{x}), g(\boldsymbol{y}), g(\boldsymbol{z}), g(\boldsymbol{w}))] \leq \tfrac{2}{3} + \tfrac{2}{3} \sum_{\alpha \in \mathbb{Z}_3^L} |\hat{f}(\pi_3(\alpha))| \cdot |\hat{g}(\alpha)|^2 \cdot (1/2)^{\#\alpha}$$

The first step is to "arithmetize" the 4NAT predicate. One can check that

$$\begin{aligned}\mathsf{4NAT}(a_1,a_2,a_3,a_4) &= \\ &= \frac{5}{9} + \frac{1}{9}\sum_{i\neq j}\omega^{a_i}\overline{\omega}^{a_j} - \frac{1}{9}\sum_{i<j<k}\omega^{a_i}\omega^{a_j}\omega^{a_k} - \frac{1}{9}\sum_{i<j<k}\overline{\omega}^{a_i}\overline{\omega}^{a_j}\overline{\omega}^{a_k} \\ &= \frac{5}{9} + \frac{2}{9}\sum_{i<j}\Re[\omega^{a_i}\overline{\omega}^{a_j}] - \frac{2}{9}\sum_{i<j<k}\Re[\omega^{a_i}\omega^{a_j}\omega^{a_k}].\end{aligned}$$

Using the symmetry between $\boldsymbol{y}$, $\boldsymbol{z}$, and $\boldsymbol{w}$, we deduce

$$\begin{aligned}\mathbf{E}[\mathsf{4NAT}(f(\boldsymbol{x}),g(\boldsymbol{y}),g(\boldsymbol{z}),g(\boldsymbol{w}))] = \\ \tfrac{5}{9} + \tfrac{2}{3}\Re\,\mathbf{E}[f(\boldsymbol{x})\overline{g(\boldsymbol{y})}] + \tfrac{2}{3}\Re\,\mathbf{E}[g(\boldsymbol{y})\overline{g(\boldsymbol{z})}] \\ - \tfrac{2}{3}\Re\,\mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{y})g(\boldsymbol{z})] - \tfrac{2}{9}\Re\,\mathbf{E}[g(\boldsymbol{y})g(\boldsymbol{z})g(\boldsymbol{w})]. \quad (1)\end{aligned}$$

In the second term in the RHS of (1) we in fact have $\mathbf{E}[f(\boldsymbol{x})\overline{g(\boldsymbol{y})}] = 0$. This is because $\boldsymbol{x}$ and $\boldsymbol{y}$ are independent, and hence $\mathbf{E}[f(\boldsymbol{x})\overline{g(\boldsymbol{y})}] = \mathbf{E}[f(\boldsymbol{x})]\,\mathbf{E}[\overline{g(\boldsymbol{y})}] = 0 \cdot 0$ since f and g are folded. Regarding the third term of the RHS in (1), this also turns out to be 0 by virtue of g being folded.

Lemma 3.7. $\mathbf{E}[g(\boldsymbol{y})\overline{g(\boldsymbol{z})}] = 0$.

This can be proven using a Fourier-analytic argument; in the full version of the paper [3] we present an alternate combinatorial argument.

Equation (1) has now been reduced to

$$(1) = \tfrac{5}{9} - \tfrac{2}{3}\Re\,\mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{y})g(\boldsymbol{z})] - \tfrac{2}{9}\Re\,\mathbf{E}[g(\boldsymbol{y})g(\boldsymbol{z})g(\boldsymbol{w})]. \quad (2)$$

As $g(\boldsymbol{y})g(\boldsymbol{z})g(\boldsymbol{w})$ is always in U_3, $\Re\,\mathbf{E}[g(\boldsymbol{y})g(\boldsymbol{z})g(\boldsymbol{w})]$ is always at least $-\frac{1}{2}$. Therefore,

$$(2) \leq \tfrac{2}{3} - \tfrac{2}{3}\Re\,\mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{y})g(\boldsymbol{z})]. \quad (3)$$

It remains to handle the $\mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{y})g(\boldsymbol{z})]$ term, which is the subject of our next lemma. The proof, which appears in the full version of the paper [3], is a standard argument in the style of Håstad [13].

Lemma 3.8. $\mathbf{E}[f(\boldsymbol{x})g(\boldsymbol{y})g(\boldsymbol{z})] = \sum_{\alpha\in\mathbb{Z}_3^L}\hat{f}(\pi_3(\alpha))\hat{g}(\alpha)^2\left(-\frac{1}{2}\right)^{\#\alpha}$.

Substituting this result into (3) we obtain Lemma 3.6.

4 Hardness of 4NAT

In this section, we show the following theorem:

Theorem 4.1. *Theorem 1.4 (detailed) For all $\epsilon > 0$, it is* NP*-hard to $(1, \frac{2}{3}+\epsilon)$-decide the 4NAT problem. In fact, in the "yes case", all 4NAT constraints can be satisfied by TwoPair assignments.*

Combining this with Lemma 3.2 yields Theorem 1.3, and combining this with Corollary 3.4 yields Theorem 1.2. It is not clear whether this gives optimal hardness assuming perfect completeness. The 4NAT predicate is satisfied by a uniformly random input with probability $\frac{5}{9}$, and by the method of conditional expectation this gives a deterministic algorithm which $(1, \frac{5}{9})$-approximates the 4NAT CSP. This leaves a gap of $\frac{1}{9}$ in the soundness, and to our knowledge there are no better known algorithms.

On the hardness side, consider a uniformly random satisfying assignment to the TwoPair predicate. It is easy to see that each of the four variables is assigned a uniformly random value from $\mathbb{Z}_3$, and also that the variables are pairwise independent. As any satisfying assignment to the TwoPair predicate also satisfies the 4NAT predicate, the work of Austrin and Mossel [2] immediately implies that $(1 - \epsilon, \frac{5}{9} + \epsilon)$-approximating the 4NAT problem is NP-hard under the Unique Games conjecture. Thus, if we are willing to sacrifice a small amount in the completeness, we can improve the soundness parameter in Theorem 1.4. Whether we can improve upon the soundness without sacrificing perfect completeness is open.

The proof of Theorem 1.4, which appears in the full version [3], is entirely standard, and proceeds by reduction from d-to-1 Label Cover. It makes use of our analysis of the 4NAT Test, which is presented in Section 3.2.

References

[1] Arora, S., Barak, B., Steurer, D.: Subexponential algorithms for Unique Games and related problems. In: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science, pp. 563–572 (2010)

[2] Austrin, P., Mossel, E.: Approximation resistant predicates from pairwise independence. Computational Complexity 18(2), 249–271 (2009)

[3] Austrin, P., O'Donnell, R., Wright, J.: A new point of NP-hardness for 2-to-1 Label Cover. CoRR, abs/1204.5666 (2012)

[4] Barak, B., Raghavendra, P., Steurer, D.: Rounding semidefinite programming hierarchies via global correlation. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (2011)

[5] Dinur, I., Mossel, E., Regev, O.: Conditional hardness for approximate coloring. SIAM Journal on Computing 39(3), 843–873 (2009)

[6] Feige, U., Kilian, J.: Two prover protocols: low error at affordable rates. In: Proceedings of the 26th Annual ACM Symposium on Theory of Computing, pp. 172–183 (1994)

[7] Frieze, A., Jerrum, M.: Improved approximation algorithms for MAX k-CUT and MAX BISECTION. Algorithmica 18(1), 67–81 (1997)

[8] Goemans, M., Williamson, D.: Approximation algorithms for MAX-3-CUT and other problems via complex semidefinite programming. Journal of Computer & System Sciences 68(2), 442–470 (2004)

[9] Guruswami, V., Khot, S., O'Donnell, R., Popat, P., Tulsiani, M., Wu, Y.: SDP Gaps for 2-to-1 and Other Label-Cover Variants. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010. LNCS, vol. 6198, pp. 617–628. Springer, Heidelberg (2010)

[10] Guruswami, V., Lewin, D., Sudan, M., Trevisan, L.: A tight characterization of NP with 3 query PCPs. In: Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science, pp. 8–17 (1998)
[11] Guruswami, V., Sinop, A.: Lasserre hierarchy, higher eigenvalues, and approximation schemes for quadratic integer programming with PSD objectives. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (2011)
[12] Guruswami, V., Sinop, A.K.: Improved inapproximability results for Maximum k-Colorable Subgraph. In: Proceedings of the 12th Annual International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, pp. 163–176 (2009)
[13] Håstad, J.: Some optimal inapproximability results. Journal of the ACM 48(4), 798–859 (2001)
[14] Impagliazzo, R., Paturi, R.: On the complexity of k-SAT. Journal of Computer and System Sciences 62(2), 367–375 (2001)
[15] Khot, S.: On the power of unique 2-prover 1-round games. In: Proc. 34th ACM Symposium on Theory of Computing, pp. 767–775 (2002)
[16] Khot, S., Kindler, G., Mossel, E., O'Donnell, R.: Optimal inapproximability results for Max-Cut and other 2-variable CSPs? SIAM Journal on Computing 37(1), 319–357 (2007)
[17] Khot, S., Regev, O.: Vertex Cover might be hard to approximate to within $2-\epsilon$. In: Proc. 18th IEEE Conference on Computational Complexity, pp. 379–386 (2003)
[18] Moshkovitz, D., Roz, R.: Two-query PCP with subconstant error. Journal of the ACM 57(5), 29 (2010)
[19] O'Donnell, R., Wright, J.: A new point of NP-hardness for Unique-Games. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing, pp. 289–306 (2012)
[20] O'Donnell, R., Wu, Y.: Conditional hardness for satisfiable 3-CSPs. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 493–502 (2009)
[21] Raghavendra, P.: Optimal algorithms and inapproximability results for every CSP? In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 245–254 (2008)
[22] Raz, R.: A parallel repetition theorem. In: Proceedings of the 27th Annual ACM Symposium on Theory of Computing, pp. 447–456 (1995)
[23] Steurer, D.: Subexponential algorithms for d-to-1 two-prover games and for certifying almost perfect expansion. Available at the author's website (2010)
[24] Trevisan, L., Sorkin, G., Sudan, M., Williamson, D.: Gadgets, approximation, and linear programming. SIAM J. Comput. 29(6), 2074–2097 (2000)
[25] Zwick, U.: Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In: Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 201–210 (1998)

Inapproximability of Treewidth, One-Shot Pebbling, and Related Layout Problems*

Per Austrin, Toniann Pitassi, and Yu Wu

Department of Computer Science
University of Toronto
{austrin,toni,wuyu}@cs.toronto.edu

Abstract. We study the approximability of a number of graph problems: treewidth and pathwidth of graphs, one-shot black (and black-white) pebbling costs of directed acyclic graphs, and a variety of different graph layout problems such as minimum cut linear arrangement and interval graph completion. We show that, assuming the recently introduced Small Set Expansion Conjecture, all of these problems are hard to approximate within any constant factor.

1 Introduction

One of the great accomplishments in the last twenty years in complexity theory has been the development of ideas that has led to a deep understanding of the approximability of an astonishing number of NP-hard optimization problems. More recently, in the last ten years, the formulation of the Unique Games Conjecture (UGC) due to Khot [15] has inspired a remarkable body of work, clarifying the complexity of many optimization problems, and exposing the central role of semidefinite programming in the development of approximation algorithms.

Despite this tremendous progress, for certain expansion problems such as the c-Balanced Separator problem, and graph layout problems such as the Minimum Linear Arrangement (MLA) problem, their approximation status remained unresolved. That is, even assuming the UGC is not known to be sufficient to obtain hardness of approximation for either of these problems. Moreover, the approximability of many other graph layout problems is similarly unresolved, even under the UGC. Intuitively this is because the hard instances for these problems seem to require a certain global structure such as expansion. Typical reductions for these problems are gadget reductions which preserve global properties of the unique games instance, such as the lack of expansion. Therefore, barring radically new types of reductions that do not preserve global properties, proving hardness for c-Balanced Separator seems to require a stronger version of UGC, where the instance is guaranteed to have good expansion.

In [21], the Small Set Expansion (SSE) Conjecture was introduced, and it was shown that it implies the UGC, and that the SSE Conjecture follows if

* Research supported by NSERC.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 13–24, 2012.
© Springer-Verlag Berlin Heidelberg 2012

one assumes that the UGC is true for somewhat expanding graphs. In follow-up work by Raghavendra et al. [22], it was shown that the SSE Conjecture is in fact equivalent to the UGC on somewhat expanding graphs, and that the SSE Conjecture implies hardness of approximation for c-Balanced Separator and MLA. In this light, the Small Set Expansion conjecture serves as a natural unified conjecture that yields all of the implications of UGC and also hardness for expansion-like problems that appear to be beyond the reach of the UGC.

In this paper, we study the approximability of a host of such graph layout problems, including: treewidth and pathwidth of graphs, one-shot black and black-white pebbling, Minimum Cut Linear Arrangement (MCLA) and Interval Graph Completion (IGC). We prove that all of these problems are SSE-hard to approximate to within any constant factor. Our main contributions, giving SSE-hardness of approximation for all of the graph layout problems mentioned above, are described in the following subsections. For all of these problems, no evidence of hardness of approximation was known prior to our results.

It should be noted that the status of the SSE Conjecture is very open at this point. In particular, by the recent result of Arora et al. [3] (see also subsequent work [5, 14]), it has algorithms running in subexponential time. Still, despite this recent progress providing negative evidence against the SSE Conjecture, it remains open, and we think that investigating what open problems in approximability we can show SSE-hardness for is a worthwhile venture.

1.1 Width Parameters of Graphs

The *treewidth* of a graph, introduced by Robertson and Seymour [24, 25], is a fundamental parameter of a graph that measures how close a graph is to being a tree. The concept is very important since problems of small treewidth can usually be solved efficiently by dynamic programming. Indeed, a large body of NP-hard problems (including all problems definable in monadic second-order logic [11]) are solvable in polynomial time and often even linear time on graphs of bounded treewidth. Examples of such optimization problems include finding a maximum independent set or a Hamiltonian cycle in a graph. In machine learning, tree decompositions play a key role in the development of efficient algorithms for fundamental problems such as probabilistic inference, constraint satisfaction and query optimization. (See the excellent survey [8] for motivation, including theoretical as well as practical applications of treewidth.)

The complexity of approximating treewidth is a longstanding open problem. Determining the exact treewidth of a graph and producing an associated optimal tree decomposition (see Definition 2.3) is known to be NP-hard [2]. A central open problem is to determine whether or not there exists a polynomial time constant factor approximation algorithm for treewidth (see e.g., [9, 13, 8]). The current best polynomial time approximation algortihm for treewidth [13], computes the treewidth $\mathsf{tw}(G)$ within a factor $O(\sqrt{\log \mathsf{tw}(G)})$. On the other hand, the only hardness result to date for treewidth shows that it is NP-hard to compute treewidth within an *additive* error of n^ϵ for some $\epsilon > 0$ [9]. No hardness of approximation is known and not even the possibility of a polynomial-time

approximation scheme for treewidth has been ruled out. In many important special classes of graphs, such as planar graphs [27] and H-minor-free graphs [13], constant factor approximations are known, but the general case has remained elusive.

On the positive side, there is a large body of literature developing fixed-parameter algorithms for treewidth. In particular, when the runtime is allowed to be exponential in the $\mathsf{tw}(G)$ there are constant factor approximations. Furthermore, even exactly determining the treewidth is fixed-parameter tractable: there is a linear time algorithm for computing the (exact) treewidth for graphs of constant treewidth [7].

A related graph parameter is the so-called *pathwidth*, which can be viewed as measuring how close G is to a path. The pathwidth $\mathsf{pw}(G)$ is always at least $\mathsf{tw}(G)$, but can be much larger. The current state of affairs here is similar as for treewidth; though the current best approximation algorithm only has an approximation ratio of $O(\sqrt{\log \mathsf{pw}(G)} \log n)$ [13], the best hardness result is NP-hardness of additive n^ϵ error approximation.

Using the recently proposed *Small Set Expansion* (SSE) Conjecture [21] discussed earlier, we show that both $\mathsf{tw}(G)$ and $\mathsf{pw}(G)$ are hard to approximate within any constant factor. In fact, we show something stronger: it is hard to distinguish graphs with small pathwidth from graphs with large treewidth.

Theorem 1.1. *For every $\alpha > 1$ there is a $c > 0$ such that given a graph $G = (V, E)$ it is SSE-hard to distinguish between the case when $\mathsf{pw}(G) \leq c \cdot |V|$ and the case when $\mathsf{tw}(G) \geq \alpha \cdot c \cdot |V|$. In particular, both treewidth and pathwidth are SSE-hard to approximate within any constant factor.*

This is the first result giving hardness of (relative) approximation for these problems, and gives evidence that no constant factor approximation algorithm exists for either of them.

1.2 Pebbling Problems

Graph pebbling is a rich and relatively mature topic in theoretical computer science. *Pebbling* is a game defined on a directed acyclic graph (DAG), where the goal is to *pebble* the sink nodes of the DAG according to certain rules, using the minimum number of pebbles. The rules for pebbling are as follows. A *black pebble* can be placed on a node if all of the node's immediate predecessors contain pebbles, and can always be removed. A white pebble can always be placed on a node, but can only be removed if all of the node's immediate predecessors contain pebbles. A pebbling *strategy* is a process of pebbling the sink nodes in a graph according to the above rules. The *pebbling cost* of a pebbling strategy is the maximum number of pebbles used in the strategy. The black-white pebbling cost of a DAG is the minimum pebbling cost of all possible pebbling strategies. The black pebbling cost is the minimum pebbling cost over all pebbling strategies that only use black pebbles.

Pebbling games were originally devised for studying programming languages and compiler construction, but have later found a broad range of applications

in computational complexity theory. Pebbling is a tool for studying the relationship between computation time and space by means of a game played on directed acyclic graphs. It was employed to model register allocation, and to analyze the relative power of time and space as Turing machine resources. For a comprehensive recent survey on graph pebbling, see [20].

Apart from the cost of a pebbling, another important measure is the *pebbling time*, which is the number of steps (pebble placements/removals) performed. In the context of measuring memory used by computations, this corresponds to computation time, and hence keeping the pebbling time small is a natural priority. The extreme case of this is what we refer to as *one-shot pebbling*, also known as progressive pebbling (see e.g., [26, 18, 17]). In one-shot pebbling, we have the restriction that each node can be pebbled only once. Note that this restriction can cause a huge increase in the pebbling cost of the graph [19].

The one-shot pebbling problem is easier to analyze for the following reasons. In the original pebbling problem, in order to achieve the minimum pebbling number, the pebbling time might be required to be exponentially long, which becomes impractical when n is large. On the other hand, the one-shot pebbling problem is more amenable to complexity theoretic analysis as it minimizes the space used in a computation subject to the execution time being minimum. In particular, the decision problem for one-shot pebbling is in NP (whereas the unrestricted pebbling problems are PSPACE-complete).

The one-shot black/black-white pebbling problems admit $O(\sqrt{\log n}\log n)$ approximation ratios. We show that they are SSE-hard to approximate to within any constant factor. For black pebbling we show that this holds for single sink DAGs with in-degree 2, which is the canonical setting for pebbling games (it seems plausible that the black-white hardness can be shown to hold for this case as well, though we have not attempted to prove this).

Theorem 1.2. *It is SSE-hard to approximate the one-shot black pebbling problem within any constant factor, even in DAGs with a single sink and maximum in-degree* 2*.*

Theorem 1.3. *It is SSE-hard to approximate the one-shot black-white pebbling problem within any constant factor.*

No hardness of approximation result of any form was known for one-shot pebbling problems. We believe that these results can be extended to obtain hardness for more relaxed versions of bounded time pebbling costs as well. We are currently working on this, and have some preliminary results.

1.3 The Connection: Layout Problems

The graph width and one-shot pebbling problems discussed in the previous sections may at first glance appear to be unrelated. However, both sets of problems are instances of a general family of problems, known as *graph layout problems*. In a graph layout problem (also known as an arrangement problem, or a vertex ordering problem), the goal is to find an ordering of the vertices, optimizing some

condition on the edges, such as adjacent pairs being close. Layout problems are an important class of problems that have applications in many areas such as VLSI circuit design.

A classic example is the *Minimum Cut Linear Arrangement* (MCLA) Problem. In this problem, the objective is to find a permutation π of the vertices V of an undirected graph $G = (V, E)$, such that the largest number of edges crossing any point,

$$\max_i |\{(u, v) \in E | \pi(u) \leq i < \pi(v)\}|, \tag{1}$$

is minimized. MCLA is closely related to the *Minimum Linear Arrangement* Problem (MLA), in which the max in (1) is replaced by a sum.

The MCLA problem can be approximated to within a factor $O(\log n \sqrt{\log n})$. To the best of our knowledge, there is no hardness of approximation for MCLA in the literature. Its cousin MLA was recently proved SSE-hard to approximate within any constant factor [22], and we observe that the same hardness applies to the MCLA problem.

Theorem 1.4. *Assuming the SSE Conjecture, Minimum Cut Linear Arrangement is hard to approximate within any constant factor.*

Another example of graph layout is the *Interval Graph Completion* Problem (IGC). In this problem, the objective is to find a supergraph $G' = (V, E')$ of G such that G' is an interval graph (i.e., the intersection graph of a set of intervals on the real line) and of minimum size. While not immediately appearing to be a layout problem, using a simple structural characterization of interval graphs [23] one can show that IGC can be reformulated as finding a permutation of the vertices that minimizes the sum over the longest edges going out from each vertex, i.e., minimizing

$$\sum_{u \in V} \max_{(u,v) \in E} \max\{\pi(v) - \pi(u), 0\}. \tag{2}$$

See e.g., [10]. The current best approximation algorithm for IGC achieves a ratio of $O(\sqrt{\log n} \log \log n)$ [10]. It turns out that the SSE Conjecture can be used to prove super-constant hardness for this problem as well.

Theorem 1.5. *Assuming the SSE Conjecture, Interval Graph Completion is hard to approximate within any constant factor.*

There is a distinction in IGC of whether one counts the number of edges in the final interval graph – this is the most common definition – or whether one only counts the number of edges added to make G an interval graph (which makes the problem harder from an approximability viewpoint). Our result holds for the common definition and therefore applies also to the harder version.

Theorems 1.4 and 1.5 are just two examples of layout problems that we prove hardness of approximation for. By varying the precise objective function and also considering directed acyclic graphs, in which case the permutation π must be a topological ordering of the graph, one can obtain a wide variety of graph

layout problems. We consider a set of eight such problems, generated by three natural variations (see Section 2.1 for precise details), and show super-constant SSE-based hardness for all of them in a unified way. This set of problems includes MLA, MCLA, and IGC, but not problems such as Bandwidth (but on the other hand, strong NP-hardness inapproximability results for Bandwidth are already known [12]). See Table 1 in Section 2.1 for a complete list of problems covered.

Theorem 1.6. *Assuming the SSE Conjecture, all problems listed in Table 1 (see page 20) are hard to approximate to within any constant factor.*

Let us now return to the problems discussed in the previous sections. It should not be surprising that the one-shot black pebbling problem is equivalent to a graph layout problem: the one-shot constraint reduces the problem to determining in which order to pebble the vertices; such an ordering induces a pebbling strategy in an obvious way. For the black-white case, it is known that the one-shot black-white pebbling cost of D is interreducible with a layout problem on an undirected graph G. Both of these layout problems are included in the set of problems we show hardness for, so Theorems 1.2 and 1.3 follow immediately from Theorem 1.6.

Turning to the width parameters, treewidth is equivalent to a graph layout problem called elimination width. Here the objective function is somewhat more intricate than in the set of basic layout problems we consider in Theorem 1.6, but we are able to extend those results to hold also for elimination width. Pathwidth is also known to be equivalent to a certain graph layout problem, and in fact is equivalent to the layout problem which one-shot black-white pebbling reduces to. We use these connections to prove the hardness of approximation for both treewidth and pathwidth, thereby obtaining Theorem 1.1.

1.4 Previous Work

As the reader may have noticed, for all the problems mentioned, the best current algorithms achieve similar poly-logarithmic approximation ratios. Given their close relation, this is of course not surprising. Most of the algorithms are obtained by recursively applying some algorithm for the c-balanced separator problem, An improved algorithm for c-balanced separator will also improve the approximation algorithms for the various layout problems. On the other hand, hardness of approximating c-balanced separator [22] does not necessarily imply hardness of approximating layout problems.

On the hardness side, our work builds upon the work of [22], which showed that the SSE Conjecture implies superconstant hardness of approximation for MLA (and for c-balanced separator). The only other hardness of relative approximation that we are aware of for these problems is a result of Ambühl et al. [1], showing that MLA does not have a PTAS unless NP has randomized subexponential time algorithms.

1.5 Organization

The outline for the rest of the paper is as follows. In Section 2, we formally define the layout problems studied as well as treewidth and pathwidth. Section 3 gives a high level overview of the reductions used, and some concluding remarks and open problems are given in Section 4. Full proofs can be found in the full version of the paper [4].

2 Definitions and Preliminaries

2.1 Graph Layout Problems

In this section, we describe the set of graph layout problems that we consider. A problem from the set is described by three parameters, giving rise to several different problems. These three parameters are by no means the only interesting graph layout problems (and some of the settings give rise to more or less uninteresting layout problems). However, they are sufficient to capture the problems we are interested in except treewidth, which in principle could be incorporated as well though we refrain from doing so in order to keep the definitions simple (see Section 2.2 for more details).

First a word on notation. Throughout the paper, $G = (V, E)$ denotes an undirected graph, and $D = (V, E)$ denotes a directed (acyclic) graph. Letting n denote the number of vertices of the graph, we are interested in bijective mappings $\pi : V \to [n]$. We say that an edge $(u, v) \in E$ *crosses* point $i \in [n]$ (with respect to the permutation π, which will always be clear from context), if $\pi(u) \leq i < \pi(v)$.

We consider the following variations:

1. **Undirected or directed acyclic:** In the case of an undirected graph G, any ordering π of the vertices is a feasible solution. In the case of a DAG D, only the topological orderings of D are feasible solutions.
2. **Counting edges or vertices:** for a point $i \in [n]$ of the ordering, we are interested in the set $E_i(\pi)$ of edges crossing this point. When counting edges, we use the cardinality of E_i as our basic measure. When counting vertices, we only count the set of vertices V_i to the left of i that are incident upon some edge crossing i. In other words, V_i is the projection of $E_i(\pi)$ to the left-hand side vertices. Formally:

$$E_i(\pi) = \{e \in E \mid \pi(u) \leq i < \pi(v) \text{ where } e = (u, v)\}$$
$$V_i(\pi) = \{u \in V \mid \pi(u) \leq i < \pi(v) \text{ for some } (u, v) \in E\}$$

   We refer to $|E_i(\pi)|$ or $|V_i(\pi)|$ (depending on whether we are counting edges or vertices) as the *cost* of π at i.
3. **Aggregation by sum or max:** given an ordering π, we aggregate the costs of each point $i \in [n]$, by either summation or by taking the maximum cost.

Given these choices, the objective is to find a feasible ordering π that minimizes the aggregated cost.

Definition 2.1 *(Layout value). For a graph H (either an undirected graph G or a DAG D), a cost function C (either E or V), and an aggregation function* $\mathrm{agg} : \mathbb{R}^* \to \mathbb{R}$ *(either Σ or* max*), we define* $\mathsf{Layout}(H; C, \mathrm{agg})$ *as the minimum aggregated cost over all feasible orderings of H. Formally:*

$$\mathsf{Layout}(H; C, \mathrm{agg}) = \min_{\textit{feasible } \pi} \mathop{\mathrm{agg}}_{i \in [n]} |C_i(\pi)|.$$

Example 2.2. $\mathsf{Layout}(G; E, \max) = \min_\pi \max_{i \in [n]} |E_i(\pi)|$, where π ranges over all orderings of $V(G)$. This we recognize from Section 1.3 as the Minimum Cut Linear Arrangement value of G.

Combining the different choices gives rise to a total of eight layout problems (some more natural than others). Several of these appear in the literature under one or more names, and some turn out to be equivalent[1] to problems that at first sight appear to be different. We summarize some of these names in Table 1 (in some cases the standard definitions of these problems look somewhat different than the unified definition given here, e.g., for pathwidth, one-shot pebblings, and interval graph completion).

Table 1. Taxonomy of Layout Problems

Problem			Also known as / Equivalent with
undir.	edge	sum	Minimum/Optimal Linear Arrangement
undir.	edge	max	Minimum Cut Linear Arrangement CutWidth
undir.	vertex	sum	Interval Graph Completion SumCut
undir.	vertex	max	Pathwidth One-shot Black-White Pebbling Vertex Separation
DAG	edge	sum	Minimum Storage-Time Sequencing Directed MLA/OLA
DAG	edge	max	
DAG	vertex	sum	
DAG	vertex	max	One-shot Black Pebbling Register Sufficiency

2.2 Treewidth and Pathwidth

Definition 2.3 (Tree decomposition, Treewidth). *Let $G = (V, E)$ be a graph, T a tree, and let $\mathcal{V} = (V_t)_{t \in T}$ be a family of vertex sets $V_t \subseteq V$ indexed by the vertices t of T. The pair $(T, \mathcal{V})$ is called a* tree decomposition *of G if it satisfies the following three conditions:*

[1] Here, we consider two optimization problems equivalent if there are reductions between them that change the objective values by at most an additive constant.

(T1) $V = \cup_{t \in T} V_t$*;*
(T2) for every edge $e \in E$*, there exists a* $t \in T$ *such that both endpoints of* e *lie in* V_t*;*
(T3) for every vertex $v \in V$*,* $\{t \in T \mid v \in V_t\}$ *is a subtree of* T*'.*

The width of $(T, \mathcal{V})$ *is the number* $\max\{|V_t| - 1 \mid t \in T\}$*, and the* treewidth *of* G*, denoted* $\mathsf{tw}(G)$*, is the minimum width of any tree decomposition of* G*.*

Treewidth can be characterized in terms of elimination width, which is another example of a layout problem (see e.g., [6]). In principle this layout problem can be formulated in the framework of Section 2.1, but the choice of cost function is now more involved than the vertex- and edge-counting considered there.

Definition 2.4 (Path decomposition, Pathwidth). *Given a graph* G*, we say that* $(T, \mathcal{V})$ *is a* path decomposition *of* G *if it is a tree decomposition of* G *and* T *is a path. The* pathwidth *of* G*, denoted* $\mathsf{pw}(G)$*, is the minimum width of any path decomposition of* G*.*

As claimed earlier, pathwidth is in fact equivalent with a graph layout problem:

Theorem 2.5 ([16]). *For every graph* G*,* $\mathsf{pw}(G) = \mathsf{Layout}(G; V, \max)$*, also known (among many other names) as the "vertex separation" number of* G*.*

2.3 Small Set Expansion Conjecture

In this section we define the SSE Conjecture. Let $G = (V, E)$ be an undirected d-regular graph. For a set $S \subseteq V$ of vertices, we write $\Phi_G(S)$ for the (normalized) edge expansion of S,

$$\Phi_G(S) = \frac{|E(S, V \setminus S)|}{d|S|}$$

The Small Set Expansion Problem with parameters η and δ, denoted $\mathsf{SSE}(\eta, \delta)$, asks if G has a small set S which does not expand or whether all small sets are highly expanding.

Definition 2.6 ($\mathsf{SSE}(\eta, \delta)$). *Given a* d*-regular graph* $G = (V, E)$*,* $\mathsf{SSE}(\eta, \delta)$ *is the problem of distinguishing between the following two cases:*

Yes *There is an* $S \subseteq V$ *with* $|S| = \delta|V|$ *and* $\Phi_G(S) \leq \eta$*.*
No *For every* $S \subseteq V$ *with* $|S| = \delta|V|$ *it holds that* $\Phi_G(S) \geq 1 - \eta$*.*

This problem was introduced by Raghavendra and Steurer [21], who conjectured that the problem is hard.

Conjecture 2.7 (Small Set Expansion Conjecture). *For every* $\eta > 0$*, there is a* $\delta > 0$ *such that* $\mathsf{SSE}(\eta, \delta)$ *is NP-hard.*

As has become common for a conjecture like this (such as the Unique Games Conjecture), we say that a problem is *SSE-hard* if it is as hard to solve as the SSE problem. Formally, a decision problem $\mathcal{P}$ (e.g., a gap version of some optimization problem) is *SSE-hard* if there is some $\eta > 0$ such that for every $\delta > 0$, $\mathsf{SSE}(\eta, \delta)$ polynomially reduces to $\mathcal{P}$.

3 Brief Overview of Reductions

We now give a very brief overview of the reductions used to prove that the layout problems of Table 1 are SSE-hard to approximate within any constant factor. The details of these reductions can be found in the full version of the paper [4].

For the two undirected edge problems (i.e., MLA and MCLA), the hardness follows immediately from the strong form of the SSE Conjecture – for the case of MLA this was proved in [22] and the proof for MCLA is similar. This is our starting point for the remaining problems. Unfortunately, the results do not follow from hardness for MLA/MCLA in a black-box way; for the soundness analyses we end up having to use the expansion properties of the original SSE instance.

We then give a reduction from MLA/MCLA with expansion, to the four directed problems. This reduction simply creates the bipartite graph where the vertex set is the union of the edges and vertices of the original graph G, with directed arcs from an edge e to the vertices incident upon e in G. The use of direction here is crucial: it essentially ensures that both the vertex and edge counts of any feasible ordering corresponds very closely to the number of edges crossing the point in the induced ordering of G.

To obtain hardness for the remaining two undirected problems, we perform a similar reduction as for the directed case, creating the bipartite graph of edge-vertex incidences. However, since we are now creating an undirected graph, we can no longer force the edges to be chosen before the vertices upon which they are incident, which was a key property in the reduction for the directed case. In order to overcome this, we duplicate each original vertex a large number of times. This gives huge penalties to orderings which do not "essentially" obey the desired direction of the edges, and makes the reduction work out.

The results for treewidth follows from an additional analysis of the instances produced by the reduction for undirected vertex problems. Finally, the reduction for directed problems, implying hardness for one-shot black pebbling, does not produce the kind of "nice" instances promised by Theorem 1.2. We give some additional transformation to achieve these properties in the full version as well.

Figure 1 gives a high-level overview of these reductions.

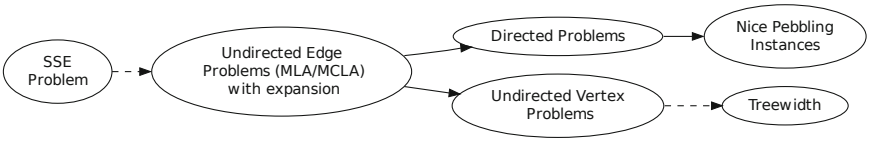

Fig. 1. Overview of Reductions. Dashed arrows indicate that the reduction is obtained by the identity mapping, whereas solid arrows indicate a nontrivial transformation from one problem to the other.

4 Conclusion and Open Problems

We proved SSE-hardness of approximation for a variety of graph problems. Most importantly we obtained the first inapproximability result for the treewidth problem. Some remarks are in order. The status of the SSE conjecture is, at this point in time, very uncertain, and our results should therefore not be taken as absolute evidence that there is no polynomial time approximation for (e.g.) treewidth. However, at the very least, our results do give an indication of the difficulty involved in obtaining such an algorithm for treewidth, and builds a connection between these two important problems. We also find it remarkable how simple our reductions and proofs are. We leave the choice of whether to view this as a healthy sign of strength of the SSE Conjecture, or whether to view it as an indication that the conjecture is too strong, to the reader.

There are many important open questions and natural avenues for further work, including:

1. It seems plausible that these results can be extended to a wider range of graph layout problems. For instance, our two choices of aggregators max and Σ can be viewed as taking ℓ_∞ and ℓ_1 norms, and it seems likely that the results would apply for any ℓ_p norm (though we are not aware of any previous literature studying such variants).
2. It would be nice to obtain hardness of approximation result for our problems based on a weaker hardness assumption such as UGC. It is conjectured in [22] that the SSE conjecture is equivalent to UGC. Alternatively, it would be nice to show that hardness of some of our problems imply hardness for the SSE Problem.
3. For pebbling, it would be very interesting to obtain results for the unrestricted pebbling problems (for which finding the exact pebbling cost is even PSPACE-hard). As far as we are aware, nothing is known for these problems, not even, say, whether one can obtain a non-trivial approximation in NP. As mentioned in the introduction, we are currently working on extending our one-shot pebbling results to bounded time pebblings. We have some preliminary progress there and are hopeful that we can relax the pebbling results to a much larger class of pebblings.

References

[1] Ambuhl, C., Mastrolilli, M., Svensson, O.: Inapproximability Results for Sparsest Cut, Optimal Linear Arrangement, and Precedence Constrained Scheduling. In: Proceedings of the IEEE Symposium on Foundations of Computer Science, pp. 329–337 (2007)
[2] Arnborg, S., Corneil, D.G., Proskurowski, A.: Complexity of finding embeddings in a k-tree. SIAM J. Algebraic Discrete Methods 8, 277–284 (1987)
[3] Arora, S., Barak, B., Steurer, D.: Subexponential Algorithms for Unique Games and Related Problems. In: FOCS, pp. 563–572 (2010)
[4] Austrin, P., Pitassi, T., Wu, Y.: Inapproximability of treewidth, one-shot pebbling, and related layout problems. CoRR, abs/1109.4910 (2011)

[5] Barak, B., Raghavendra, P., Steurer, D.: Rounding Semidefinite Programming Hierarchies via Global Correlation. In: FOCS, pp. 472–481 (2011)
[6] Bodlaender, H.L.: Treewidth: Structure and Algorithms. In: Prencipe, G., Zaks, S. (eds.) SIROCCO 2007. LNCS, vol. 4474, pp. 11–25. Springer, Heidelberg (2007)
[7] Bodlaender, H.L.: A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. SIAM J. Comput. 25(6), 1305–1317 (1996)
[8] Bodlaender, H.L.: Discovering Treewidth. In: Vojtáš, P., Bieliková, M., Charron-Bost, B., Sýkora, O. (eds.) SOFSEM 2005. LNCS, vol. 3381, pp. 1–16. Springer, Heidelberg (2005)
[9] Bodlaender, H.L., Gilbert, J.R., Hafsteinsson, H., Kloks, T.: Approximating treewidth, pathwidth, frontsize, and shortest elimination tree. Journal of Algorithms 18(2), 238–255 (1995)
[10] Charikar, M., Hajiaghayi, M., Karloff, H., Rao, S.: l_2^2 spreading metrics for vertex ordering problems. Algorithmica 56, 577–604 (2010)
[11] Courcelle, B.: Graph Rewriting: An Algebraic and Logic Approach. In: Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B), pp. 193–242 (1990)
[12] Dubey, C.K., Feige, U., Unger, W.: Hardness results for approximating the bandwidth. J. Comput. Syst. Sci. 77(1), 62–90 (2011)
[13] Feige, U., Hajiaghayi, M., Lee, J.R.: Improved approximation algorithms for minimum-weight vertex separators. In: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing, pp. 563–572 (2005)
[14] Guruswami, V., Sinop, A.K.: Lasserre Hierarchy, Higher Eigenvalues, and Approximation Schemes for Graph Partitioning and Quadratic Integer Programming with PSD Objectives. In: FOCS, pp. 482–491 (2011)
[15] Khot, S.: On the power of unique 2-prover 1-round games. In: Proceedings of the ACM Symposium on Theory of Computing, STOC 2002, pp. 767–775 (2002)
[16] Kinnersley, N.G.: The vertex separation number of a graph equals its path-width. Information Processing Letters 42(6), 345–350 (1992)
[17] Kirousis, L.M., Papadimitriou, C.H.: Searching and pebbling. Theor. Comput. Sci. 47, 205–218 (1986)
[18] Lengauer, T.: Black-white pebbles and graph separation. Acta Informatica 16, 465–475 (1981), doi:10.1007/BF00264496
[19] Lengauer, T., Tarjan, R.E.: Asymptotically tight bounds on time-space trade-offs in a pebble game. J. ACM 29, 1087–1130 (1982)
[20] Nordström, J.: New wine into old wineskins: A survey of some pebbling classics with supplemental results. Draft manuscript (November 2010)
[21] Raghavendra, P., Steurer, D.: Graph expansion and the unique games conjecture. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, pp. 755–764. ACM, New York (2010)
[22] Raghavendra, P., Steurer, D., Tulsiani, M.: Reductions Between Expansion Problems. To appear in CCC (2012)
[23] Ramalingam, G., Rangan, C.P.: A unified approach to domination problems on interval graphs. Inf. Process. Lett. 27, 271–274 (1988)
[24] Robertson, N., Seymour, P.D.: Graph minors. III. Planar tree-width. J. Comb. Theory, Ser. B 36(1), 49–64 (1984)
[25] Robertson, N., Seymour, P.D.: Graph minors. II. Algorithmic aspects of tree-width. Journal of Algorithms 7(3), 309–322 (1986)
[26] Sethi, R.: Complete register allocation problems. In: Proceedings of the Fifth Annual ACM Symposium on Theory of Computing, STOC 1973, pp. 182–195 (1973)
[27] Seymour, P.D., Thomas, R.: Call routing and the ratcatcher. Combinatorica 14(2), 217–241 (1994)

Additive Approximation for Near-Perfect Phylogeny Construction*

Pranjal Awasthi, Avrim Blum, Jamie Morgenstern, and Or Sheffet

Carnegie Mellon University, Pittsburgh,
5000 Forbes Ave., Pittsburgh PA 15213, USA
{pawasthi,avrim,jamiemmt,osheffet}@cs.cmu.edu

Abstract. We study the problem of constructing phylogenetic trees for a given set of species. The problem is formulated as that of finding a minimum Steiner tree on n points over the Boolean hypercube of dimension d. It is known that an optimal tree can be found in linear time [1] if the given dataset has a perfect phylogeny, i.e. cost of the optimal phylogeny is exactly d. Moreover, if the data has a near-perfect phylogeny, i.e. the cost of the optimal Steiner tree is $d + q$, it is known [2] that an exact solution can be found in running time which is polynomial in the number of species and d, yet exponential in q. In this work, we give a polynomial-time algorithm (in both d and q) that finds a phylogenetic tree of cost $d+O(q^2)$. This provides the best guarantees known—namely, a $(1 + o(1))$-approximation—for the case $\log(d) \ll q \ll \sqrt{d}$, broadening the range of settings for which near-optimal solutions can be efficiently found. We also discuss the motivation and reasoning for studying such additive approximations.

1 Introduction

Phylogenetics, a subfield of computational biology, aims to construct simple and accurate descriptions of evolutionary history. These descriptions are represented as evolutionary trees for a given set of species, each of which is represented by some set of features ([3, 4]). A typical choice for these features are single nucleotide polymorphisms (SNPs), binary indicator variables for common mutations found in DNA[5, 6]; see, for example, [2, 1, 7–10]. This challenging problem has attracted much attention in recent years, with progress in studying various computational formulations of this problem ([3, 11, 2, 1, 12, 7]). The problem is often posed as that of constructing the most parsimonious tree induced by the set of species.

Formally, a *phylogeny* or a *phylogenetic tree* for a set C of n species, each represented by a string (called taxa) of length d over a finite alphabet Σ, is an unrooted tree $T = (V, E)$ such that $C \subseteq V \subseteq \Sigma^d$. Given a distance metric μ over

* This work was supported in part by the National Science Foundation under grant CCF-1116892, by an NSF Graduate Fellowship, and by the MSR-CMU Center for Computational Thinking.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 25–36, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Σ^d, we define the cost of T as $\sum_{(u,v)\in E} \mu(u,v)$. The tree of *maximum parsimony* for a dataset is the tree which minimizes this cost with respect to the Hamming metric; i.e., it is the optimum Steiner tree for the set C under this metric.

The Steiner tree problem is known to be NP-hard in general [13], and remains NP-hard even in the case of a binary alphabet with the metric induced by the Hamming distance [14]. Extensive recent work, both experimental and theoretical, has focused on the binary character set with the Hamming metric ([3, 2, 1, 12, 7, 4, 15, 16]). This version of the phylogeny problem will also be the focus of this paper.

A phylogeny is called *perfect* if each coordinate $i \in [d]$ flips exactly once in the tree (representing a single mutation of i amongst the set of species)[1]. If a dataset admits a perfect phylogeny, an optimal tree can be constructed in polynomial time [17] (even linear time, in the case where the alphabet is binary [3]). In this work, we investigate *near perfect* phylogenies – instances whose optimal phylogenetic tree has cost $d + q$, where $q \ll d$. Near perfect phylogenies have been studied in theoretical ([11, 2, 12, 16]) and experimental settings ([15]). The work of [11, 2, 12, 16] has given a series of randomized algorithms which find the optimal phylogeny in running time polynomial in n and d but exponential in q. Clearly, when $q = \omega(\log d)$, these algorithms are not tractable.

An alternative approach for finding a phylogenetic tree of low cost is to use a generic Steiner tree approximation algorithm. The best current such algorithm yields a tree of cost at most $1.39(d+q)$ [18] (we comment that the exponential size of the explicit hypercube with respect to its small representation size requires one implement such an algorithm using techniques devised especially for the hypercube, e.g. Alon et al. [7].) However, notice that for moderate q (e.g., for $q = polylog(d)$), the *excess* of this tree—meaning the difference between its cost and d—may be extremely large compared to the excess q of the optimal tree. In such cases, one would much prefer an algorithm whose excess could be written as a function of q only.

In this work, we present a randomized $\mathrm{poly}(n, d, q)$-time algorithm that finds a phylogenetic tree of cost $d + O(q^2)$.

Theorem 1. *Given a set $C \subseteq \{0,1\}^d$ of n terminals, such that the optimal phylogeny of C has cost $d+q$, there exists a randomized $\mathrm{poly}(n, d, q)$-time algorithm that finds a phylogenetic tree of cost $d + O(q^2)$ w.p. $\geq 1/2$.*

Note that Theorem 1 provides a substantial improvement over prior work for the case that $\log d \ll q \ll \sqrt{d}$. In this range, the exact algorithms are no longer tractable, and the multiplicative approximations yield significantly worse bounds. Alternatively, viewed as a multiplicative guarantee, in this range our tree is within a $1 + o(1)$ factor of optimal. To the best of our knowledge, this is the first work to give an additive poly-time approximation to either the phylogeny problem or any (non-trivial) setting the Steiner tree problem. One immediate

[1] Without loss of generality, we may assume each coordinate flips at least once, since all coordinates on which all species agree may be discarded up front.

question, which remains open, is whether our results can be improved to $d+o(q^2)$ or perhaps even to $d + O(q)$.

The rest of the paper is organized as follows. After surveying related work in Section 1.1, we detail notation and preliminaries in Section 2. The presentation of our algorithm is partitioned into two parts. In Section 3, we present the algorithm for the case where no pair of coordinates is identical over all terminals (formal definition there). In Section 4, we alter the algorithm for the simple case, in a nontrivial way, so that the modified algorithm finds a low-cost phylogeny for any dataset. We conclude in Section 5 with a discussion, motivating the problem of near-perfect phylogeny tree from a different perspective, and present open problems for future research.

1.1 Related Work

As mentioned in the introduction, the problem of constructing an optimal phylogeny is NP-complete even when restricted to binary alphabets [14]. Schwartz et al. [11] give an algorithm based on an Integer Linear Programming (ILP) formulation to solve the multi-state problem optimally, and show experimentally the algorithm is efficient on small instances. Perfect phylogenies (datasets which admit a tree in which any coordinate changes exactly once) have optimal parsimony trees which can be constructed in linear time in the binary case [1] and in polynomial time for a fixed alphabet [12]. Unfortunately, finding the perfect phylogeny for arbitrary alphabets is NP-hard [19]. Recent work [2] gives an algorithm to construct optimal phylogenetic trees for binary, near-perfect phylogenies (where only a small number of coordinates mutate more than once in the optimal tree). However, the running time of the algorithm presented in their work [2] is exponential in the number of additional mutations.

There has also been a lot of work on computing multiplicative approximations to the Steiner tree problem. A Minimum Spanning Tree (MST) over the set of terminals achieves an approximation ratio of 2 and a long line of work has led to the current best bound of 1.39 [20–24, 8, 25, 9, 10, 18]. The more recent of these papers use a result due to Borchers and Du [26] showing that an optimal Steiner tree can be approximated to arbitrary precision using k-restricted Steiner trees.

Some of these approximations to the Steiner tree problem are not immediately extendable to the problem of constructing phylogenetic trees. This is because the size of the vertex set for the phylogeny problem is exponential in d (there are 2^d vertices in the hypercube). If an algorithm works on an explicit representation of the graph G defined by the hypercube, then it does not solve the phylogeny problem in polynomial time. However, the line of work started by Robins and Zelikovsky [9, 10] used the notion of k-restricted Steiner trees, which *can* be efficiently implemented on the hypercube. In particular, Alon et al. [7] showed that in finding the optimal k-restricted component for a given set of k terminals, it is sufficient to only consider topologies with the given k terminals at the leaves. Using this, they were able to extend that work to achieve a 1.55 approximation ratio for the maximum parsimony problem, and a 16/9 approximation for maximum likelihood. Byrka et al. [18] considered a new LP relaxation to the

k-restricted Steiner tree problem and achieved an approximation ratio of 1.39, which can be combined with the topological argument from Alon et al. [7] to achieve the same ratio for phylogenies.

2 Notation and Preliminaries

Our dataset $C \subseteq \{0,1\}^d$ consists of n terminals over d binary coordinates. A Steiner tree (or phylogeny) over C consists of a tree T on the hypercube that spans C (plus possibly additional Steiner nodes), where we label each edge e in T with the index $i \in \{1,\ldots,d\}$ of the coordinate flipped on edge e. The cost of such a Steiner tree is the number of edges in the tree. Given a collection of datasets $\mathcal{P} = \{P_1, P_2, \ldots, P_k\} \subseteq C$ we define the Steiner forest problem as the problem of finding a minimal Steiner tree on every $P \in \mathcal{P}$ separately. We refer to such collection as a partition from now on, even though it may contain a subset of the original terminal set C.

In this work, we consider instances C whose minimum Steiner tree has cost $d+q$, and think of $q = o(\sqrt{d})$ (otherwise, any off-the-shelf constant approximation algorithm for the Steiner problem gives a solution of cost $\leq d + O(q^2)$). We fix T to be some optimal Steiner tree. By optimality, all leaves in T must be terminals, whereas the internal nodes of T may be either terminals or non-terminals (non-terminals are called *Steiner nodes*). We define a coordinate i to be *good* if exactly one edge in T is labeled i, and *bad* if two or more edges in T are labeled with i. We may assume all d coordinates appear in the tree, otherwise, some coordinates in C are fixed and so the dimensionality of the problem is less than d. Therefore, at most q coordinates are bad (each bad coordinate flips at least twice and thus adds a cost of at least 2 to the tree).

Given a coordinate i of a set of terminals P, we define an *i-cut* as the partition $P_0 = \{x \in P : \ x_i = 0\}$ and $P_1 = \{x \in P : \ x_i = 1\}$. We call two coordinates $i \neq j$ *interchangeable* if they define the same cut. We now present the following basic facts which are easy to verify (see [2] for proofs).

Fact 1

1. *Let S be a set of interchangeable coordinates. Then all coordinates in S appear together in the optimal tree T, adjacent to one another. That is, in T there are paths s.t. for each path: all of its edges are labeled by some $i \in S$, all coordinates in S have an edge on the path, and all internal nodes on the paths aren't terminals and have degree 2. On these paths, any reordering the S-labeled edges yields an equivalent optimal tree.*
2. *For any two good coordinates, $i \neq j$, one side of the i-cut is contained within one side of the j-cut. Equivalently, there exist values b_j such that all terminals on one side of the i-cut have their jth coordinate set to b_j.*
3. *Fix any good coordinate i and let j be a good coordinate such that all terminals on one side of the i-cut have their j coordinate set to b_j. Then both endpoints of the edge labeled i have their jth coordinate set to b_j.*
4. *A good coordinate i and a bad coordinate i' cannot define the same cut.*

It immediately follows from Fact 1 that for a given good coordinate i one can efficiently reconstruct the endpoints of the edge on which i mutates, except for at most q coordinates. This leads us to the following definition. Given i, we denote D^i as the set of all coordinates that are fixed to a constant value v_i on at least one side of the i-cut (different coordinates may be fixed on different sides), and we denote $\mathbf{b}^i$ as the vector of the corresponding values, i.e. v_i's, of the coordinates in D^i. The pair $(D^i, \mathbf{b}^i)$ is called the *pattern* of coordinate i. That set of terminals that *match the pattern* of i is the set $P_{\mathbf{b}^i} = \{x \in P : \ \forall j \in D^i, \ x_j = b^i_j\}$.

3 A Simple Case: Each Coordinate Determines a Distinct Cut

To show the main ideas behind our algorithm, we first discuss a special case in which no two coordinates i and j define the same cut on the terminal set C. Algorithms for constructing phylogenetic trees often make this assumption as they preprocess C by contracting any pair of interchangeable coordinates. However, in our case such contractions are problematic, as we discuss in the next section. So in Section 4, when we deal with the general case, we deal with interchangeable coordinates in a non-trivial fashion.

3.1 Basic Building Blocks

We now turn to the description of our algorithm. On a high level it is motivated by the notion of maintaining a proper partition of the terminals.

Definition 1. *Call a partition* $\mathcal{P}$ proper *if the forest produced by restricting the optimal tree* T *to the components* $P \in \mathcal{P}$ *is composed of edge disjoint trees.*

Equivalently, the path in T between two nodes x and y in the same component P of $\mathcal{P}$ does not pass through any node x' in any different component P' of $\mathcal{P}$. Clearly, our initial partition, $\mathcal{P} = \{C\}$, is proper. Our goal is to maintain a proper partition of the current terminals while decreasing the dimensionality of the problem in each step. This is implemented by the two subroutines we now detail.

Pluck a Leaf and Paste a Leaf. The first subroutine works by building the optimal phylogeny bottom-up, finding a good coordinate i adjacent to a leaf terminal t in the tree, and replacing t with its parent (t with i flipped) in the set of terminals. Observe that if i is a good coordinate, then this removes the only occurrence of i, leaving all terminals in our new dataset with a fixed i coordinate, thus reducing the dimensionality of the problem by 1.

The matching subroutine to `Pluck-a-leaf` is `Paste-a-leaf`: if `Pluck-a-leaf` succeeds and returns some $(x, \mathcal{P}')$, and we have found a Steiner forest for the terminals in $\mathcal{P}'$. Then `Paste-a-leaf` merely connects x with $\bar{x}^i$ by an edge labeled i, then returns the resulting forest. (We omit formal description.)

`Pluck-a-leaf`
input: A partition $\mathcal{P}$ of current terminals.
if there exists $P \in \mathcal{P}$ and $x \in P$ s.t. some coordinate i is non-constant on P, but only the terminal x has $x_i = 0$ (or $x_i = 1$), then:
- Set $P' = P \setminus \{x\} \cup \{\bar{x}^i\}$, where $\bar{x}^i$ is identical to x except for flipping i.
- Return x and $\mathcal{P}' = \mathcal{P} \setminus \{P\} \cup \{P'\}$.

else fail.

Lemma 1. *If $\mathcal{P}$ is a proper partition and* `Pluck-a-leaf` *succeeds, then $\mathcal{P}'$ is a proper partition.*

Proof (Sketch). Let $T[P]$ be the subtree in which x resides. We claim that x is a leaf in $T[P]$, attached by an edge labeled i to the rest of the terminals. If this indeed is the case, then removing i means removing a leaf-adjacent edge from $T[P]$ which clearly leaves all components in the forest edge-disjoint.

Wlog x lies on the $i = 0$ side of the cut. If x isn't a leaf, then at least two disjoint paths connect x to two other terminals. Since $\mathcal{P}$ is proper, both these terminals are in P. This means $T[P]$ crosses the i-cut twice, but then we can replace $T[P]$ with an even less costly tree in which i is flipped once, by projecting the path between the two occurrences of i onto the $i = 1$ side. □

Observe that lemma 1 holds only when the underlying alphabet of the problem is binary. In particular, for a non-binary alphabet, such x can be a non-leaf.

Split and Merge. When `Pluck-a-leaf` can no longer find leaves to pluck, we switch to the second subroutine, one that works by splitting the set of terminals into two disjoint sets, based on the value of the i-th coordinate. We would like to split our set of terminals according to the i-cut, and recurse on each side separately. But, in order to properly reconnect the two subproblems, we need to introduce the two endpoints of the i-labeled edge to their respective sides of the i-cut. Our `Split` subroutine deals with one particular case in which these endpoints are easily identified.

`Split`(i)
input: A partition $\mathcal{P}$ of current terminals, a coordinate i that is not constant on every component of $\mathcal{P}$.

- Find a component P on which i isn't constant. Denote the i-cut of P as (P_0, P_1).
- Find $P_{\mathbf{b}^i}$, the set of terminals that match the pattern of i.
- **if** exists some x which is the *unique* terminal that matches the pattern of i in one side of the cut (that is, if for some x we have $P_{\mathbf{b}^i} \cap P_0 = \{x\}$ or $P_{\mathbf{b}^i} \cap P_1 = \{x\}$)
  - Flip the i-th coordinate of x, and let $\bar{x}^i$ be the resulting node.
  - Add x to its side of the i-cut, add $\bar{x}^i$ to the other side of the cut.
  - Return $x, \bar{x}^i$ and $\mathcal{P}' = \mathcal{P} \setminus \{P\} \cup \{P_0, P_1\}$.

  else fail.

The matching subroutine to `Split` is `Merge`: Assume `Split` succeeds and returns some $(x, \bar{x}^i, \mathcal{P}')$, and assume we have found a Steiner forest for the terminals in $\mathcal{P}'$. Then `Merge` merely connects x with $\bar{x}^i$ by an edge labeled i, then returns the resulting forest. (Again, formal description is omitted.)

Lemma 2. *Assume $\mathcal{P}$ is a proper partition. Assume* `Split` *is called on a good coordinate i s.t. the edge labeled i in T has at least one endpoint which is a terminal. Then the returned partition $\mathcal{P}'$ is proper.*

Proof (Sketch). Since $\mathcal{P}$ is proper, then the induced tree $T[P]$ is the only tree in the forest that contains the i-labeled edge. The lemma then follows from showing that x and $\bar{x}^i$ are the two endpoints of i-labeled edge in $T[P]$. This follows from the observation that the endpoints of the i-labeled edge must both match the pattern of i. Let u be an endpoint and wlog u belongs to the $(i = 0)$-side of the cut. On all coordinates that are fixed on the $(i = 0)$-side, u obviously has the right values. All coordinates that are fixed on the $(i = 1)$-side can only flip on the $(i = 0)$-side, but only after traversing u, so u has them set to the value fixed on the $(i = 1)$-side. □

3.2 The Algorithm

We can now introduce our algorithm.

input: A partition $\mathcal{P}$ of current terminals. Initially, $\mathcal{P}$ is the singleton set $\mathcal{P} = \{C\}$.

1. **if** `Pluck-a-leaf` succeeds and returns $(x, \mathcal{P}')$
   – recurse on $\mathcal{P}$', then `Paste-a-leaf` x back and return the resulting forest.
2. **else-if** the number of non-constant coordinates on $\mathcal{P}$ is at least $40q^2$
   – Pick a non-constant coordinate i u.a.r and invoke `Split`(i) .
   – **if** `Split` succeeds: recurse on $\mathcal{P}'$, then `Merge` x and $\bar{x}^i$, and return the resulting forest; **otherwise** fail.
3. **else**
   – For every $P \in \mathcal{P}$ find its MST, $T(P)$, and return the forest $\{T(P)\}$.

Fig. 1. Algorithm for the simple case

Theorem 2. *With probability $\geq 1/2$, the algorithm in Figure 1 returns a tree whose cost is at most $d + O(q^2)$.*

In order to prove Theorem 2, fix an optimal phylogeny T over our initial set of terminals, and for any partition $\mathcal{P}$ our algorithm creates, denote $T[\mathcal{P}]$ as the forest induced by T on this partition. The proof of the theorem relies on the following lemma.

Lemma 3. *If $\mathcal{P}$ is a proper partition, then with probability $\geq 1 - (8q)^{-1}$,* `Split` *is called on a good coordinate and succeeds. Furthermore,* `Split` *is executed at most $4q$ times.*

Proof (of Theorem 2). The proof follows from lemmas 1 and 3. Since we start with a proper partition, then with probability at least $1-(4q)(8q)^{-1} \geq 1/2$ we keep recursing on proper partitions, until reaching the base of the recursion. By the time the algorithm reaches the base of the recursion, the dimensionality of the problem was reduced to $d' \leq 40q^2$, so the cost of the optimal Steiner forest is at most $d'+q$. As MSTs give a 2-approximation to the optimal Steiner tree problem, our forest is of cost $\leq 2(d'+q)$. Then, the algorithm reconnects the forest, adding the coordinates (edges) the algorithm as removed in the first two steps of the algorithm. Since the algorithm removed at most $d-d'$ edges, the tree it outputs is of overall cost at most $d-d'+2(d'+q) = d+40q^2+2q$. □

Proof (of Lemma 3). Let $\mathcal{P}$ be the partition in the first iteration of the algorithm for which `Split` was invoked, and assume $\mathcal{P}$ is proper. Thus, the forest $T[\mathcal{P}]$ contains disjoint components. We call any vertex in this forest of degree ≥ 3 an *internal split*. Suppose we replace each internal split v with $deg(v)$ many new vertices, each adjacent to one edge. This breaks the forest into a collections of paths we call the *path decomposition* of the tree. In addition, remove from this path decomposition all edges that are labeled with a bad coordinate to obtain the *good path decomposition*. Denote the number of paths in the good path decomposition as t.

First, we claim that any call to `Split` (on $\mathcal{P}$ or any partition succeeding $\mathcal{P}$), on a coordinate i which lies on a path of length ≥ 2 in the abovementioned decomposition, does not fail.

Assume `Split` was called on i and denote its adjacent coordinate on the path as j (choose one arbitrarily if i has two adjacent coordinates on its path), and both are non-constant on $P \in \mathcal{P}$. Observe that our decomposition leaves only good coordinates, so both i and j are good. Therefore, j is fixed on one side of the i-cut and i is fixed on one side of the j-cut. It follows that there exist binary values b_i, b_j s.t. for every $x \in P$, if $x_i = b_i$ then $x_j = b_j$; and if $x_j = 1-b_j$ then $x_i = 1-b_i$. In fact, the only node on the entire tree for which $x_i = 1-b_i$ and $x_j = b_j$ is the node connecting the i-edge and the j-edge. Recall that we assume for the special case i and j do not define the same cut. It follows that the node between i and j has to be a terminal, so now we can use Lemma 2 and deduce `Split` succeeds.

So, `Split` can either fail or return a non-proper partition only if it was invoked either on a bad coordinate or on a good coordinate that lies on a path of length 1 in our path decomposition. There are at most q bad coordinates and at most t paths of length 1, so each call to `Split` fails w.p. $\leq \frac{q+t}{40q^2}$. Furthermore, calling `Split` on a good edge i lying on a path of length at least 2 results in both i's endpoints as new leaves in their respective sides of the i-cut. As a result, `Pluck-a-leaf` then completely unravels the path on which i lies. Therefore, in a successful run of the algorithm, `Split` is called no more than t times. All that remains is to bound t.

t is the number of paths on the path decomposition of $\mathcal{P}$, a partition for which `Pluck-a-leaf` failed to execute. Observe that if the forest $T[\mathcal{P}]$ had even a single leaf connected to the rest of its tree by a good coordinate, then `Pluck-a-leaf`

would continue – such a leaf, by definition, is the only terminal on which the good coordinate takes a certain value. It follows that l, the number of leaves in $T[\mathcal{P}]$ is bounded by $2q$, the number of bad edges in T. Removing the internal splits then leaves us with at most $2l$ paths; removing the bad coordinates' edges adds at most $2q - l$ new paths (for every bad coordinate k adjacent to a leaf, removing k does not create a new path). All in all, $t \leq 2l + 2q - l \leq 4q$. Therefore, each call to `Split` has success probability $\geq 1 - \frac{4q+q}{40q^2} = 1 - \frac{1}{8q}$, and `Split` is called at most $4q$ times. □

4 The General Case: Interchangeable Coordinates May Exist

Before describing the general case, let us briefly discuss why the conventional way of initially contracting all interchangeable coordinates and applying the algorithm from the Section 3 might result in a tree of cost $d+\omega(q^2)$. The analysis of the first two steps of the algorithm still holds. The problem lies in the base of the recursion, where the algorithm runs the MST-based 2-approximation. Indeed, the MST algorithm is invoked on $< 40q^2$ contracted coordinates, but they correspond to $\tilde{d}$ original coordinates, and it is possible that $\tilde{d} \gg q^2$. So by using any constant approximation on this entire forest, we may end with a tree of cost $d + 2\tilde{d}$ which isn't $d + O(q^2)$.

Our revised algorithm does not contract edges initially. Instead, let us define a *simple* coordinate as one for which `Split`(i) succeeds. So, the first alteration we make to the algorithm is to call `Split` as long as the set of simple coordinates is sufficiently big. However, most alterations lie in the base of the recursion. Below we detail the algorithm and analyze its correctness. In the algorithm's description, for any coordinate i we denote the set of coordinates interchangeable with i by W_i, and their number as $w(i) = |W(i)|$.

Theorem 3. *With probability $\geq 1/2$, the algorithm in Figure 2 returns a tree of cost $d + O(q^2)$.*

The proof of Theorem 3 follows the same outline as the proof of Theorem 2. Observe that Lemmas 1 and 3 still hold[2]. Therefore, with probability $\geq 1/2$, the algorithm enters the base of the recursion with a proper partition. Thus, by the following lemma, the algorithm outputs a tree of cost $d + O(q^2)$.

Lemma 4. *Assume that the base of the recursion (i.e., Step 3) is called on a proper partition $\mathcal{P}$ of the terminals over d' non-constant coordinates. Then the algorithm returns a forest of cost $d' + O(q^2)$.*

The full proof of Lemma 4 is deferred to the full version of the paper. However, let us sketch the main outline of the proof. Recall the good path decomposition we used in the proof of Lemma 3. We partition its paths in the following way.

[2] Clearly, `Split` cannot abort now, but it might be the case that the algorithm picks i which is a bad coordinate. This can happen with probability $\leq q/8q^2 = 1/8q$.

input: A partition $\mathcal{P}$ of current terminals. Initially, $\mathcal{P}$ is the singleton set $\mathcal{P} = \{C\}$.

1. **if** `Pluck-a-leaf` succeeds and returns $(x, \mathcal{P}')$
   - recurse on $\mathcal{P}$', then `Paste-a-leaf` x and return the resulting forest.
2. **else-if** the number of simple coordinates on $\mathcal{P}$ is at least $8q^2$
   - Pick a simple coordinate i u.a.r and invoke `Split`(i) .
   - **if** `Split` succeeds: recurse on $\mathcal{P}'$, then `Merge` x and $\bar{x}^i$, and return the resulting forest; **otherwise** fail.
3. **else**
   - Contract all W_i into $\bar{i}$
   - For every $\bar{i}$ with $w(i) > q$ and the (unique) component P in which the i-cut resides,
     - Apply pattern matching to (P, i). Let $(D^i, \mathbf{b}^i)$ be the pattern of i.
     - **if** i is simple, split P into $P_0 \cup \{x^i\}$ and $P_1 \cup \{\overline{x}^i\}$.
     - **else**
       * Define the node $y(i)$ as the node where $y_i = 0$, every coordinate $j \in D^i$ is set to b^i_j, and every coordinate $j \notin D^i$ is set to 0.
       * Define $\overline{y(i)}$ to be $y(i)$ with coordinate i flipped.
       * $\mathcal{P} = \mathcal{P} \setminus \{P\} \cup \{P_0 \cup \{y(i)\}\} \cup \{P_1 \cup \{\overline{y(i)}\}\}$.
   - For every $P \in \mathcal{P}$ find its MST $T(P)$, and retrieve the forest $\{T(P)\}$.
   - For every i with $w(i) > q$:
     if i was simple, add an edge labeled i between x^i and $\overline{x^i}$
     else add an edge labeled i between $y(i)$ and $\overline{y(i)}$.
   - Expand all contracted coordinates to their original set of coordinates by replacing i with a path of length $w(i)$. Return the resulting forest.

Fig. 2. Algorithm for the general case

- *Paths with at least one terminal on them.* On such paths, because all interchangeable coordinates may appear in T in any order, then all coordinates on such paths are simple. So, when we enter the base of the recursion, there are at most $8q^2$ edges on such paths.
- *Paths with no terminal on them, with length* $> q$. Such paths are composed of interchangeable coordinates, and since there are more than q of those, we deduce all of them are good. Therefore, the endpoints of such paths are fixed up to at most q (bad) coordinates. We therefore contract these edges, split on them, and introduce into each side of the cut an arbitrary endpoint, by replacing non-fixed coordinates with zeros. So on each side of the cut the cost of the subtree increases by at most q, and since there are at most $4q$ such paths, our overall cost for introducing these artificial endpoints is $O(q^2)$.
- *Paths with no terminal on them, with length* $\leq q$. Such paths are composed of interchangeable coordinates, but we do not contract them. Since there are at most $4q \cdot q$ edges on such paths, we run the MST approximation, and incur a cost of $O(q^2)$ for edges on such paths.

Runtime Analysis: `Pluck-a-leaf` can be implemented in time linear in the size of the dataset, i.e. $O(nd)$. Counting the number of simple coordinates takes time $O(nd^2)$, and `Split` takes time $O(nd)$. A naive implementation of the base

case of the recursion takes time $O(nd^2)$ for contracting coordinates, and the rest can be implemented in time $O(nd)$. Hence the time to process each node in the recursion tree is at most $O(nd^2)$. Since there are at most $O(q)$ nodes in the recursion tree, the total runtime is $O(qnd^2)$.

5 Discussion and Open Problems

This paper presents a randomized approximation algorithm for constructing near-perfect phylogenies. In order to achieve this, we obtain a Steiner tree of low additive error. However, from the biological perspective, the goal is to find a good evolutionary tree, one that will give correct answers to questions like "what is the common ancestor of the following species?" or "which of the two gene-mutations happened earlier?". Such questions, we hope, can be answered by finding the most-parsimonious phylogenetic tree over the given taxa. Hence, it is also desirable that any low-cost tree which we output also captures a lot of the structure of the optimal tree.

We would like to point out that our algorithm in fact has this valuable property. Notice that until the base case of the recursion, both `Pluck-a-leaf` and `Split` subroutines construct the optimal tree, and correctly identify the endpoints of the edges they remove. Even when the algorithm reaches the base case of the recursion – we can declare every edge of weight $> q$ to be good, and we know its endpoints up to at most q coordinates. In total, our algorithm gives the structure of the optimal tree up to $O(q^2)$ edges, and those edges can be marked as "unsure".

Several open problems remain for this work. The most straight-forward one is whether one can devise an algorithm outputting a phylogenetic tree of cost $d + O(q)$? Alternatively, one may try to design exact algorithms that are efficient even for $q = \omega(\log d)$. We suspect that even the case of $q = O((\log d)^2)$ poses quite a challenge. Finally, extending our results to non-binary alphabets is intriguing. Note however that even the case of perfect phylogenies is NP-hard, and tractable only for moderately sized alphabets. Furthermore, our bottom-up approach completely breaks down for non-binary alphabets (see comment past Lemma 1), so devising an additive-approximation algorithm for the phylogeny problem with non-binary alphabets requires a different approach altogether.

References

1. Ding, Z., Filkov, V., Gusfield, D.: A Linear-Time Algorithm for the Perfect Phylogeny Haplotyping (PPH) Problem. In: Miyano, S., Mesirov, J., Kasif, S., Istrail, S., Pevzner, P.A., Waterman, M. (eds.) RECOMB 2005. LNCS (LNBI), vol. 3500, pp. 585–600. Springer, Heidelberg (2005)
2. Blelloch, G.E., Dhamdhere, K., Halperin, E., Ravi, R., Schwartz, R., Sridhar, S.: Fixed Parameter Tractability of Binary Near-Perfect Phylogenetic Tree Reconstruction. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4051, pp. 667–678. Springer, Heidelberg (2006)
3. Gusfield, D.: Algorithms on strings, trees, and sequences: computer science and computational biology. Cambridge University Press (1997)
4. Semple, C., Steel, M.: Phylogenetics. Oxford lecture series in mathematics and its applications. Oxford University Press (2003)

5. Hinds, D.A., Stuve, L.L., Nilsen, G.B., Halperin, E., Eskin, E., Ballinger, D.G., Frazer, K.A., Cox, D.R.: Whole-genome patterns of common dna variation in three human populations. Science 307(5712), 1072–1079 (2005)
6. The international hapmap project. Nature 426(6968), 789–796 (2003)
7. Alon, N., Chor, B., Pardi, F., Rapoport, A.: Approximate maximum parsimony and ancestral maximum likelihood. IEEE/ACM Trans. Comput. Biol. Bioinformatics 7, 183–187 (2010)
8. Robins, G., Zelikovsky, A.: Improved steiner tree approximation in graphs. In: SODA, pp. 770–779. Society for Industrial and Applied Mathematics (2000)
9. Robins, G., Zelikovsky, A.: Improved steiner tree approximation in graphs (2000)
10. Robins, G., Zelikovsky, A.: Tighter bounds for graph steiner tree approximation. SIAM Journal on Discrete Mathematics 19, 122–134 (2005)
11. Misra, N., Blelloch, G., Ravi, R., Schwartz, R.: Generalized Buneman Pruning for Inferring the Most Parsimonious Multi-state Phylogeny. In: Berger, B. (ed.) RECOMB 2010. LNCS, vol. 6044, pp. 369–383. Springer, Heidelberg (2010)
12. Fernández-Baca, D., Lagergren, J.: A polynomial-time algorithm for near-perfect phylogeny. SIAM J. Comput. 32, 1115–1127 (2003)
13. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) Complexity of Computer Computations, pp. 85–103. Plenum, New York (1972)
14. Foulds, L.R., Graham, R.L.: The Steiner problem in phylogeny is NP-complete. Adv. Appl. Math. 3 (1982)
15. Sridhar, S., Dhamdhere, K., Blelloch, G., Halperin, E., Ravi, R., Schwartz, R.: Algorithms for efficient near-perfect phylogenetic tree reconstruction in theory and practice. IEEE/ACM Trans. Comput. Biol. Bioinformatics 4, 561–571 (2007)
16. Damaschke, P.: Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. Theor. Comput. Sci. 351, 337–350 (2006)
17. Agarwala, R., Fernandez-Baca, D.: A polynomial-time algorithm for the perfect phylogeny problem when the number of character states is fixed. In: SFCS, pp. 140–147 (November 1993)
18. Byrka, J., Grandoni, F., Rothvoß, T., Sanità, L.: An improved lp-based approximation for steiner tree. In: STOC. ACM (2010)
19. Bodlaender, H.L., Fellows, M.R., Warnow, T.: Two Strikes against Perfect Phylogeny. In: Kuich, W. (ed.) ICALP 1992. LNCS, vol. 623, pp. 273–283. Springer, Heidelberg (1992)
20. Takahashi, H., Matsuyama, A.: An approximate solution for the steiner problem in graphs. Mathematica Japonica 24, 573–577 (1980)
21. Berman, P., Ramaiyer, V.: Improved approximations for the steiner tree problem. In: SODA, pp. 325–334 (1992)
22. Prömel, H.J., Steger, A.: RNC-Approximation Algorithms for the Steiner Problem. In: Reischuk, R., Morvan, M. (eds.) STACS 1997. LNCS, vol. 1200, pp. 559–570. Springer, Heidelberg (1997)
23. Karpinski, M., Zelikovsky, A.: New approximation algorithms for the steiner tree problems. Journal of Combinatorial Optimization 1, 47–65 (1995)
24. Zelikovsky, A.: Better approximation bounds for the network and euclidean steiner tree problems. Technical report (1996)
25. Hougardy, S., Promel, H.J.: A 1.598 approximation algorithm for the steiner problem in graphs. In: SODA, pp. 448–453 (1999)
26. Borchers, A., Du, D.Z.: The k-steiner ratio in graphs. In: STOC, pp. 641–649. ACM (1995)

Improved Spectral-Norm Bounds for Clustering

Pranjal Awasthi and Or Sheffet*

Carnegie Mellon University,
5000 Forbes Ave., Pittsburgh PA 15213, USA
{pawasthi,osheffet}@cs.cmu.edu

Abstract. Aiming to unify known results about clustering mixtures of distributions under separation conditions, Kumar and Kannan [1] introduced a *deterministic* condition for clustering datasets. They showed that this single deterministic condition encompasses many previously studied clustering assumptions. More specifically, their *proximity condition* requires that in the target k-clustering, the projection of a point x onto the line joining its cluster center μ and some other center μ', is a large additive factor closer to μ than to μ'. This additive factor can be roughly described as k times the spectral norm of the matrix representing the differences between the given (known) dataset and the means of the (unknown) target clustering. Clearly, the proximity condition implies *center separation* – the distance between any two centers must be as large as the above mentioned bound.

In this paper we improve upon the work of Kumar and Kannan [1] along several axes. First, we weaken the center separation bound by a factor of $\sqrt{k}$, and secondly we weaken the proximity condition by a factor of k (in other words, the revised separation condition is independent of k). Using these weaker bounds we still achieve the same guarantees when all points satisfy the proximity condition. Under the same weaker bounds, we achieve *even better* guarantees when only $(1-\epsilon)$-fraction of the points satisfy the condition. Specifically, we correctly cluster all but a $(\epsilon + O(1/c^4))$-fraction of the points, compared to $O(k^2\epsilon)$-fraction of [1], which is meaningful even in the particular setting when ϵ is a constant and $k = \omega(1)$. Most importantly, we greatly simplify the analysis of Kumar and Kannan. In fact, in the bulk of our analysis we ignore the proximity condition and use only center separation, along with the simple triangle and Markov inequalities. Yet these basic tools suffice to produce a clustering which (i) is correct on all but a constant fraction of the points, (ii) has k-means cost comparable to the k-means cost of the target clustering, and (iii) has centers very close to the target centers.

Our improved separation condition allows us to match the results of the Planted Partition Model of McSherry [2], improve upon the results of Ostrovsky et al [3], and improve separation results for mixture of Gaussian models in a particular setting.

* This work was supported in part by the National Science Foundation under grants CCF-0830540, IIS-1065251, and CCF-1116892 as well as by CyLab at Carnegie Mellon under grants DAAD19-02-1-0389 and W911NF-09-1-0273 from the Army Research Office.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 37–49, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1 Introduction

In the long-studied field of clustering, there has been substantial work [4–12] studying the problem of clustering data from mixture of distributions under the assumption that the means of the distributions are sufficiently far apart. Each of these works focuses on one particular type (or family) of distribution, and devise an algorithm that successfully clusters datasets that come from that particular type. Typically, they show that w.h.p. such datasets have certain nice properties, then use these properties in the construction of the clustering algorithm.

The recent work of Kumar and Kannan [1] takes the opposite approach. First, they define a separation condition, deterministic and thus not tied to any distribution, and show that any set of data points satisfying this condition can be successfully clustered. Having established that, they show that many previously studied clustering problems indeed satisfy (w.h.p) this separation condition. These clustering problems include Gaussian mixture-models, the Planted Partition model of McSherry [2] and the work of Ostrovsky et al [3]. In this aspect they aim to unify the existing body of work on clustering under separation assumptions, proving that one algorithm applies in multiple scenarios.[1]

However, the attempt to unify multiple clustering works is only successful in part. First, Kumar and Kannan's analysis is "wasteful" w.r.t the number of clusters k. Clearly, motivated by an underlying assumption that k is constant, their separation bound has linear dependence in k and their classification guarantee has quadratic dependence on k. As a result, Kumar and Kannan overshoot best known bounds for the Planted Partition Model and for mixture of Gaussians by a factor of $\sqrt{k}$. Similarly, the application to datasets considered by Ostrovsky et al only holds for constant k. Secondly, the analysis in Kumar-Kannan is far from simple – it relies on most points being "good", and requires multiple iterations of Lloyd steps before converging to good centers. Our work addresses these issues.

To formally define the separation condition of [1], we require some notation. Our input consists of n points in $\mathbb{R}^d$. We view our dataset as a $n \times d$ matrix, A, where each datapoint corresponds to a row A_i in this matrix. We assume the existence of a target partition, $T_1, T_2, \ldots, T_k$, where each cluster's center is $\mu_r = \frac{1}{n_r} \sum_{i \in T_r} A_i$, where $n_r = |T_r|$. Thus, the target clustering is represented by a $n \times d$ matrix of cluster centers, C, where $C_i = \mu_r$ iff $i \in T_r$. Therefore, the k-means cost of this partition is the squared Frobenius norm $\|A - C\|_F^2$, but the focus of this paper is on the spectral (L_2) norm of the matrix $A - C$. Indeed, the deterministic equivalent of the maximal variance in any direction is, by definition, $\frac{1}{n}\|A - C\|^2 = \max_{\{v:\ \|v\|=1\}} \frac{1}{n}\|(A - C)v\|^2$.

Definition 1. *Fix $i \in T_r$. We say a datapoint A_i satisfies the* Kumar-Kannan proximity condition *if for any $s \neq r$, when projecting A_i onto the line connecting*

[1] We comment that, implicitly, Achlioptas and McSherry [8] follow a similar approach, yet they focus only on mixtures of Gaussians and log-concave distributions. Another deterministic condition for clustering was considered by [13], which generalized the Planted Partition Model of [2].

μ_r and μ_s, the projection of A_i is closer to μ_r than to μ_s by an additive factor of $\Omega\left(k(\frac{1}{\sqrt{n_r}}+\frac{1}{\sqrt{n_s}})\|A-C\|\right)$.

Kumar and Kannan proved that if all but at most ϵ-fraction of the data points satisfy the proximity condition, they can find a clustering which is correct on all but an $O(k^2\epsilon)$-fraction of the points. In particular, when $\epsilon = 0$, their algorithm clusters all points correctly. Observe, the Kumar-Kannan proximity condition gives that the distance $\|\mu_r - \mu_s\|$ is also bigger than the above mentioned bound. The opposite also holds – one can show that if $\|\mu_r - \mu_s\|$ is greater than this bound then only few of the points do not satisfy the proximity condition.

1.1 Our Contribution

Our Separation Condition. In this work, the bulk of our analysis is based on the following quantitatively weaker version of the proximity condition, which we call *center separation*. Formally, we define $\Delta_r = \frac{1}{\sqrt{n_r}} \min\{\sqrt{k}\|A-C\|, \|A-C\|_F\}$ and we assume throughout the paper that for a large constant[2] c we have that the means of any two clusters T_r and T_s satisfy

$$\|\mu_r - \mu_s\| \geq c(\Delta_r + \Delta_s) \tag{1}$$

Observe that this is a simpler version of the Kumar-Kannan proximity condition, scaled down by a factor of $\sqrt{k}$. Even though we show that (1) gives that only a few points do not satisfy the proximity condition, our analysis (for the most part) does not partition the dataset into good and bad points, based on satisfying or non-satisfying the proximity condition. Instead, our analysis relies on basic tools, such as the Markov inequality and the triangle inequality. In that sense one can view our work as "aligning" Kumar and Kannan's work with the rest of clustering-under-center-separation literature – we show that the bulk of Kannan and Kumar's analysis can be simplified to rely merely on center-separation.

Our Results. We improve upon the results of [1] along several axes. In addition to the weaker condition of Equation (1), we also weaken the Kumar-Kannan proximity condition by a factor of k, and still retrieve the target clustering, if all points satisfy the (k-weaker) proximity condition. Secondly, if at most ϵn points do not satisfy the k-weaker proximity condition, we show that we can correctly classify all but a $(\epsilon + O(1/c^4))$-fraction of the points, improving over the bound of [1] of $O(k^2\epsilon)$. Note that our bound is meaningful even if ϵ is a

[2] We comment that throughout the paper, and much like Kumar and Kannan, we think of c as a large constant ($c = 100$ will do). However, our results also hold when $c = \omega(1)$, allowing for a $(1+o(1))$-approximation. We also comment that we think of $d \gg k$, so one should expect $\|A-C\|_F^2 \geq k\|A-C\|^2$ to hold, thus the reader should think of Δ_r as dependent on $\sqrt{k}\|A-C\|$. Still, including the degenerate case, where $\|A-C\|_F^2 < k\|A-C\|$, simplifies our analysis in Section 3. One final comment is that (much like all the work in this field) we assume k is given, as part of the input, and not unknown.

constant whereas $k = \omega(1)$. Furthermore, we prove that the k-means cost of the clustering we output is a $(1 + O(1/c))$-approximation of the k-means cost of the target clustering.

Once we have improved on the main theorem of Kumar and Kannan, we derive immediate improvements on its applications. In Section 3.1 we show our analysis subsumes the work of Ostrovsky et al [3], and applies also to non-constant k. Using the fact that Equation (1) "shaves off" a $\sqrt{k}$ factor from the separation condition of Kumar and Kannan, we obtain a separation condition of $\Omega(\sigma_{\max}\sqrt{k})$ for learning a mixture of Gaussians, and we also match the separation results of the Planted Partition model of McSherry [2]. These results are detailed in the full version [14] of this paper.

From an approximation-algorithms perspective, it is clear why the case of $k = \omega(1)$ is of interest, considering the ubiquity of k-partition problems in TCS (e.g., k-Median, Max k-coverage, Knapsack for k items, maximizing social welfare in k-items auction – all trivially simple for constant k). In addition, we comment that in our setting only the case where $k = \omega(1)$ is of interest, since otherwise one can approximate the k-means cost using the PTAS of Kumar et al [15], which doesn't even require any separation assumptions. From a practical point of view, there is a variety of applications where k is quite large. This includes problems such as clustering images by who is in them, clustering protein sequences by families of organisms, and problems such as deduplication where multiple databases are combined and entries corresponding to the same true entity are to be clustered together [16, 17]. The challenges that arise from treating k as a non-constant are detailed in the proofs overview (Section 1.4).

To formally detail our results, we first define some notations and discuss a few preliminary facts.

1.2 Notations and Preliminaries

The Frobenius norm of a $n \times m$ matrix M, denoted as $\|M\|_F$ is defined as $\|M\|_F = \sqrt{\sum_{i,j} M_{i,j}^2}$, and the spectral norm of M is $\|M\| = \max_{x:\|x\|=1} \|Mx\|$. It is a well known fact that if the rank of M is t, then $\|M\|_F^2 \leq t\|M\|^2$. The Singular Value Decomposition (SVD) of M is the decomposition of M as $M = U\Sigma V^T$, where U is a $n \times n$ unitary matrix, V is a $m \times m$ unitary matrix, Σ is a $n \times m$ diagonal matrix whose entries are nonnegative real numbers, and its diagonal entries satisfy $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min\{m,n\}}$. The diagonal entries in Σ are called the singular values of M, and the columns of U and V, denoted u_i and v_i resp., are called the left- and right-singular vectors. As a convention, when referring to singular vectors, we mean the right-singular vectors. Observe that the Singular Value Decomposition allows us to write $M = \sum_{i=1}^{\text{rank}(\Sigma)} \sigma_i u_i v_i^T$. Projecting M onto its top t singular vectors means taking $\hat{M} = \sum_{i=1}^{t} \sigma_i u_i v_i^T$. It is a known fact that for any t, the t-dimensional subspace which best fits the rows of M, is obtained by projecting M onto the subspace spanned by the top t singular vectors (corresponding to the top t singular values). Another way to phrase this result is by saying that $\hat{M} = \arg\min_{N:\text{rank}(N)=t}\{\|M - N\|_F\}$. For a

proof, see [18]. The same matrix, $\hat{M}$, also minimizes the spectral norm of this difference, meaning $\hat{M} = \arg\min_{N:\mathrm{rank}(N)=t}\{\|M-N\|\}$ (see [19] for proof).

As previously defined, $\|A-C\|$ denotes the spectral norm of $A-C$. The target clustering, $\mathcal{T}$, is composed of k clusters $T_1, T_2, \ldots, T_k$. Observe that we use μ as an operator, where for every set X, we have $\mu(X) = \frac{1}{|X|}\sum_{i\in X} A_i$. We abbreviate, and denote $\mu_r = \mu(T_r)$. From this point on, we denote the projection of A onto the subspace spanned by its top k-singular vectors as $\hat{A}$, and for any vector v, we denote $\hat{v}$ as the projection of v onto this subspace. Throughout the paper, we abuse notation and use i to iterate over the rows of A, whereas r and s are used to iterate over *clusters* (or submatrices). So A_i represents the ith row of A whereas A_r represents the submatrix $[A_i]_{\{i\in T_r\}}$.

Basic Facts. The analysis of our main theorem makes use of the following facts, from [1, 2, 18]. The first fact bounds the cost of assigning the points of $\hat{A}$ to their original centers.

Fact 1 (Lemma 9 from [2]). $\|\hat{A}-C\|_F^2 \le 8\min\{k\|A-C\|^2, \|A-C\|_F^2\}$.

Next, we show that we can match each target center μ_r to a unique, relatively close, center ν_r that we get in Part I of the algorithm.

Fact 2 (Claim 1 in Section 3.2 of [18]). *For every μ_r there exists a center ν_s s.t. $\|\mu_r-\nu_s\| \le 6\Delta_r$, so we can match each μ_r to a unique ν_r.*

Finally, we exhibit the following fact, which is detailed in the analysis of [1].

Fact 3. *Fix a target cluster T_r and let S_r be a set of points created by removing $\rho_{out}n_r$ points from T_r and adding $\rho_{in}(s)n_r$ points from each cluster $s\neq r$, s.t. every added point x satisfies $\|x-\mu_s\| \ge \frac{2}{3}\|x-\mu_r\|$. Assume $\rho_{out} < \frac{1}{4}$ and $\rho_{in} \stackrel{\mathrm{def}}{=} \sum_{s\neq r}\rho_{in}(s) < \frac{1}{4}$. Then $\|\mu(S_r)-\mu_r\|$ is upper bounded by*

$$\frac{1}{\sqrt{n_r}}\left(\sqrt{\rho_{out}} + \tfrac{3}{2}\sum_{s\neq r}\sqrt{\rho_{in}(s)}\right)\|A-C\| \le \left(\sqrt{\frac{\rho_{out}}{n_r}} + \tfrac{3}{2}\sqrt{k}\sqrt{\frac{\rho_{in}}{n_r}}\right)\|A-C\|$$

1.3 Formal Description of the Algorithm and Our Theorems

Having established notation, we now present our algorithm, in Figure 1. Our algorithm's goal is three fold: (a) to find a partition that identifies with the target clustering on the majority of the points, (b) to have the k-means cost of this partition comparable with the target, and (c) output k centers which are close to the true centers. It is partitioned into 3 parts. Each part requires stronger assumptions, allowing us to prove stronger guarantees.

- Assuming only the center separation of (1), then **Part I** gives a clustering which (a) is correct on at least $1-O(c^{-2})$ fraction of the points *from each target cluster* (Theorem 1), and (b) has k-means cost smaller than $(1+O(1/c))\|A-C\|_F^2$ (Theorem 2).

Part I: Find initial centers:
- Project A onto the subspace spanned by the top k singular vectors.
- Run a 10-approximation algorithm[a] for the k-means problem on the projected matrix $\hat{A}$, and obtain k centers $\nu_1, \nu_2, \ldots, \nu_k$.

Part II: Set $S_r \leftarrow \{i : \|\hat{A}_i - \nu_r\| \leq \frac{1}{3}\|\hat{A}_i - \nu_s\|, \text{ for every } s\}$ and $\theta_r \leftarrow \mu(S_r)$.

Part III: Repeatedly run Lloyd steps until convergence.
- Set $\Theta_r \leftarrow \{i : \|A_i - \theta_r\| \leq \|A_i - \theta_s\|, \text{ for every } s\}$.
- Set $\theta_r = \mu(\Theta_r)$.

[a] Throughout the paper, we assume the use of a 10-approximation algorithm. Clearly, it is possible to use *any* t-approximation algorithm, assuming c/t is large enough.

Fig. 1. Algorithm $\sim$Cluster

- Assuming also that $\Delta_r = \frac{\sqrt{k}}{\sqrt{n_r}}\|A - C\|$, i.e. assuming the *non-degenerate* case where $\|A - C\|_F^2 \geq k\|A - C\|^2$, then **Part II** finds centers that are $O(1/c)\frac{\|A-C\|}{\sqrt{n_r}}$ close to the true centers (Theorem 3). As a result (see Section 4.1), if $(1-\epsilon)n$ points satisfy the proximity condition (weakened by a k factor,), then we misclassify no more than $(\epsilon + O(c^{-4}))n$ points.
- Assuming all points satisfy the proximity condition (weakened by a k-factor), **Part III** finds *exactly* the target partition (Theorem 4).

1.4 Organization and Proofs Overview

Organization. Related work is detailed in Section 2. The analysis of Part I of our algorithms is in Section 3. Part I is enough for us to give a "one-line" proof in Section 3.1 showing how the work of Ostrovsky et al falls into our framework. The analysis of Part II of the algorithm is in Section 4. The improved guarantees we get by applying the algorithm to the Planted Partition model and to the Gaussian mixture model are discussed in the full version of this paper [14].

Proof Outline for Section 3. The first part of our analysis is an immediate application of Facts 1 and 2. Our assumption dictates that the distance between any two centers is big ($\geq c(\Delta_r + \Delta_s)$). Part I of the algorithm assigns each projected point $\hat{A}_i$ to the nearest ν_r instead of the true center μ_r and Fact 2 assures that the distance $\|\mu_r - \nu_r\|$ is small ($< 6\Delta_r$). Consider a misclassified point A_i, where $\|A_i - \mu_r\| < \|A_i - \mu_s\|$ yet $\|\hat{A}_i - \nu_s\| < \|\hat{A}_i - \nu_r\|$. The triangle inequality assures that $\hat{A}_i$ has a fairly big distance to its true center ($> (\frac{c}{2} - 12)\Delta_r$). We deduce that each misclassified point contributes $\Omega(c^2\Delta_r^2)$ to the k-means cost of assigning all projected points to their true centers. Fact 1 bounds this cost by $\|\hat{A} - C\|_F^2 \leq 8n_r\Delta_r^2$, so the Markov inequality proves only a few points are misclassified. Additional application of the triangle inequality for misclassified points gives that the distance between the original point A_i and a true center

μ_r is comparable to the distance $\|A_i - \mu_s\|$, and so assigning A_i to the cluster s only increases the k-means cost by a small factor.

Proof Outline for Section 4. In the second part of our analysis we compare between the true clustering $\mathcal{T}$ and some proposed clustering $\mathcal{S}$, looking *both* at the number of misclassified points *and* at the distances between the matching centers $\|\mu_r - \theta_r\|$. As Kumar and Kannan show, the two measurements are related: Fact 3 shows how the distances between the means depend on the number of misclassified points, and the main lemma (Lemma 3) essentially shows the opposite direction. These two relations are how Kumar and Kannan show that Lloyd steps converge to good centers, yielding clusters with few misclassified points. They repeatedly apply (their version of) the main lemma, showing that with each step the distances to the true means decrease and so fewer of the good points are misclassified.

To improve on Kumar and Kannan analysis, we improve on the two above-mentioned relations. Lemma 3 is a simplification of a lemma from Kumar and Kannan, where instead of projecting into a k-dimensional space, we project only into a 4-dimensional space, thus reducing dependency on k. However, the dependency of Fact 3 on k is tight[3]. So in Part II of the algorithm we devise sub-clusters S_r s.t. $\rho_{in}(s) = \rho_{out}/k^2$. The crux in devising S_r lies in Proposition 1 – we show that any misclassified projected point $i \in T_s \cap S_r$ is essentially misclassified by $\hat{\mu_r}$. And since (see [8]) $\|\mu_r - \hat{\mu_r}\| \leq \frac{1}{\sqrt{k}}\Delta_r$ (compared to the bound $\|\mu_r - \nu_r\| \leq 6\Delta_r$), we are able to give a good bound on $\rho_{in}(s)$.

Recall that we rely only on center separation rather than a large batch of points satisfying the Kumar-Kannan separation, and so we do not apply iterative Lloyd steps (unless all points are good). Instead, we apply the main lemma only once, w.r.t to the misclassified points in $T_s \cap S_r$, and deduce that the distances $\|\mu_r - \theta_r\|$ are small. In other words, Part II is a single step that retrieve centers whose distances to the original centers are $\sqrt{k}$-times better than the centers retrieved by Kumar and Kannan in numerous Lloyd iterations.

2 Related Work

The work of [4] was the first to give theoretical guarantees for the problem of learning a mixture of Gaussians under separation conditions. He showed that one can learn a mixture of k spherical Gaussians provided that the separation between the cluster means is $\tilde{\Omega}(\sqrt{n}(\sigma_r + \sigma_s))$ and the mixing weights are not too small. Here σ_r^2 denotes the maximum variance of cluster r along any direction. This separation was improved to $\tilde{\Omega}((\sigma_r + \sigma_s)n^{1/4})$ by [5]. Arora and Kannan [6] extended these results to the case of general Gaussians. For the case of spherical Gaussians, [7] showed that one can learn under a much weaker separation of $\tilde{\Omega}((\sigma_r + \sigma_s)k^{1/4})$. This was extended to arbitrary Gaussians by [8] and to various other distributions by [10], although requiring a larger separation. In

[3] In fact, Fact 3 is exactly why the case of $k = \omega(1)$ is hard – because the L_1 and L_2 norms of the vector $(\frac{1}{\sqrt{k}}, \frac{1}{\sqrt{k}}, \ldots, \frac{1}{\sqrt{k}})$ are not comparable for non-constant k.

particular, the work of [8] requires a separation of $\Omega((\sigma_r+\sigma_s)(\frac{1}{\sqrt{\min(w_r,w_s)}}+\sqrt{k\log(k\min\{2^k,n\})}))$ whereas [10] require a separation of $\tilde{\Omega}(\frac{k^{3/2}}{w_{\min}^2}(\sigma_r+\sigma_s))$. Here w_r's refer to the mixing weights. [9, 20] gave algorithms for clustering mixtures of product distributions and mixtures of heavy tailed distributions. [12] gave an algorithm for clustering the mixture of 2 Gaussians assuming only that the two Gaussians are separated by a hyperplane. They also give results for learning a mixture of $k > 2$ Gaussians. The work of [21] gave an algorithm for learning a mixture of 2 Gaussians, with provably minimal assumptions. This was extended in [22] to the case when $k > 2$ although the algorithm runs in time exponential in k. Similar results were obtained in the work of [23] who can also learn more general distribution families. The work of [13] studied a deterministic separation condition required for efficient clustering. The precise condition presented in [13] is technical but essentially assumes that the underlying graph over the set of points has a "low rank structure" and presents an algorithm to recover this structure which is then enough to cluster well. In addition, previous works (e.g. [24, 25]) addressed the problem of clustering from the viewpoint of minimizing the number of mislabeled points.

3 Part I of the Algorithm

In this section, we look only at Part I of our algorithm. Our approximation algorithm defines a clustering $\mathcal{Z}$, where $Z_r = \{i : \|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \nu_s\| \text{ for every } s\}$. Our goal in this section is to show that $\mathcal{Z}$ is correct on all but a small constant fraction of the points, and furthermore, the k-means cost of $\mathcal{Z}$ is no more than $(1+O(1/c))$ times the k-means cost of the target clustering.

Theorem 1. *There exists a matching (given by Fact 2) between the target clustering $\mathcal{T}$ and the clustering $\mathcal{Z} = \{Z_r\}_r$ where $Z_r = \{i : \|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \nu_s\|$ for every $s\}$ that satisfies the following properties:*

- *For every cluster T_{s_0} in the target clustering, no more than $O(1/c^2)|T_{s_0}|$ points are misclassified.*
- *For every cluster Z_{r_0} in the clustering that the algorithm outputs, we add no more than $O(1/c^2)|T_{r_0}|$ points from other clusters.*
- *At most $O(1/c^2)|T_{r_2}|$ points are misclassified overall, where T_{r_2} is the second largest cluster.*

Proof. Let us denote $T_{s\to r}$ as the set of points $\hat{A}_i$ that are assigned to T_s in the target clustering, yet are closer to ν_r than to any other ν'_r. From triangle inequality we have that $\|\hat{A}_i - \mu_s\| \geq \|\hat{A}_i - \nu_s\| - \|\mu_s - \nu_s\|$. We know from Fact 2 that $\|\mu_s - \nu_s\| \leq 6\Delta_s$. Also, since $\hat{A}_i$ is closer to ν_r than to ν_s, the triangle inequality gives that $2\|\hat{A}_i - \nu_s\| \geq \|\nu_r - \nu_s|$. So,

$$\|\hat{A}_i - \mu_s\| \geq \frac{1}{2}\|\nu_r - \nu_s\| - 6\Delta_s \geq \frac{1}{2}\|\mu_r - \mu_s\| - 12(\Delta_r + \Delta_s) \geq \frac{c}{4}(\Delta_r + \Delta_s)$$

Thus, we can look at $\|\hat{A} - C\|_F^2$, and using Fact 1 we immediately have that for every fixed r'

$$\sum_r \sum_{s \neq r} |T_{s \to r}| \frac{c^2}{16} (\Delta_r + \Delta_s)^2 \leq \sum_r \sum_{i \in T_r} \|\hat{A}_i - \mu_r\|^2 = \|\hat{A} - C\|_F^2 \leq 8 n_{r'} \Delta_{r'}^2$$

The proof of the theorem follows from fixing some r_0, and deducing that $\Delta_{r_0}^2 \sum_{s \neq r_0} |T_{s \to r_0}| \leq \sum_{s \neq r_0} |T_{s \to r_0}| (\Delta_{r_0} + \Delta_s)^2 \leq \sum_r \sum_{s \neq r} |T_{s \to r}| (\Delta_r + \Delta_s)^2 \leq \frac{128}{c^2} n_{r_0} \Delta_{r_0}^2$. Alternatively, one can fix some s_0 and have that $\Delta_{s_0}^2 \sum_{r \neq s_0} |T_{s_0 \to r}| \leq \sum_{r \neq s_0} |T_{s_0 \to r}| (\Delta_r + \Delta_{s_0})^2 \leq \sum_r \sum_{s \neq r} |T_{s \to r}| (\Delta_r + \Delta_s)^2 \leq \frac{128}{c^2} n_{s_0} \Delta_{s_0}^2$. Observe that for every $r \neq s$ we have that $\Delta_r + \Delta_s \geq \Delta_{r_2}$ (where r_2 is the cluster with the second largest number of points), so we also have that $\Delta_{r_2}^2 \sum_r \sum_{s \neq r} |T_{s \to r}| \leq \sum_r \sum_{s \neq r} |T_{s \to r}| (\Delta_r + \Delta_s)^2 \leq \frac{128}{c^2} n_{r_2} \Delta_{r_2}^2$.

We now show that the k-means cost of $\mathcal{Z}$ is close to the k-means cost of $\mathcal{T}$. Observe that the k-means cost of $\mathcal{Z}$ is computed w.r.t the best center of each cluster (i.e., $\mu(Z_r)$), and *not* w.r.t the centers ν_r.

Theorem 2. *The k-means cost of $\mathcal{Z}$ is at most $(1 + O(1/c))\|A - C\|_F^2$.*

Proof. Given $\mathcal{Z}$, it is clear that the centers that minimize its k-means cost are $\mu(Z_r) = \frac{1}{|Z_r|} \sum_{i \in Z_r} A_i$. Recall that the majority of points in each Z_r belong to a unique T_r, and so, throughout this section, we assume that all points in Z_r were assigned to μ_r, and not to $\mu(Z_r)$. (Clearly, this can only increase the cost.) We show that by assigning the points of Z_r to μ_r, our cost is at most $(1 + O(1/c))\|A - C\|_F^2$, and so Theorem 2 follows. In fact, we show something stronger. We show that by assigning all the points in Z_r to μ_r, each point A_i pays no more than $(1 + O(1/c))\|A_i - C_i\|^2$. This is clearly true for all the points in $Z_r \cap T_r$. We show this also holds for the misclassified points.

Because $i \in T_{s \to r}$, it holds that $\|\hat{A}_i - \nu_r\| \leq \|\hat{A}_i - \nu_s\|$. Observe that for every s we have that $\|A_i - \nu_s\|^2 = \|A_i - \hat{A}_i\|^2 + \|\hat{A}_i - \nu_s\|^2$, because $\hat{A}_i - \nu_s$ is the projection of $A_i - \nu_s$ onto the subspace spanned by the top k-singular vectors of A. Therefore, it is also true that $\|A_i - \nu_r\| \leq \|A_i - \nu_s\|$. Because of Fact 2, we have that $\|\mu_r - \nu_r\| \leq 6\Delta_r$ and $\|\mu_s - \nu_s\| \leq 6\Delta_s$, so we apply the triangle inequality and get

$$\|A_i - \mu_r\| \leq \|A_i - \mu_s\| + \|\mu_r - \nu_r\| + \|\mu_s - \nu_s\| \leq \|A_i - \mu_s\| \left(1 + \frac{6(\Delta_r + \Delta_s)}{\|A_i - \mu_s\|}\right)$$

So all we need to do is to lower bound $\|A_i - \mu_s\|$. As noted, $\|A_i - \nu_s\| \geq \|\hat{A}_i - \nu_s\|$. Thus $\|A_i - \mu_s\| \geq \|A_i - \nu_s\| - 6\Delta_r \geq \|\hat{A}_i - \nu_s\| - 6\Delta_r \geq \frac{1}{2}\|\nu_s - \nu_r\| - 6\Delta_r \geq \frac{1}{4} c(\Delta_r + \Delta_s)$ and we have the bound $\|A_i - \mu_r\| \leq \left(1 + \frac{24}{c}\right)\|A_i - \mu_s\|$, so $\|A_i - \mu_r\|^2 \leq \left(1 + \frac{49}{c}\right)\|A_i - \mu_s\|^2$.

3.1 Application: The ORSS-Separation

One straight-forward application of Theorem 2 is for the datasets considered by Ostrovsky et al [3], where the optimal k-means cost is an ϵ-fraction of the

optimal $(k-1)$-means cost. Ostrovsky et al proved that for such datasets a variant of the Lloyd method converges to a good solution in polynomial time. Kumar and Kannan's non-trivial analysis shows that datasets satisfying the ORSS-separation also have the property that most points satisfy their proximity-condition, resulting in a $(1+O(\sqrt{k\epsilon}))$-approximation.

Here, we provide a "one-line" proof that Part I of Algorithm $\sim$Cluster yields a $(1+O(\sqrt{\epsilon}))$-approximation, for any k. Suppose we have a dataset satisfying the ORSS-separation condition, so any $(k-1)$-partition of the dataset have cost $\geq \frac{1}{\epsilon}\|A-C\|_F^2$. For any r and any $s \neq r$, by assigning all the points in T_r to the center μ_s, we get some $(k-1)$-partition whose cost is exactly $\|A-C\|_F^2 + n_r\|\mu_r - \mu_s\|^2$, so $\|\mu_r - \mu_s\| \geq \frac{\sqrt{\frac{1}{\epsilon}-1}}{\sqrt{n_r}}\|A-C\|_F$. Setting $c = O(1/\sqrt{\epsilon})$, Theorem 2 is immediate.

4 Part II of the Algorithm

In this section, our goal is to show that Part II of our algorithm gives centers that are very close to the target clusters. We should note that from this point on, we assume we are in the non-degenerate case, where $\|A-C\|_F^2 \geq k\|A-C\|^2$. Therefore, $\Delta_r = \frac{\sqrt{k}}{\sqrt{n_r}}\|A-C\|$. Due to space limitation, all proofs in this section are omitted and are deferred to the full version [14] of this paper.

Recall, in Part II we define the sets $S_r = \{i : \|\hat{A}_i - \nu_r\| \leq \frac{1}{3}\|\hat{A}_i - \nu_s\|,\ \forall s \neq r\}$. Observe, these set do not define a partition of the dataset! There are some points that are not assigned to any S_r. However, we only use the centers of S_r.

Theorem 3. *Denote $S_r = \{i : \|\hat{A}_i - \nu_r\| \leq \frac{1}{3}\|\hat{A}_i - \nu_s\|,\ \forall s \neq r\}$. Then for every r it holds that $\|\mu(S_r) - \mu_r\| = O(1/c)\ \frac{1}{\sqrt{n_r}}\|A-C\| = O(\frac{1}{c\sqrt{k}}\Delta_r)$.*

The proof of Theorem 3 is an immediate application of Fact 3 combined with the following two lemmas, that bound the number of misclassified points. Observe that for every point that belongs to T_s yet is assigned to S_r (for $s \neq r$) is also assigned to Z_r in the clustering $\mathcal{Z}$ discussed in the previous section. Therefore, any misclassified point $i \in T_s \cap S_r$ satisfies that $\|A_i - \mu_r\| \leq (1+O(c^{-1}))\|A_i - \mu_s\|$ as the proof of Theorem 2 shows. So all conditions of Fact 3 hold.

Lemma 1. *Assume that for every r we have that $\|\mu_r - \nu_r\| \leq 6\Delta_r$. Then at most $\frac{512}{c^2}n_r$ points of T_r do not belong to S_r.*

Lemma 2. *Redefine $T_{s\to r}$ as the set $T_s \cap S_r$. Assume that for every r we have that $\|\mu_r - \nu_r\| \leq 6\Delta_r$. Then $\forall r, s \neq r$ we have that $|T_{s\to r}| = \left(\frac{48^2}{c^4k^2}\right)n_r$.*

We now turn to proving Lemma 2. Proposition 1 exhibit some property that every point in $T_{s\to r}$ must satisfy, and then we show that only few of the points in T_s satisfy this property. Recall that $\hat{\mu_r}$ indicates the projection of μ_r onto the subspace spanned by the top k-singular vectors of A.

Proposition 1. *Fix $i \in T_s$ s.t. $\|\hat{A}_i - \hat{\mu}_s\| \leq 2\|\hat{A}_i - \hat{\mu}_r\|$. Then $\|\hat{A}_i - \nu_s\| < 3\|\hat{A}_i - \nu_r\|$, so $i \notin S_r$.*

Proposition 1 shows that in order to bound $|T_{s\to r}|$ it suffices to bound the number of points in T_s satisfying $\|\hat{A}_i - \hat{\mu}_s\| \geq 2\|\hat{A}_i - \hat{\mu}_r\|$. The major tool in providing this bound is the following technical lemma. This lemma is a variation on the work of [1], on which we improve on the dependency on k and simplify the proof. Following Lemma 3, the proof of Lemma 2 is fairly straight-forward.

Lemma 3 (Main Lemma). *Fix $\alpha, \beta > 0$. Fix $r \neq s$ and let ζ_r and ζ_s be two points s.t. $\|\mu_r - \zeta_r\| \leq \alpha\Delta_r$ and $\|\mu_s - \zeta_s\| \leq \alpha\Delta_s$. We denote $\tilde{A}_i$ as the projection of A_i onto the line connecting ζ_r and ζ_s. Define $X = \{i \in T_s : \|\tilde{A}_i - \zeta_s\| - \|\tilde{A}_i - \zeta_r\| \geq \beta\|\zeta_s - \zeta_r\|\}$. Then $|X| \leq 256\frac{\alpha^2}{\beta^2}\frac{1}{c^4 k}\left(\min\{n_r, n_s\}\right)$.*

4.1 The Proximity Condition – Part III of the Algorithm

Part II of our algorithm returns centers $\theta_1, \ldots, \theta_k$ which are $O(\frac{1}{c\sqrt{n_r}})\|A - C\|$ close to the true centers. Suppose we use these centers to cluster the points: $\Theta_s = \{i : \forall s',\ \|A_i - \theta_s\| \leq \|A_i - \theta_{s'}\|\}$. It is evident that this clustering correctly classifies the majority of the points. It correctly classifies any point $i \in T_s$ with $\|A_i - \mu_r\| - \|A_i - \mu_s\| = \Omega(\frac{1}{c\sqrt{n_r}})\|A - C\|$ for every $r \neq s$, and the analysis of Theorem 1 shows that at most $O(c^{-2})$-fraction of the points do not satisfy this condition. In order to have a direct comparison with the Kumar-Kannan analysis, we now bound the number of misclassified points w.r.t the fraction of points satisfying the Kumar-Kannan proximity condition.

Definition 2. *Denote $gap_{r,s} = (\frac{1}{\sqrt{n_r}} + \frac{1}{\sqrt{n_s}})\|A - C\|$. Call a point $i \in T_s$ γ-good, if for every $r \neq s$ we have that the projection of A_i onto the line connecting μ_r and μ_s, denoted $\bar{A}_i$, satisfies that $\|\bar{A}_i - \mu_r\| - \|\bar{A}_i - \mu_s\| \geq \gamma\ gap_{r,s}$; otherwise we say the point is* γ-bad.

Corollary 1. *Denote the fraction of γ-bad points as ϵ. Then (a) the clustering $\{\Theta_1, \ldots, \Theta_k\}$ misclassifies no more than $\left(\epsilon + \frac{O(1)}{\gamma^2 c^4}\right)n$ points, and (b) $\epsilon < O\left((c - \frac{\gamma}{\sqrt{k}})^{-2}\right)$, assuming $\gamma < c\sqrt{k}$.*

Observe that Corollary 1 allows for multiple scaled versions of the proximity condition, based on the magnitude of γ. In particular, setting $\gamma = 1$ we get a proximity condition whose bound is independent of k, and still our clustering misclassifies only a small fraction of the points – at most $O(c^{-2})$ fraction of all points might be misclassified because they are 1-bad, and no more than a $O(c^{-4})$-fraction of 1-good points may be misclassified. In addition, if there are no 1-bad points we show the following theorem. The proof (omitted) merely follows the Kumar-Kannan proof, plugging in the better bounds, provided by Lemma 3.

Theorem 4. *Assume all data points are* 1*-good. That is, for every point* A_i *that belongs to the target cluster* $T_{c(i)}$ *and every* $s \neq c(i)$*, by projecting* A_i *onto the line connecting* $\mu_{c(i)}$ *with* μ_s *we have that the projected point* $\bar{A}_i$ *satisfies* $\|\bar{A}_i - \mu_{c(i)}\| - \|\bar{A}_i - \mu_s\| = \Omega\left(\left(\frac{1}{\sqrt{n_{c(i)}}} + \frac{1}{\sqrt{n_s}}\right)\right)\|A - C\|$*, whereas* $\|\mu_{c(i)} - \mu_s\| = \Omega\left(\sqrt{k}\left(\frac{1}{\sqrt{n_{c(i)}}} + \frac{1}{\sqrt{n_s}}\right)\right)\|A - C\|$*. Then the Lloyd method, starting with* $\theta_1, \ldots, \theta_k$*, converges to the true centers.*

Acknowledgements. We would like to thanks Avrim Blum for multiple helpful discussions and suggestions. We thank Amit Kumar for clarifying a certain point in the original Kumar and Kannan paper. We thank the anonymous referees for their suggestions, and especially regarding a discussion about the result of Achlioptas and McSherry.

References

1. Kumar, A., Kannan, R.: Clustering with spectral norm and the k-means algorithm. In: FOCS (2010)
2. McSherry, F.: Spectral partitioning of random graphs. In: FOCS (2001)
3. Ostrovsky, R., Rabani, Y., Schulman, L.J., Swamy, C.: The effectiveness of lloyd-type methods for the k-means problem. In: FOCS, pp. 165–176 (2006)
4. Dasgupta, S.: Learning mixtures of gaussians. In: FOCS (1999)
5. Dasgupta, S., Schulman, L.: A probabilistic analysis of em for mixtures of separated, spherical gaussians. J. Mach. Learn. Res. (2007)
6. Sanjeev, A., Kannan, R.: Learning mixtures of arbitrary gaussians. In: STOC (2001)
7. Vempala, S., Wang, G.: A spectral algorithm for learning mixtures of distributions. Journal of Computer and System Sciences (2002)
8. Achlioptas, D., McSherry, F.: On Spectral Learning of Mixtures of Distributions. In: Auer, P., Meir, R. (eds.) COLT 2005. LNCS (LNAI), vol. 3559, pp. 458–469. Springer, Heidelberg (2005)
9. Chaudhuri, K., Rao, S.: Learning mixtures of product distributions using correlations and independence. In: COLT (2008)
10. Kannan, R., Salmasian, H., Vempala, S.: The spectral method for general mixture models. SIAM J. Comput. (2008)
11. Dasgupta, A., Hopcroft, J., Kannan, R., Mitra, P.: Spectral clustering with limited independence. In: SODA (2007)
12. Brubaker, S.C., Vempala, S.: Isotropic pca and affine-invariant clustering. In: FOCS (2008)
13. Coja-Oghlan, A.: Graph partitioning via adaptive spectral techniques. Comb. Probab. Comput. 19, 227–284 (2010)
14. Awasthi, P., Sheffet, O.: Improved spectral-norm bounds for clustering, full version (2012), http://arxiv.org/abs/1206.3204
15. Kumar, A., Sabharwal, Y., Sen, S.: A simple linear time $(1 + \epsilon)$-approximation algorithm for k-means clustering in any dimensions. In: FOCS (2004)
16. Cohen, W.W., Richman, J.: Learning to match and cluster large high-dimensional data sets for data integration. In: KDD, pp. 475–480 (2002)

17. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: Scop: a structural classification of proteins database for the investigation of sequences and structures. Journal of Molecular Biology 247(4), 536–540 (1995)
18. Kannan, R., Vempala, S.: Spectral algorithms. Found. Trends Theor. Comput. Sci. (March 2009)
19. Golub, G.H., Van Loan, C.F.: Matrix computations, 3rd edn. Johns Hopkins University Press, Baltimore (1996)
20. Chaudhuri, K., Rao, S.: Beyond gaussians: Spectral methods for learning mixtures of heavy-tailed distributions. In: COLT (2008)
21. Kalai, A.T., Moitra, A., Valiant, G.: Efficiently learning mixtures of two gaussians. In: STOC 2010, pp. 553–562 (2010)
22. Moitra, A., Valiant, G.: Settling the polynomial learnability of mixtures of gaussians. In: FOCS 2010 (2010)
23. Belkin, M., Sinha, K.: Polynomial learning of distribution families. Computing Research Repository abs/1004.4, 103–112 (2010)
24. Schulman, L.J.: Clustering for edge-cost minimization (extended abstract). In: STOC, pp. 547–555 (2000)
25. Balcan, M.F., Blum, A., Gupta, A.: Approximate clustering without the approximation. In: SODA, pp. 1068–1077 (2009)

Primal-Dual Approximation Algorithms for Node-Weighted Network Design in Planar Graphs*

Piotr Berman and Grigory Yaroslavtsev

Pennsylvania State University, USA
{berman,grigory}@cse.psu.edu

Abstract. We present primal-dual algorithms which give a 2.4 approximation for a class of node-weighted network design problems in planar graphs, introduced by Demaine, Hajiaghayi and Klein (ICALP'09). This class includes Node-Weighted Steiner Forest problem studied recently by Moldenhauer (ICALP'11) and other node-weighted problems in planar graphs that can be expressed using $(0, 1)$-proper functions introduced by Goemans and Williamson. We show that these problems can be equivalently formulated as feedback vertex set problems and analyze approximation factors guaranteed by different violation oracles within the primal-dual framework developed by Goemans and Williamson.

1 Introduction

In feedback vertex set problems the input is a graph $G = (V, E)$, a family of cycles $\mathcal{C}$ in G and a function $w\colon V \to \mathbb{R}^{\geq 0}$. The goal is to find a set of vertices $H \subset V$ which contains a node in every cycle in $\mathcal{C}$ such that the total weight of vertices in H is minimized. This is a special case of the hitting set problem, where sets correspond to the cycles of $\mathcal{C}$. There five natural examples for the family $\mathcal{C}$.

- All cycles. This is Feedback Vertex Set problem (FVS).
- Odd cycles. If $H \subset V$ is a hitting set for all odd-length cycles then the subgraph of G, induced by the vertex set $V \setminus H$ is bipartite. This is Bipartization problem (BIP).
- The set of all cycles which contain at least one node from a given set of nodes. This is Subset Feedback Vertex Set problem (S-FVS).
- The set of all directed cycles of a given directed graph. This is Directed Feedback Vertex Set problem (D-FVS).
- In Node-Weighted Steiner Forest problem we are given a weighted graph and a set of terminal pairs (s_i, t_i). The goal is to select $S \subset V$ such that in the subgraph induced by S all terminal pairs are connected. In Section 2.1 we show that Node-Weighted Steiner Forest belongs to a class of problems which can be expresed as a hitting set problem for an appropriately defined collection of cycles.

* G.Y. is supported by NSF / CCF CAREER award 0845701 and by College of Engineering Fellowship.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 50–60, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Table 1. Planar graphs

Problem	Previous work (our analysis)[1]	Our work	Hardness
FVS	10 [3], 3 (18/7) [11], 2 [4,1]	2.4	NP-hard [20]
BIP, D-FVS, S-FVS	3 (18/7) [11]		
Node-Weighted Steiner Forest	6 [5], 3 (18/7) [17]		

While in general graphs FVS can be approximated within factor of 2 for all graphs, as shown by Becker and Geiger [4] and Bafna, Berman and Fujito [1], hitting a restricted family of cycles can be much harder. For example, the best known approximation ratio for graph bipartization in general graphs is $O(\log n)$ by Garg, Vazirani and Yannakakis [9]. For D-FVS the best known approximation is $O(\log n \log\log n)$, as shown by Even, Naor, Schieber and Sudan [8]. These and other results for general graphs are discussed in the full version.

Yannakakis [20] has given an NP-hardness proof for many vertex deletion problems restricted to planar graphs which applies to all problems that we consider. For planar graphs, the unweighted Feedback Vertex Set problem admits a PTAS, as shown by Demaine and Hajiaghayi [6] using a bidimensionality technique. Goemans and Williamson [11] created a framework for primal-dual algorithms that for planar instances of all above problems provide approximation algorithms with constant approximation factors. More specifically, they showed 9/4-approximations for FVS, S-FVS, D-FVS and BIP. For Node-Weighted Steiner Forest it was shown by Demaine, Hajiaghayi and Klein [5] that the generic framework of Goemans and Williamson gives a 6-approximation which was improved to 9/4-approximation by Moldenhauer [17]. However, the original paper by Goemans and Williamson [11] contains a mistake in the analysis. Similar mistake was repeated in [17]. We exhibit the mistake on an example and prove that no worse example exists. More precisely, primal-dual approximation algorithms of Goemans and Williamson for all problems described above give approximation factor 18/7 rather than 9/4. We also give an improved version of the violation oracle which can be used within the primal-dual framework of Goemans and Williamson and guarantees approximation factor 2.4. Results for planar graphs are summarized in Table 1.

Applications and ramifications. Node-weighted Steiner problems have been studied theoretically in many different settings, see e.g. [15,18,19,16]. Applications of such problems range from maintenance of electric power networks [12] to computational sustainability [7]. Experimental evaluation of primal-dual algorithms for feedback vertex set problems in planar graphs in applications to VLSI design has been done by Kahng, Vaya and Zelikovsky [13].

Organization. We give basic definitions and preliminary observations in Section 2. In Section 2.1 we show that a wide class of node-weighted network design problems in planar graphs, introduced by Demaine, Hajiaghayi and Klein [5], can

[1] See discussion in the text.

be equivalently defined as a class of hitting set problems for appropriately defined collections of cycles satisfying *uncrossing property*, as introduced by Goemans and Williamson [11]. In Section 3 we introduce local-ratio analog of primal-dual framework of Goemans and Williamson for such problems and give examples of violation oracles which can be used within this framework.

In Section 4 we give corrected analysis of the approximation factor achieved by the generic primal-dual algorithm with a violation oracle, presented by Goemans and Williamson in [11]. In the full version we present analysis of primal-dual algorithms with a new violation oracle which gives approximation factor 2.4. In Section 4.2 we show examples, on which these approximation factors are achieved.

2 Preliminaries

A *simple* cycle of length k is a sequence of vertices $v_1, \ldots, v_{k+1}$, where $v_{k+1} \equiv v_1$, all vertices $v_1, \ldots v_k$ are distinct, $(v_i, v_{i+1}) \in E$ for all $1 \leq i \leq k$ and all these edges are distinct. Note that in undirected simple graphs a simple cycle has length at least three. For a cycle C, the edge set of C is denoted as $E(C)$, although to simplify presentation we may refer to it as just C.

Every planar graph has a combinatorial embedding which for every vertex specifies a cyclic ordering of edges that are adjacent to it. A subset $U \subset V$ defines $G[U]$, the *induced subgraph* of G, with node set U and edges $\{(u, v) \in E : \ u, v \in U\}$. An embedding of a planar graph naturally defines embeddings of all its induced subgraphs. We denote the set of faces of a planar graph as F (for a standard definition of the set of faces via a combinatorial embedding, see e.g. [14]). The *planar dual* of a graph G is graph $G^* = (F, E')$ where F is the set of faces of G, and E' is the set of pairs of faces that share an edge. We select one face F_0 as the *outer face*.

For a simple cycle $C = (v_1, \ldots, v_{k+1})$ we denote the set of faces that are surrounded by C as $Faces(C)$. More formally, let E'' be the set of pairs of faces that share an edge that is not on C then in (F, E'') has exactly two connected components. We denote as $Faces(C)$ the connected component of (F, E'') that does not contain the outer face F_0.

For a weight function $w : V \to \mathbb{R}$ and a set $S \subseteq V$ we denote $w(S) = \sum_{e \in S} w(e)$.

2.1 Uncrossable Families of Cycles and Proper Functions

Two simple cycles C, D are *crossing* if neither $Faces(C) \subset Faces(D)$, nor $Faces(D) \subset Faces(C)$, nor $Faces(D) \cap Faces(C) = \varnothing$. A family of simple cycles $\mathcal{Z}$ is *laminar* iff it does not contain a pair of crossing cycles.

Our algorithms apply to every family of cycles that satisfies the following (similar to the *uncrossing property* of [11]). If two simple cycles C_1, C_2 are crossing then there exist paths $P_1 \subseteq C_1$ and $P_2 \subseteq C_2$, such that P_1 (P_2) intersects C_2 (C_1) only at its endpoints and P_2 contains an edge in the interior of C_1.

Definition 2.1 (Uncrossing property [11]). *A family of simple cycles $\mathcal{C}$ has the uncrossing property if for every pair of crossing cycles $C_1, C_2 \in \mathcal{C}$ as described above either $P_1 \cup P_2 \in \mathcal{C}$ and $(C_1 \setminus P_1) \cup (C_2 \setminus P_2)$ contains a cycle in $\mathcal{C}$, or $(C_1 \setminus P_1) \cup P_2 \in \mathcal{C}$ and $(C_2 \setminus P_2) \cup P_1$ contains a cycle in $\mathcal{C}$.*

Many natural families of cycles satisfy the *uncrossing property*. Goemans and Williamson [11] showed this for FVS, D-FVS, BIP, and S-FVS. We show that these problems belong to a wider class of node-weighted connectivity problems in planar graphs which can be expressed as problems of finding hitting sets for families of cycles satisfying the uncrossing property. To state it formally we introduce some definitions.

Definition 2.2 ($(0,1)$-proper function). *A Boolean function $f\colon 2^V \to \{0,1\}$ is* proper *if $f(\emptyset) = 0$ and it satisfies the following properties:*

1. *(Symmetry) $f(S) = f(V \setminus S)$.*
2. *(Disjointness) If $S_1 \cap S_2 = \emptyset$ and $f(S_1) = f(S_2) = 0$ then $f(S_1 \cup S_2) = 0$.*

These properties imply the property known as *complementarity*: if $A \subseteq S$ and $f(S) = f(A) = 0$ then $f(S \setminus A) = 0$.

For a set $S \subseteq V$, let $\Gamma(S)$ be its boundary, i.e. the set of nodes not in S which have a neighbor in S, or formally $\Gamma(S) = \{v \in V | v \notin S, \exists u \in S\colon (u,v) \in E\}$. As observed by Demaine, Hajiaghayi and Klein [5], a wide class of node-weighted network design problems can be formulated as the following generic integer program, where $f\colon 2^V \to \{0,1\}$ is a $(0,1)$-proper function:

$$\text{Minimize:} \quad \sum_{v \in V} w(v)x(v) \tag{1}$$

$$\text{Subject to:} \quad \sum_{v \in \Gamma(S)} x(v) \geq f(S) \qquad \text{for all } S \subseteq V \tag{2}$$

$$x(v) \in \{0,1\} \qquad \text{for all } v \in V, \tag{3}$$

For example, for Node-Weighted Steiner Forest the corresponding $(0,1)$-proper function is defined as follows: $f(S) = 1$ iff there exists a pair of terminals (s_i, t_i), such that $|S \cap \{s_i, t_i\}| = 1$. The edge-weighted version of this program was introduced by Goemans and Williamson in [10]. Note that without loss of generality we can assume that the input graph is triangulated. Otherwise we add extra nodes of infinite cost inside each face and connect these new nodes to all nodes on their faces without changing the cost of the optimum solution. Let V' be the set of nodes after such extension. Then the corresponding $(0,1)$-proper function f' for the extended instance is defined for all $S \subseteq V'$ as $f'(S) = f(S \cap V)$.

In Theorem 2.1 we show that a problem expressed by an integer program (1-3) with some $(0,1)$-proper function f can also be expressed as a problem of hitting a collection of cycles with the uncrossing property. We give some definitions and simplifying assumptions first.

Definition 2.3 (Active sets and boundaries). *Assume that* $f\colon 2^V \to \{0,1\}$ *is a* $(0,1)$*-proper function. If* $f(S) = 1$ *we say that* S *is* active*, and that* $\Gamma(S)$ *is an* active boundary*. If* $\Gamma(S)$ *is a simple cycle we call it an* active simple boundary*. We denote the collection of all active simple boundaries as* $\mathcal{C}^f$*.*

Using this terminology the integer program (1-3) expresses the problem of finding a minimum weight hitting set for the collection of all active boundaries. Note that every active singleton set $\{s\}$ must be included in the solution because $\{s\} = \Gamma(V \setminus \{s\})$ and $V \setminus \{s\}$ is active by symmetry, so $\{s\}$ has to be hit. Let S_0 be the set of such singletons. Using the observation above we can simplify the integer program (1-3) by using only inequalities of type (2) such that $\Gamma(S) \cap S_0 = \varnothing$. By disjointness of f, if $\Gamma(S) \cap S_0 = \varnothing$ then $f(\Gamma(S)) = 0$, *i.e.* every active boundary in the inequalities (2) of the simplified program is inactive.

In Lemma 2.1 we show that hitting all active boundaries is equivalent to hitting $\mathcal{C}^f$ because every active boundary contains an active simple boundary as a subset. This lemma is proved in the full version.

Lemma 2.1. *Let* $G(V,E)$ *be a connected triangulated planar graph,* f *be a* $(0,1)$*-proper function and* $\Gamma \subset V$ *be a set with the following properties:*

1. $f(\{a\}) = 0$ *for every* $a \in \Gamma$*.*
2. $f(B) = 1$ *for some* B *that is a connected component of* $V \setminus \Gamma$*.*

Then every set C *which is a minimal subset of* Γ *satisfying the two properties above is a simple cycle.*

Then we show that the family of active simple boundaries $\mathcal{C}^f$ satisfies the uncrossing property.

Theorem 2.1. *Let* $G(V,E)$ *be a triangulated planar graph. For every* $(0,1)$*-proper function* $f\colon 2^V \to \{0,1\}$ *the collection of active simple boundaries* $\mathcal{C}^f$ *forms an uncrossable family of cycles.*

Proof. Consider two active simple boundaries $\Gamma(S_1)$ and $\Gamma(S_2)$. If $\Gamma(S_2)$ crosses $\Gamma(S_1)$ then there exists a collection of edge-disjoint paths in $\Gamma(S_2)$ which we denote as P, such that each path $P_i \in P$ has only two nodes in common with $\Gamma(S_1)$. Each path $P_i \in P$ partitions $S_1 \setminus P_i$ into two parts which we denote as A_i^1 and A_i^2 respectively. Let's fix a path $P_i \in P$, such that at A_i^1 doesn't contain any other paths from P.

There are two cases: $A_i^1 \cap S_2 = \emptyset$ and $A_i^1 \subseteq S_2$. They are symmetric because if $A_i^1 \subseteq S_2$ we can replace the set S_2 by a set $S_2' = V \setminus S_2 \setminus \Gamma(S_2)$, ensuring that $A_i^1 \cap S_2 = \emptyset$. Note that the boundary doesn't change after such replacement, because $\Gamma(S_2) = \Gamma(S_2')$. By symmetry of f we have that $f(S_2) = f(V \setminus S_2) = 1$. Because $f(\Gamma(S_2)) = 0$ by disjointness we have $f(V \setminus S_2 \setminus \Gamma(S_2)) = f(S_2') = 1$, so S_2' is also an active set.

This is why it is sufficient to consider only the case when $A_i^1 \cap S_2 = \emptyset$. We will show the following auxiliary lemma:

Lemma 2.2. *Let $A_1, A, B \subseteq V$ be such that $A_1 \subseteq A$, $A_1 \cap B = \emptyset$ and $f(A) = f(B) = 1$. Then at least one of the following two statements holds:*
1. $f(A_1 \cup B) = f(A \setminus A_1) = 1$.
2. $f(A_1) = \max[f(B \setminus (A \setminus A_1)), f((A \setminus A_1) \setminus B)] = 1$.

The proof of the lemma follows from the properties of $(0,1)$-proper functions and is given in the full version.

To show the uncrossing property for cycles $C_1 = \Gamma(S_1)$ and $C_2 = \Gamma(S_2)$ we select the paths in the definition of the uncrossing property as $P_1 = \Gamma(A_i^2) \setminus P_i$ and $P_2 = P_i$. Now we can apply Lemma 2.2 to sets A_i^1, S_1 and S_2, because $A_i^1 \subseteq S_1$, $A_i^1 \cap S_2 = \emptyset$ and $f(S_1) = f(S_2) = 1$. Thus, by Lemma 2.2 either $f(A_i^1 \cup S_2) = f(S_1 \setminus A_i^1) = 1$ or $f(A_i^1) = max(f(S_2 \setminus (S_1 \setminus A_i^1)), f((S_1 \setminus A_i^1) \setminus S_2)) = 1$. In the first case we have $f(A_i^2) = f(A_i^1 \cup S_2) = 1$ and thus both cycles $P_1 \cup P_2 = \Gamma(A_i^2)$ and $(C_1 \setminus P_1) \cup (C_2 \setminus P_2) = \Gamma(A_i^1 \cup S_2)$ are active simple boundaries. In the second case $f(A_i^1) = 1$ and thus the cycle $(C_1 \setminus P_1) \cup P_2 = \Gamma(A_1)$ is an active simple boundary. The cycle $(C_2 \setminus P_2) \cup P_1$ is not necessarily simple, but it forms a boundary of an active set $(S_2 \setminus (S_1 \setminus A_i^1)) \cup ((S_1 \setminus A_i^1) \setminus S_2)$. Thus, by Lemma 2.1 it contains an active simple boundary, which is a cycle in $\mathcal{C}^f$.

3 Algorithm

3.1 Generic Local-Ratio Algorithm

We will use a local-ratio analog of a generic primal-dual algorithm formulated by Goemans and Williamson [11] which we state as Algorithm 1. As observed in the full verison of [17] these two formulations are equivalent for the problems that we consider (see also [2]).

Algorithm 1: Generic local-ratio algorithm $(G(V,E), w, \mathcal{C})$

1 $\bar{w} \leftarrow w$.
2 $S \leftarrow \{u \in V : \bar{w}(u) = 0\}$.
3 **while** *S is not a hitting set for $\mathcal{C}$* **do**
4 $\quad \mathcal{M} \leftarrow$ VIOLATION$(G, \mathcal{C}, S)$.
5 $\quad c_{\mathcal{M}}(u) \leftarrow |\{M \in \mathcal{M} : u \in M\}|$, for all $u \in V \setminus S$.
6 $\quad \alpha \leftarrow \min_{u \in V \setminus S} \frac{\bar{w}(u)}{c_{\mathcal{M}}(u)}$.
7 $\quad \bar{w}(u) \leftarrow \bar{w}(u) - \alpha c_{\mathcal{M}}(u)$, for all $u \in V \setminus S$.
8 $\quad S \leftarrow \{u \in V : \bar{w}(u) = 0\}$.
end
9 **return** a minimal hitting set $H \subset S$ of $\mathcal{C}$.

We say that a hitting set for a collection of cycles is minimal, if it doesn't contain another hitting set as its proper subset. Note that we don't need to specify the collection of cycles $\mathcal{C}$ explicitly. Instead the generic algorithm requires that we specify an oracle VIOLATION$(G, \mathcal{C}, S)$ used in Step 4. Given a graph G, collection

of cycles $\mathcal{C}$ and a solution S if there are some cycles in $\mathcal{C}$ which are not hit by S this oracle should return a non-empty collection of such cycles, otherwise it should return the empty set. Such an oracle also allows to perform Step 3 and Step 9 without explicitly specifying $\mathcal{C}$.

The performance guarantee of the generic algorithm depends on the oracle used as described below.

Theorem 3.1 (Local-ratio analog of Theorem 3.1 in [11]). *If the set $\mathcal{M}$ returned by a violation oracle used in Step 4 of the generic local-ratio Algorithm 1 satisfies that for any minimal solution $\breve{H}$:*

$$c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|,$$

then Algorithm 1 returns a hitting set H of cost $w(H) \leq \gamma w(H^*)$, where H^* is the optimum solution.

We give the proof of this theorem for completeness in the full version.

The simplest violation oracles return a single cycle. Bar-Yehuda, Geiger, Naor and Roth [3] show that for FVS this approach can give a 10-approximation for planar graphs and Goemans and Willamson [11] improve it to a 5-approximation. They also analyzed an oracle, which returns a collection of all faces in $\mathcal{C}$, which are not hit by the current solution, and showed such oracle gives a 3-approximation for all families of cycles satisfying uncrossing property. Thus, by Theorem 2.1 such oracle gives a 3-approximation for all problems that we consider. We now give more complicated examples of violation oracles which give better approximation factors.

3.2 Face Minimal Violation Oracles

Definition 3.1. *Given $S \subset V$, $\mathcal{C}(S) = \{C \in \mathcal{C} : C \cap S = \varnothing\}$. A cycle $C \in \mathcal{C}(S)$ is* face minimal *if there is no $D \in \mathcal{C}(S)$ such that $Faces(D) \subsetneq Faces(C)$.* $\textsc{Minimal}(S) = \{C \in \mathcal{C}(S) : C$ *is face minimal*$\}$.

Goemans and Williamson [11] showed that using $\textsc{Minimal}(S)$ as $\textsc{Violation}(G, \mathcal{C}, S)$ leads to approximation ratio 3. Other violation oracles we discuss can be computed by selecting a subset of $\textsc{Minimal}(S)$. Thus the algorithms we discuss run in polynomial time if the function $\textsc{Minimal}(S)$ can be computed in polynomial time. This is shown in [11,17] for the problems considered there. This also holds in general for sets of cycles defined by $(0, 1)$-proper functions.

Lemma 3.1. *For a family of cycles $\mathcal{C}^f$ defined by a $(0, 1)$-proper function* $\textsc{Minimal}(S)$ *can be computed in polynomial time.*

We give a sketch of the proof below. Let $\mathcal{A}$ be the set of active connected components of $V \setminus S$. Each cycle in $\textsc{Minimal}(S)$ will be a minimal subset of $\Gamma(A)$ for some $A \in \mathcal{A}$. However, we need to show how to find all cycles of $\textsc{Minimal}(S)$ rather than one.

We start by defining a partial order on $\mathcal{A}$. For a fixed $A \in \mathcal{A}$ we have set $\mathcal{K}(A)$ of connected components of $V \setminus \Gamma(A)$; note that $A \in \mathcal{K}(A)$. We say that a $B \in \mathcal{K}(A)$ is an outer (inner) component if B contains at least one node of the outer face (does not contain any). One can see that there exists at most one outer component in $\mathcal{K}(A)$. We say that A dominates $A' \in \mathcal{A}$ if some B is an inner component of $\mathcal{K}(A)$, $B \neq A$ and $A' \subset B$. This relation is anti-symmetric and transitive, hence it defines a partial order. We can show that each cycle of MINIMAL(S) is contained in $\Gamma(A)$ where A is a minimal element of $\mathcal{A}$ in terms of domination.

Then given such minimal A we first insert to A those nodes from $\Gamma(A)$ that have neighbors only in $\Gamma(A) \cup A$. Then we can show that resulting smaller $\Gamma(A)$ induces a subgraph that can be uniquely decomposed into a family of simple cycles, and exactly one of those cycles is a boundary of an active set. Details will be provided in the full version.

3.3 Minimal Pocket Violation Oracles

The following oracle, introduced by Goemans and Williamson [11], returns a collection of faces in $\mathcal{C}$ inside a minimal *pocket* not hit by the current solution H.

Definition 3.2. *A* pocket *for a planar graph $G(V, E)$ and a cycle collection $\mathcal{C}$ is a set $U \subseteq V$ such that:*

1. *The set U contains at most two nodes with neighbors outside U.*
2. *The induced subgraph $G[U]$ contains at least one cycle in $\mathcal{C}$.*

Algorithm 2: MINIMAL-POCKET-VIOLATION $(G, \mathcal{C}, S)$

```
C_0 ← {c ∈ C : c not hit by S}
M ← MINIMAL(S)
Construct a graph G_S by removing from G:
    All edges in the interior of cycles of M.
    All vertices which are not adjacent to any edges.
Let U_0 be a pocket for G_S and C_0 which doesn't contain any other pockets.
return A collection of all cycles in C_0 which are faces of G_S[U_0].
```

As in the generic algorithm, we will not specify $\mathcal{C}$ and $\mathcal{C}_0$ explicitly, but will rather use an oracle to check relevant properties with respect to them. We show analysis of the approximation factor obtained with this oracle in Section 4.

We will obtain a better approximation ratio by analyzing the following oracle in the full version.

Definition 3.3. *A* triple pocket *for a planar graph $G(V, E)$ and a cycle collection $\mathcal{C}$ is a set $U \subseteq V$ such that:*

1. *The set U contains at most three nodes with neighbors outside U.*
2. *The induced subgraph $G_S[U]$ has at least three faces in $\mathcal{C}$.*

The violation oracle Minimal-3-Pocket-Violation finds a minimal U_0 that is either a pocket or a triple pocket, and otherwise works like Minimal-Pocket-Violation.

4 18/7 Approximation Ratio with Pocket Oracle

According to Theorem 3.1, to show that Algorithm 1 with Minimal-Pocket-Violation oracle has approximation factor 18/7 it suffices to prove the following:

Theorem 4.1. *In every iteration of the generic local-ratio algorithm (Algorithm 1) with oracle* Minimal-Pocket-Violation *for every minimal hitting set $\breve{H}$ of $\mathcal{C}$ we have $c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|$ for $\gamma = 18/7$.*

The proof is in the full version.

4.1 12/5 Approximation Ratio with Triple Pocket Oracle

In the generic local-ratio algorithm we can change the implementation of the oracle Violation. Namely we can use Minimal-3-Pocket-Violation which in turn is a modification of Algorithm 2: in line 6 select U_0 as a minimal triple pocket. Note that a triple pocket is defined by three (or less) nodes that form $\Gamma(V - U_0)$, hence we still have a polynomial time. In the full version we prove

Theorem 4.2. *In every iteration of the generic local-ratio algorithm (Algorithm 1) with oracle* Minimal-3-Pocket-Violation *for every minimal hitting set $\breve{H}$ of $\mathcal{C}$ we have $c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|$ for $\gamma = 12/5$.*

4.2 Tight Examples

We show instances of graphs, on which the primal-dual algorithm with oracles Minimal-Pocket-Violation and Minimal-3-Pocket-Violation gives 18/7 and 12/5 approximations respectively.

Our examples are for the Subset Feedback Vertex Set problem. Recall that in this problem we need to hit all cycles which contain a specified set of "special" nodes. Our examples are graphs with no *pockets* (or triple pockets), in which every face belongs to the family of cycles that we need to hit – this is ensured by selection of "special" nodes, which are marked with a star . The weights of vertices are assigned as follows. Given a node u with degree $d(u)$, its weight is $w(u) = d(u)$ if u is a solid dot and $w(u) = d(u) + \epsilon$ otherwise (for some negligibly small value of ϵ).

First we show an example for the oracle Minimal-Pocket-Violation in Figure 1. Because there are no pockets, the first execution of the violation oracle returns the collection of all faces in the graph. Thus, in each building block of Picture 1 (which shows 5 such blocks from left to right), the primal-dual algorithm selects the black dots with total weight 18 while stars also form a

valid solution with weight $7 + 3\epsilon$. Hence the ratio will be arbitrarily close to 18/7, if we repeat the building block many times.

Similar family of examples for the oracle Minimal-3-Pocket-Violation is shown in Figure 2. In these examples there are no pockets or triple pockets, so the oracle Minimal-3-Pocket-Violation returns the collection of all faces in the graph. As above, the primal-dual algorithm selects the black dots with total weight 12 within each block, while the cost of the solution given by the stars is $5 + 2\epsilon$, so we can make the ratio arbitrarily close to 12/5.

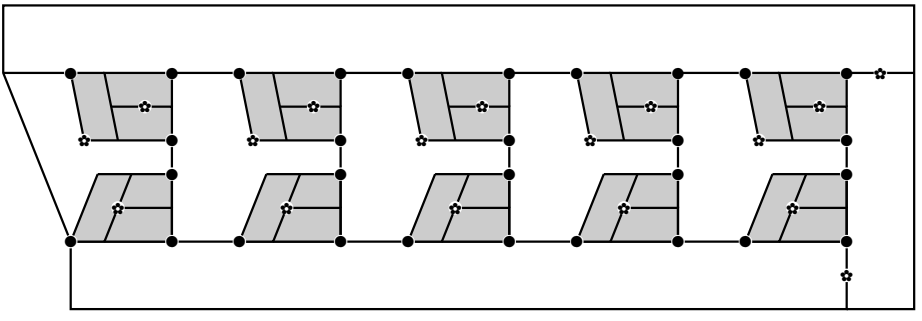

Fig. 1. Family of instances of S-FVS with approximation factor 18/7 for the primal-dual algorithm with oracle Minimal-Pocket-Violation

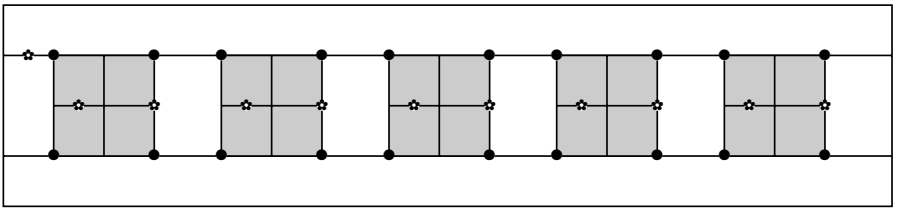

Fig. 2. Family of instances of S-FVS with approximation factor 12/5 for primal-dual algorithm with oracle Minimal-3-Pocket-Violation

References

1. Bafna, V., Berman, P., Fujito, T.: A 2-approximation algorithm for the undirected feedback vertex set problem. SIAM J. Discrete Math. 12(3), 289–297 (1999)
2. Bar-Yehuda, R., Bendel, K., Freund, A., Rawitz, D.: Local ratio: A unified framework for approxmation algrithms in memoriam: Shimon even 1935-2004. ACM Comput. Surv. 36(4), 422–463 (2004)
3. Bar-Yehuda, R., Geiger, D., Naor, J.S., Roth, R.M.: Approximation algorithms for the vertex feedback set problem with applications to constraint satisfaction and bayesian inference. In: SODA 1994, pp. 344–354. SIAM, Philadelphia (1994), http://dl.acm.org/citation.cfm?id=314464.314514

4. Becker, A., Geiger, D.: Optimization of pearl's method of conditioning and greedy-like approximation algorithms for the vertex feedback set problem. Artif. Intell. 83(1), 167–188 (1996)
5. Demaine, E.D., Hajiaghayi, M., Klein, P.N.: Node-Weighted Steiner Tree and Group Steiner Tree in Planar Graphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555, pp. 328–340. Springer, Heidelberg (2009)
6. Demaine, E.D., Hajiaghayi, M.: Bidimensionality: new connections between fpt algorithms and ptass. In: SODA 2005, pp. 590–601. SIAM, Philadelphia (2005), http://dl.acm.org/citation.cfm?id=1070432.1070514
7. Dilkina, B., Gomes, C.P.: Solving Connected Subgraph Problems in Wildlife Conservation. In: Lodi, A., Milano, M., Toth, P. (eds.) CPAIOR 2010. LNCS, vol. 6140, pp. 102–116. Springer, Heidelberg (2010)
8. Even, G., (Seffi) Naor, J., Schieber, B., Sudan, M.: Approximating minimum feedback sets and multicuts in directed graphs. Algorithmica 20, 151–174 (1998)
9. Garg, N., Vazirani, V.V., Yannakakis, M.: Approximate max-flow min-(multi)cut theorems and their applications. SIAM J. Comput. 25, 235–251 (1996)
10. Goemans, M.X., Williamson, D.P.: A general approximation technique for constrained forest problems. SIAM J. Comput. 24(2), 296–317 (1995)
11. Goemans, M.X., Williamson, D.P.: Primal-dual approximation algorithms for feedback problems in planar graphs. Combinatorica 18, 37–59 (1998)
12. Guha, S., Moss, A., Naor, J., Schieber, B.: Efficient recovery from power outage (extended abstract). In: STOC 1999, pp. 574–582 (1999)
13. Kahng, A.B., Vaya, S., Zelikovsky, A.: New graph bipartizations for double-exposure, bright field alternating phase-shift mask layout. In: ASP-DAC 2001, pp. 133–138. ACM, New York (2001)
14. Klein, P.: Optimization Algorithms for Planar Graphs, http://www.planarity.org/
15. Klein, P.N., Ravi, R.: A nearly best-possible approximation algorithm for node-weighted steiner trees. J. Algorithms 19(1), 104–115 (1995)
16. Li, X., Xu, X.-H., Zou, F., Du, H., Wan, P., Wang, Y., Wu, W.: A PTAS for Node-Weighted Steiner Tree in Unit Disk Graphs. In: Du, D.-Z., Hu, X., Pardalos, P.M. (eds.) COCOA 2009. LNCS, vol. 5573, pp. 36–48. Springer, Heidelberg (2009)
17. Moldenhauer, C.: Primal-Dual Approximation Algorithms for Node-Weighted Steiner Forest on Planar Graphs. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011. LNCS, vol. 6755, pp. 748–759. Springer, Heidelberg (2011)
18. Moss, A., Rabani, Y.: Approximation algorithms for constrained node weighted steiner tree problems. SIAM J. Comput. 37(2), 460–481 (2007)
19. Remy, J., Steger, A.: Approximation Schemes for Node-Weighted Geometric Steiner Tree Problems. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX and RANDOM 2005. LNCS, vol. 3624, pp. 221–232. Springer, Heidelberg (2005)
20. Yannakakis, M.: Node-and edge-deletion np-complete problems. In: STOC 1978, pp. 253–264. ACM, New York (1978)

What's the Frequency, Kenneth?: Sublinear Fourier Sampling Off the Grid

Petros Boufounos[1,⋆], Volkan Cevher[2,⋆⋆], Anna C. Gilbert[3,⋆⋆⋆], Yi Li[4], and Martin J. Strauss[5,†]

[1] Mitsubishi Electric Research Labs, 201 Broadway, Cambridge, MA 02139
petrosb@merl.com
[2] EPFL, Laboratory for Information and Inference Systems, Lausanne, Switzerland
volkan.cevher@epfl.ch
[3] Department of Mathematics, University of Michigan, Ann Arbor
annacg@umich.edu
[4] Department of EECS, University of Michigan, Ann Arbor
leeyi@umich.edu
[5] Departments of Mathematics and EECS, University of Michigan, Ann Arbor
martinjs@umich.edu

Abstract. We design a sublinear Fourier sampling algorithm for a case of sparse *off-grid* frequency recovery. These are signals with the form $f(t) = \sum_{j=1}^{k} a_j e^{i\omega_j t} + \hat{\nu}$, $t \in \mathbb{Z}$; i.e., exponential polynomials with a noise term. The frequencies $\{\omega_j\}$ satisfy $\omega_j \in [\eta, 2\pi - \eta]$ and $\min_{i \neq j} |\omega_i - \omega_j| \geq \eta$ for some $\eta > 0$. We design a sublinear time randomized algorithm, which takes $O(k \log k \log(1/\eta)(\log k + \log(\|a\|_1/\|\nu\|_1))$ samples of $f(t)$ and runs in time proportional to number of samples, recovering $\{\omega_j\}$ and $\{a_j\}$ such that, with probability $\Omega(1)$, the approximation error satisfies $|\omega'_j - \omega_j| \leq \eta/k$ and $|a_j - a'_j| \leq \|\nu\|_1/k$ for all j with $|a_j| \geq \|\nu\|_1/k$.

1 Introduction

Many natural and man-made signals can be described as having a few degrees of freedom relative to their size due to natural parameterizations or constraints; examples include AM, FM, and other communication signals and per-flow traffic measurements of the Internet. Sparse models capture the inherent structure of such signals via concise linear representations: A signal $y \in \mathbb{R}^N$ has a sparse representation as $y = \Psi x$ in a basis $\Psi \in \mathbb{R}^{N \times N}$ when $k \ll N$ coefficients x can exactly represent the signal y. Sparse models guide the way we acquire signals (e.g., sampling or sketching) and how we efficiently recover them from limited observations (e.g., sublinear recovery algorithms).

⋆ Exclusively supported by Mitsubishi Electric Research Laboratories.
⋆⋆ Supported by a Rice Faculty Fellowship, MIRG-268398, ERC Future Proof, SNSF 200021-132620, and DARPA KeCoM program #11-DARPA-1055.
⋆⋆⋆ Supported in part by NSF DMS 0354600 and partially supported by DARPA ONR N66001-06-1-2011.
† Supported in part by NSF DMS 0354600 and NSF DMS 0510203 and partially supported by DARPA ONR N66001-06-1-2011.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 61–72, 2012.
© Springer-Verlag Berlin Heidelberg 2012

There has been considerable effort to develop sublinear algorithms within the theoretical computer science community for recovering signals with a few significant discrete Fourier components, beginning with Kushilevitz and Mansour [1], including [2–4], and culminating in the recent work of Hassanieh, et al. [5, 6]. All of these algorithms are predicated upon treating the vector y as periodic and the discrete Fourier transform of a vector x being approximately k-sparse.

Unfortunately, these assumptions are too strong for many practical applications where the discrete Fourier transform coefficients are only approximation of an underlying continuous Fourier transform. For example, if we want to measure the approaching speed (the "doppler") of an object via the Doppler effect, we transmit a sinusoid wave $\mathrm{e}^{i\omega_0 t}$ (where t is time in this example) and receive a sinusoid wave whose frequency offset from ω_0 depends on the unknown doppler, v. Since v can be essentially any continuous value, so can be the received frequency. If there are two or more speeding objects in view, the received signal is of the form $f(t) = a_1\mathrm{e}^{i\omega_1 t} + a_2\mathrm{e}^{i\omega_2 t}$, where ω_1/ω_2 is not necessarily a rational number, so that $f(t)$ is not periodic. This practical and common example does not directly fit the discrete Fourier transform setting of [1–6].

To illustrate why we cannot simply reduce the continuous problem to the previous discrete Fourier sampling techniques, consider $f(t) = a_1\mathrm{e}^{i\omega_1 t} + a_2\mathrm{e}^{i\omega_2 t}$ and simply sample it on N equally-spaced points t. The Discrete Fourier Transform (DFT) of these samples produces a set of N coefficients at corresponding frequencies $2\pi\ell/N$, $\ell = 0, \ldots, N-1$, uniformly spaced in the interval $[0, 2\pi]$. It is also possible to compute the oversampled DFT, producing a larger set of coefficients $N' > N$, also corresponding to frequencies $2\pi\ell/N'$, $\ell = 0, \ldots, N'-1$, uniformly spaced in $[0, 2\pi]$. In this setting, the existing DFT-based methods often fail to capture the true sparsity of the signal and may blow up the sparsity in an unacceptable fashion. Indeed, even a 1-sparse original signal $f(t) = \mathrm{e}^{i\omega_1 t}$ for, say, $\omega_1 = 5.3 \cdot 2\pi/N$, will lead to a discretized signal whose Fourier transform is concentrated around $5 \cdot 2\pi/N$ and $6 \cdot 2\pi/N$, but is significant in $\Omega(N)$ places. This phenomenon arises even with an oversampled DFT, no matter how finely we discretize the frequency grid; *i.e.*, no matter how large N' is.

To this end, our approach lets ω range continuously while keeping t discrete. In Fourier analysis on (locally compact abelian) groups, the variables t and ω must be members of dual groups, which include the pairings $\mathbb{Z}_N \leftrightarrow \mathbb{Z}_N$, $\mathbb{Z} \leftrightarrow \mathbb{S}^1$ (where $\mathbb{S}^1$ denotes a continuous circle, $\mathbb{R}/\mathbb{Z}$), and $\mathbb{R} \leftrightarrow \mathbb{R}$. We take $t \in \mathbb{Z}$ and $\omega \in \mathbb{S}^1$. The generalization benefits over $\mathbb{Z}_N \leftrightarrow \mathbb{Z}_N$ are as follows.

- In $\mathbb{Z}_N \leftrightarrow \mathbb{Z}_N$, the data $\{f(n)\}$ are completely specified by N consecutive samples; we can treat double-ended infinite sequences $\{f(n)\}_{n\in\mathbb{Z}}$ provided $\sum_{n\in\mathbb{Z}} |f(n)| < \infty$.
- In $\mathbb{Z}_N \leftrightarrow \mathbb{Z}_N$, the frequencies ω_j must lie on a discrete grid; we can treat frequencies in the continuous space $\mathbb{S}^1$.

A concrete application of our approach (studied in the extended version of this paper) is the bearing (or angular direction) estimation of sources transmitting at fixed frequencies, a canonical array signal processing problem with applications

to radar, sonar, and remote sensing. Other applications also include the finite rate of innovation problems [7].

The organization of the paper is as follows. In Sect. 2, we define our model and problem formulation. In Sect. 3, we present our algorithm and its analysis. An application of this result to bearing estimation problems can be found in the full version of this paper.

2 Preliminaries

In this section, we define the problem of sublinear recovery of sparse off-grid frequencies, set the stage notationally, and then detail our results.

2.1 The Problem

We define a spectrally sparse function f with *off-grid* frequencies as a function $f : \mathbb{Z} \to \mathbb{C}$ with k frequencies $\omega_1, \ldots, \omega_k \in \mathbb{S}^1$, and we allow for noise ν in the spectrum that is supported on a set $I_\nu \subset \mathbb{S}^1$. We fix a minimum frequency resolution η and assume that $\{[\omega_j - \eta/2, \omega_j + \eta/2)\}_{j=1}^k$ and $[I_\nu - \eta/2, I_\nu + \eta/2)$ are all mutually disjoint. That is, the frequencies are not on a fixed, discrete grid but they are separated from each other and from the noise by a minimum frequency resolution. In our analysis below, we assume that $|\omega_j| > \eta$ without loss of generality. Specifically, we assume f is of the form

$$f(t) = \sum_{j=1}^{k} a_j \mathrm{e}^{i\omega_j t} + \int_{I_\nu} \nu(\omega) \mathrm{e}^{i\omega t} d\omega, \quad t \in \mathbb{R},$$

with $\nu \in L^1(I_\nu)$. Without loss of generality, we assume that $a_j \neq 0$ for all j. [1]

Our goal is to find all (a_j, ω_j) with

$$|a_j| \geq \frac{1}{k} \int_{I_\nu} |\nu(\omega)| d\omega \tag{1}$$

making as few samples on $\mathbb{Z}$ as possible (and with the smallest support) from f and for the shortest duration and to produce such a list in time comparable to the number of samples. The number of samples and the size of the support set of the samples should be proportional to a polynomial in k and $\log(1/\eta)$, the number of desired frequencies and precision. We call the frequencies ω_j whose associated amplitude a_j meet the threshold condition (1) *significant*.

[1] Strictly speaking these functions are not well-defined as, in the current definition, f is not in $L^1(\mathbb{Z})$ and does not have a mathematically well-defined Fourier transform (without resorting to more sophisticated mathematical techniques, such as tempered distributions). To be mathematically correct, we define f as above and then multiply it by a Gaussian window of width η^{100}. Call this mollified function $\tilde{f}$. The spectrum of $\tilde{f}$ is thus the convolution of $\hat{f}$ with a Gaussian of width η^{-100}. Up to the precision factor η/k, the spectra of $\tilde{f}$ and f are indistinguishable. Henceforth, we consider f with the understanding that $\tilde{f}$ is the well-defined version.

If we dilate the frequency domain $\mathbb{S}^1$ by a factor $1/d \in \mathbb{R}$ (*i.e.,* map ω to ω/d), we produce an equivalent sequence of samples $f(t)$, at regularly spaced real-valued points $t = nd$, $n \in \mathbb{Z}$. The dilation factor d determines the "rate" at which we sample the underlying signal and the total number of samples times the sampling rate is the duration over which we sample. Both the rate and the total number of samples are resources for our algorithm.

2.2 Notation

Let Ω be a domain (which can be either continuous or discrete). Roughly speaking, we call a function $K : \Omega \to \mathbb{R}$ a *filter* if K is or approximates the characteristic function χ_E of some set $E \subset \Omega$, which will be called the pass region of K. The resulting signal of applying filter K to signal f (viewed as a function on Ω) is the pointwise product $K \cdot f$.

Let K_m be a kernel defined on $\mathbb{S}^1$ (identified with $(-\pi, \pi]$) that satisfies the following properties:

- it is continuous on $\mathbb{S}^1$,
- its Fourier transform $\widehat{K_m} : \mathbb{Z} \to \mathbb{C}$ has finite support: $|\operatorname{supp} \widehat{K_m}| = O(\frac{m}{\alpha} \log \frac{1}{\epsilon})$,
- it approximates $\chi_{[-\frac{\pi}{m}, \frac{\pi}{m}]}$ (so K_m is a filter): $|K_m(x)| \leq \epsilon$ for $|x| \geq \frac{\pi}{m}$, $|K_m(x) - 1| \leq \epsilon$ for $|x| \leq (1-\alpha)\frac{\pi}{m}$ and $K_m(x) \in [-\epsilon, 1+\epsilon]$ elsewhere.

A Dolph-Chebyshev filter convolved with the characteristic function of an interval meets these criteria. We call the region $[-(1-\alpha)\frac{\pi}{m}, (1-\alpha)\frac{\pi}{m}]$ the plateau of K_m. The pass region of K_m is $[-\frac{\pi}{m}, \frac{\pi}{m}]$ and we define the transition region to be the complement of plateau in the pass region. A similar kernel was used in [5] and [6] with the only difference that their kernel was constructed by a Gaussian kernel convolved with the characteristic function of an interval.

2.3 Main Result

Theorem 1. *There is a distribution $\mathcal{D}$ on a set of sampling points $t \in \mathbb{R}$ and an algorithm $\mathcal{A}$ such that for each perturbed exponential polynomial $f(t) = \sum_{j=1}^{k} a_j e^{i\omega_j t} + \hat{\nu}(t)$, with constant probability, the algorithm returns a list $\Lambda = \{(a'_j, \omega'_j)\}_{j=1}^{k}$ of coefficients and frequencies such that*

1. *For each $|a_j| \geq \|\nu\|_1/k$ there exists $\omega'_j \in \Lambda$ such that*

$$|\omega_j - \omega'_j| \leq \frac{\eta}{k}.$$

2. *Let $\Lambda_0 = \left\{\omega'_j \in \Lambda : \exists \omega_{j_0} \text{ such that } |\omega_{j_0} - \omega'_j| \leq \frac{\eta}{k} \text{ and } |a_{j_0}| \geq \frac{\|\nu\|_1}{k}\right\}$, then for each $\omega'_j \in \Lambda_0$ it holds that*

$$|a'_j - a_j| \leq \frac{\|\nu\|_1}{k}.$$

3. For each $\omega'_j \in \Lambda \setminus \Lambda_0$*, it holds that*

$$|a'_j| \leq \frac{\|\nu\|_1}{k}.$$

The algorithm takes $O(k \log k \log(1/\eta)(\log k + \log(\|a\|_1/\|\nu\|_1)))$ *samples and runs in time proportional to number of samples. Furthermore, the size of the support of* $\mathcal{D}$*, i.e., the total duration of sampling, is* $O(k/\eta(\log k + \log(\|a\|_1/\|\nu\|_1)))$.

3 Analysis

Almost all sublinear sparse recovery algorithms (including both the Fourier and canonical basis) randomly hash frequencies or vector elements into buckets. Since the representation of the vector is sparse (in either the Fourier or the canonical basis), it is likely that each bucket contains exactly one coefficient and small noise so that the position of the "heavy hitter" can be found and then its value estimated. At a high level, our algorithm also follows this recipe. Some of these sublinear algorithms are iterative (*i.e.*, non-adpative hashing and estimation of the difference between the original vector and significant frequencies found in previous iterations) to use fewer samples or measurements or to refine inaccurate estimates. In contrast, our algorithm is not iterative. We hash the range of the frequencies into buckets and repeat sufficiently many times so that all frequencies are isolated, then we locate the frequency and estimate its amplitude.

A main difference between the discrete and continuous Fourier sampling problems is that, in the continuous frequency setting, it is impossible to recover a frequency exactly (from finite samples) so that one can subtract off recovered signals at exact positions. Typically in the discrete setting, an iterative algorithm uses a loop invariant either as in [8, 6] or in [3]. In the former case [8, 6], the number of buckets decreases per round as the number of remaining heavy hitters decreases. In the continuous case, however, the accuracy of the frequency estimates produced by location procedure are dependent on the width the pass region of the filter: the wider the pass region is, the more inaccurate the frequency estimate is. Unless the algorithm not only estimates the coefficient at a given frequency but also improves the frequency estimate, we must increase the distance d between samples from $O(k/\eta)$ to $O(k^2/\eta)$ to the achieve the same accuracy for the final frequency estimate, i.e., we must increase the duration over which samples are collected.

In the latter case [3], the number of buckets is kept the same at each round while the energy of the residual signal drops, and there are typically $\log \|a\|$ rounds. In hashing, we need to bound the inaccuracy $|K(h(\omega)) - K(h(\omega'))|$, where ω' is the recovered estimate of some real frequency ω, h the hash function and K the kernel. We can achieve this with a kernel that does not have a significant portion of its total energy outside of its pass region (*i.e.*, a "non-leaking" kernel), but it is not obvious how to achieve such an accurate estimate using a Dirichlet or Fejér kernel which was used in [3]. Unfortunately, using a "non-leaking" kernel like the one used in [5, 6] or the one used in this paper introduces a factor $\log \|a\|$ into the number of samples in order to decrease the noise in a bucket.

3.1 Recovery Algorithm

See Algorithm 1 for detailed pseudo-code.

3.2 Analysis of Algorithm

In this subsection, we provide a modular characterization of the algorithm.

Isolation. This portion of the analysis is similar to that of [6] but we emphasize the continuous frequency setting.

Let K_m be the kernel as described in Sec. 2 and set $D = 2\pi/\eta$. Define

$$\mathcal{H} = \{K_m(\omega d) = h_d(\omega) | d \in [D, 2D]\}$$

to be a family of hash functions. We choose h_d randomly from $\mathcal{H}$ by drawing d from the interval $[D, 2D]$ uniformly at random. Observe that the map $\omega \mapsto \omega d$ is a random dilation of $\mathbb{S}^1$. Similar to [6] and [3], we shall consider m-translations of K_m, denoted by $\{K_m^{(j)}\}_{j=0}^{m-1}$, where $K_m^{(j)}(x) = K_m\left(x + \frac{2\pi j}{m}\right)$ $(x \in \mathbb{S}^1)$, so that their pass regions cover $\mathbb{S}^1$. The pass regions will be referred to as *buckets* and the pass region of $K_m^{(j)}$ as j-th bucket. For convenience we shall also call the plateau of $K_m^{(j)}$ the plateau of the j-th bucket. It is clear that each frequency ω, under the random dilation $\omega \mapsto \omega d$, will land in some bucket with index $b(\omega, d)$. Similar to the hashing in [6], our hashing scheme guarantees that

- (small collision) Suppose that $|\omega - \omega'| \geq \eta$ then $\Pr\{b(\omega, d) = b(\omega', d)\} \leq c/m$ for some absolute constant $c > 0$.
- (good plateau landing) Suppose that $\omega \geq \eta$ and let $0 < \alpha < 1/2$ be as given in the definition of K_m, then ω lands in the plateau of the bucket with index $b(\omega, d)$ with probability $\geq (1 - \alpha)(1 - 1/m)$.

If a bucket contains exactly one frequency ω_{j_0}, we say that ω_{j_0} is isolated. Furthermore, if ω_{j_0} lands in the plateau of the bucket, we say that ω_{j_0} is well-isolated. Notice that when ω_{j_0} is isolated, it holds that $|h_d(\omega_j)| \leq \epsilon$ for all $j \neq j_0$.

The next lemma, an imitation of Lemma 3.1 in [3], allows us to bound the inaccuracy of its estimate in terms of the noise $\|\nu\|_1$.

Lemma 1. *Suppose that ξ is a random variable on $[D, 2D]$ such that $|\xi| \leq \pi/m$. Let $\omega \geq \eta$. Then $\mathbb{E}_d[|K_m(\omega d + \xi)|] \leq c/m$ for some absolute constant $c > 0$.*

Now we are ready to show that our algorithm isolates frequencies.

Fix j_0 and choose $m = \Omega(k)$. The hashing guarantees that ω_{j_0} is well-isolated with probability $\Omega(1)$ by taking a union bound. Also, it follows immediately from Lemma 1 that the expected contribution of ν to the bucket is at most $c\|\nu\|_1/m$. Therefore we conclude by Markov's inequality that

Lemma 2. *Conditioned on ω_{j_0} being well-isolated under $h_d \in \mathcal{H}$, w.p. $\Omega(1)$,*

$$\left| \sum_{j \neq j_0} a_j h_d(\omega_j) + \int_{I_\nu} \nu(\omega) h_d(\omega) d\omega \right| \leq C_1 \epsilon \|a\|_1 + \frac{C_2}{m} \|\nu\|_1$$

for some constants C_1, C_2 that depend on the failure probability.

Algorithm 1. The overall recovery algorithm

function MAIN
  $y \leftarrow$ signal samples
  $L \leftarrow$ IDENTIFY(y)
  $\Lambda \leftarrow$ ESTIMATE(L)
  return $\sum_{\omega \in \Lambda} a_\omega e^{i\omega t}$
end function

function IDENTIFY(y)
  $L \leftarrow \emptyset$
  for $t \leftarrow 1$ **to** $\Theta(\log m)$ **do**
    Choose a random d as described
    Collect z_{st}, the sample taking at time point with index (s, t)
    $b_i \leftarrow 0$ for all $i = 0, \ldots, m-1$
    for $r \leftarrow 1$ **to** $\lceil \log_2(1/\eta) \rceil$ **do**
      Compute $\{u_\ell\}_{\ell=0}^{m-1}$ and $\{v_\ell\}_{\ell=0}^{m-1}$ according to Remark 1
        where $u_\ell = \sum_j a_j K_m \left(\omega_j d - \frac{2\pi\ell}{m}\right) K_n \left(\frac{\omega_j d}{2^r} - \frac{2\pi}{2^r m}\ell - \frac{2b_\ell \pi}{2^r}\right)$
        and $v_\ell = \sum_j a_j K_m \left(\omega_j d - \frac{2\pi\ell}{m}\right) K_n \left(\frac{\omega_j d}{2^r} - \frac{2\pi}{2^r m}\ell - \frac{2b_\ell \pi}{2^r} - \pi\right)$
      for $\ell \leftarrow 0$ **to** $m-1$ **do**
        if $|v_\ell| > |u_\ell|$ **then**
          $b_r \leftarrow b_r + 2^{r-1}$
        end if
      end for
    end for
    for $\ell \leftarrow 0$ **to** $m-1$ **do**
      $L \leftarrow L \cup \{\frac{2\pi\ell}{md} + \frac{2b_\ell \pi}{d}\}$
    end for
  end for
  return L
end function

function ESTIMATE(L)
  Choose hash families $\mathcal{H}_1$ and $\mathcal{H}_2$ as described.
  for $r \leftarrow 1$ **to** $\Theta(\log k)$ **do**
    for each $\omega \in L$ **do**
      $a_\omega^{(r)} \leftarrow$ measurement w.r.t. $\mathcal{H}_1$
      $b_\omega^{(r)} \leftarrow$ measurement w.r.t. $\mathcal{H}_2$
    end for
  end for
  for each $\omega \in L$ **do**
    $a_\omega \leftarrow \text{median}_t\, a_\omega^{(r)}$
    $b_\omega \leftarrow \text{median}_t\, b_\omega^{(r)}$
  end for
  $L' \leftarrow \{x \in L : |b_\omega| \geq |a_\omega|/2\}$.
  $\Lambda \leftarrow \{(\omega, a_\omega) : \omega \in L'\}$.
  Cluster $\Lambda = \{(\omega, a_\omega)\}$ by x and retain only one element in the cluster.
  Retain top k ones (w.r.t. a_ω) in Λ
  return Λ
end function

Bit Testing. The isolation precedure above reduces the problem to the following: The parameter d is known, and exactly one of $\{\omega_j d\}_{j=1}^k$, say $\omega_{j_0} d$, belongs to $\bigcup_{n=0}^{N-1}[2n\pi - \delta, 2n\pi + \delta]$ for some small δ and (large) N. Suppose that $\omega_{j_0} d \in [2s\pi - \delta, 2s\pi + \delta]$. We shall find s and thus recover ω_{j_0}. Assume that ω_{j_0} is significant, i.e., a_{j_0} satisfies (1).

We recover s from the least significant bit to the most significant bit, as in [3]. Assume we have already recovered the lowest r bits of s, and by translation, the lowest r bits of s are 0s. We shall now find the $(r+1)$-st lowest bit.

Let K_n (n is a constant, possibly $n = 3$) be another kernel with parameter ϵ'. The following lemma shows that Line 6–14 of IDENTIFY gives the correct s.

Lemma 3. *Suppose that the lowest r bits of s are 0, let $G_1 = K_m(x)K_n\left(\frac{x}{2^r}\right)$, $G_2 = K_m(x)K_n\left(\frac{x}{2^r} - \pi\right)$ and u be the sample taken using G_1 and v using G_2, then $|u| > |v|$ if $s \equiv 0 \pmod{2^r}$ and $|u| < |v|$ if $s \equiv 2^r \pmod{2^{r+1}}$, provided that $m = \Omega(k)$ and $\epsilon \le \|\nu\|_1/(m\|a\|_1)$.*

Proof. We leverage the isolation discussion. By Lemma 2, when $s \equiv 0 \pmod{2^r}$,

$$|u| \ge (1-\epsilon)(1-\epsilon')|a_{j_0}| - (1+\epsilon')\left(C_1\epsilon\|a\|_1 - \frac{C_2}{m}\|\nu\|_1\right). \tag{2}$$

and when $s \equiv 2^{r-1} \pmod{2^r}$,

$$|u| \le (1+\epsilon)\epsilon'|a_{j_0}| + (1+\epsilon')\left(C_1\epsilon\|a\|_1 + \frac{C_2}{m}\|\nu\|_1\right). \tag{3}$$

Similar bounds hold for $|v|$. Thus it suffices to choose $m \ge \frac{2(1+\epsilon')(C_1+C_2)}{1-\epsilon-2\epsilon'}k$. □

Repeat this process until $r = \log_2(\pi D) = O(\log(\pi/\eta))$ to recover all bits of s. At each iteration step the number of samples needed is $O(|\operatorname{supp}\widehat{G_1}| + |\operatorname{supp}\widehat{G_2}|) = O(|\operatorname{supp}\widehat{K_m}| \cdot |\operatorname{supp}\widehat{K_n}|) = O(k\log\frac{1}{\epsilon})$, so the total number of samples used in a single execution of Line 8 of IDENTIFY is $O(k\log\frac{1}{\epsilon}\log\frac{1}{\eta})$.

The precision of $\omega_{j_0} d$ will be $\delta = \pi/m$ and thus the precision of ω_{j_0} will be $\delta/d \le \pi/(mD) = \eta/m$. In summary, the hashing process guarantees that

Lemma 4. *With probability $\Omega(1)$, IDENTIFY returns a list L such that for each ω_j with a_j satisfying (1), there exists $\omega' \in L$ such that $|\omega' - \omega_j| \le \eta/m$.*

Remark 1. Notice that $\sigma(K_m) \subseteq [-M, M] \cap \mathbb{Z}$ for integer $M = O(\frac{k}{\alpha}\log\frac{1}{\epsilon})$. We shall show that, similar to [3], despite Line 6–14 of IDENTIFY (for m translations altogether) requires mr numbers, each of which is a sum of $O(M)$ terms, this process can be done in $O((M + m\log m)r)$ time instead of $O(Mmr)$ time.

Suppose that at step r, the translation that shifts the lowest bits of s_j to 0 is b_j $(0 \le j \le m-1)$. In Line 8 of IDENTIFY, each u_j or v_j has the form

$$\sum_{s=-\Theta(n)}^{\Theta(n)} e^{-2\pi i(b_j + \frac{j}{m})\frac{s}{2^r}} \sum_{t=-M}^{M} e^{-2\pi i\frac{jt}{m}} w_{st} z_{st}, \quad j = 0, \ldots, m-1,$$

where z_{st} is the sample at time with index (s,t) and the associated weight is w_{st}. Notice that the inner sum can be rewritten as

$$\sum_{\ell=0}^{m} e^{-2\pi i \frac{j\ell}{m}} \sum_{t \in \left(m\mathbb{Z}+\{\ell\}\right)\cap[-M,M]} w_{st} z_{st},$$

which can be done in $O(M + m \log m)$ time using FFT. The outer sum has only constantly many terms. Hence Line 8 of IDENTIFY takes $O(M + m \log m)$ times. There are r steps, so the total time complexity is $O((M + m \log m)r)$.

Amplitude Estimation. The isolation procedure generates a list L of candidate frequencies. Like [6], we estimate the amplitude at each position in L by hasing it into buckets using the same kernel but with possibly different parameters. We shall show how to extract good estimates and eliminate unreliable estimates among $|L|$ estimates.

The following lemma states that if a frequency candidate is near a true frequency then they fall in the same bucket with a good probability and if a frequency candidate is adequately away from a true frequency then they fall in different buckets with a good probability.

Lemma 5. *Let $D = \Theta(1/\eta)$ and $\delta > 0$. Choose d uniformly at random from $[\theta_1 D, \theta_2 D]$.*

1. *if $|\omega - \omega'| \le \beta_1 \delta/D \le \eta$ then $\Pr\{b(\omega', d) = b(\omega, d)\} \ge 1 - \beta_1\theta_2$. Thus except with probability $\le \beta_1\theta_2 + \alpha$ it holds that ω falls in the same bucket as ω';*
2. *if $|\omega - \omega'| \ge \beta_2 \delta/D$ then $\Pr\{b(\omega', d) = b(\omega, d)\} \le 1/(\beta_2(\theta_2 - \theta_1)) + c\delta D$ for some universal constant $c > 0$.*

Choose parameters $0 < \beta_1 < \beta_2$, $0 < \theta_1 < \theta_2$ such that $\beta_1\theta_2 + \alpha < 1/3$ and $1/(\beta_2(\theta_2 - \theta_1)) < 1/3$. Let $D = C\pi/\eta$. Define a hash family

$$\mathcal{H} = \{K_m(\omega d) = h_d(\omega) | d \in [\theta_1 D, \theta_2 D]\}.$$

As a direct corollary of Lemma 5 we have

Lemma 6. *Let $\omega' \ge \eta$ and $j_0 = \arg\min_j |\omega' - \omega_j|$. Obtain a measurement $a_{\omega'}$ w.r.t. $h_d \in \mathcal{H}$.*

1. *If $|\omega' - \omega_{j_0}| \le \beta_1 C\eta/m$, with probability $\Omega(1)$, it holds that $|a_{\omega'} - a_{j_0}| \le \epsilon\|a\|_1 + c'\|\nu\|_1/m$ for some $c' > 0$ dependent on the failure probability;*
2. *If $|\omega' - \omega_{j_0}| \ge \beta_2 C\eta/m$, with probability $\Omega(1)$, it holds that $|a_{\omega'}| \le \epsilon\|a\|_1 + c'\|\nu\|_1/m$ for some $c' > 0$ dependent on the failure probability.*

Let $\Delta = \epsilon\|a\|_1 + c'\|\nu\|_1/m$, where c' is a constant dependent on the failure probability guaranteed in the lemma.

Take different $C_1 > C_2$ (and thus different D_1 and D_2) such that $\beta_1 C_2 \ge 1$ and that $C_2\beta_2 \le C_1\beta_1$. Define hash families $\mathcal{H}_i$ $(i = 1, 2)$ as

$$\mathcal{H}_i = \{K_m(\omega d) = h_d(\omega) | d \in [\theta_1 D_i, \theta_2 D_i]\}, \quad i = 1, 2.$$

It then follows that

Lemma 7. *Upon termination of execution of line 13 in* ESTIMATE*, with probability $\Omega(1)$, for each $\omega' \in L'$ let $j_0 = \arg\min_j |\omega' - \omega_j|$ it holds that*

1. *If $|\omega' - \omega_{j_0}| \le \beta_1 C_1 \eta/m$, then $|a_{\omega'} - a_{j_0}| \le \Delta$;*
2. *If $|\omega' - \omega_{j_0}| \ge \beta_2 C_1 \eta/m$, then $|a_{\omega'}| \le \Delta$*
3. *If $\beta_1 C_1 \eta/m \le |\omega' - \omega_{j_0}| \le \beta_2 C_1 \eta/m$, then $|a_{\omega'}| \le 2\Delta$.*

Loosely speaking, Lemma 7 guarantees a multiplicative gap between the amplitude estimates for the "good" estimates of significant frequencies and the amplitudes estimates for all other frequency estimates. Next, we merge estimates of the same true source utilizing the gap as follows. In increasing order, for each $\omega' \in L'$ with amplitude estimate $a_{\omega'}$, find

$$I(\omega') = \left\{ \omega \in L' : \omega' \le \omega \le \omega' + \frac{C_1\beta_1\eta}{m} \text{ and } \frac{2}{\gamma-1}|a_{\omega'}| < |a_\omega| < \frac{\gamma-1}{2}|a_{\omega'}| \right\},$$

where $\gamma > 3$ is a constant to be determined later.

Choose an arbitary element from I as the representative of all elements in I and add it to Λ. Continue this process from the next $\omega' \in L$ that is larger than all elements in I. Retain the top k items of Λ.

Lemma 8. *Suppose that* ESTIMATE *is called with argument L. With probability $\Omega(1)$, it produces a list Λ such that*

1. *For each j with $|a_j| \ge \gamma\Delta$ for some $\gamma > 2 + \sqrt{5}$, if there exists $\omega' \in L$ such that $|\omega' - \omega_j| \le \pi/m$, then there exists $(\omega'', a_{\omega''}) \in \Lambda$ (we say that $\omega'' \in \Lambda$ is paired) such that $|\omega'' - \omega_j| \le C_1\beta_1\eta/m$ and $|a_{\omega''} - a_j| \le \Delta$.*
2. *For each unpaired $\omega \in \Lambda$ it holds that $|a_\omega| \le 2\Delta$.*

Proof. In case (1), for all $\omega \in L'$ such that $|\omega - \omega_j| \le C_1\beta_1\eta/m$ it holds that $|a_\omega| \ge (\gamma-1)\Delta$ while for other ω it holds that $|a_\omega| \le 2\Delta$. There is a multiplicative gap so the merging process does not mix frequencies that are close to and far away from a true source. It is easy to verify that $\omega \in L'$ upon termination of line 13 since $C_2\beta_1 \ge 1$. The rest is obvious. □

Our main result is now ready.

Proof (of Theorem 1). We show that MAIN returns the claimed result with probability $\Omega(1)$. Choose ϵ in the estimation procedure to be $\epsilon = \|\nu\|_1/(2\gamma k\|a\|_1)$ and $m \ge \gamma c' k$, then $\Delta \le \|\nu\|_1/(\gamma k)$ and thus whenever $|a_j|$ satisfies (1) it holds that $|a_j| \ge \gamma\Delta$. Combining Lemma 4 and Lemma 8 completes the proof. □

Number of Samples. There are $O(\log k)$ repetitions in isolation and each takes $O(k \log \frac{1}{\epsilon} \log \frac{1}{\eta})$ samples, hence the isolation procedure takes $O(k \log k \log \frac{1}{\epsilon} \log \frac{1}{\eta})$ samples in total.

The input of ESTIMATE is a list L of size $|L| = O(m \log m) = O(k \log k)$. Use the same trick as in isolation, it takes $O(M) = O(k \log(1/\epsilon))$ samples for each of $O(\log k)$ repetitions. Hence the estimation takes $O(k \log k \log \frac{1}{\epsilon} \log \frac{1}{\eta})$ samples.

The total number of samples is therefore

$$O\left(k \log k \log \frac{1}{\epsilon} \log \frac{1}{\eta}\right) = O\left(k \log k \left(\log \frac{\|a\|_1}{\|\nu\|_1} + \log k\right) \log \frac{1}{\eta}\right).$$

Run Time. It follows from Remark 1 that each isolation repetition takes $O((M + m \log m)r) = O(k \log \frac{k}{\epsilon} \log \frac{1}{\eta})$ time. There are $O(\log m) = O(\log k)$ repetitions so the total time for isolation is $O(k \log k \log \frac{k}{\epsilon} \log \frac{1}{\eta})$.

The input of ESTIMATE is a list L of size $|L| = O(k \log k)$. Use the same trick as in isolation, it takes $O(M + m \log m + |L|)$ to obtain values for all buckets and compute $a_\omega^{(s)}$ and $b_\omega^{(s)}$ for all $\omega \in L$ and each s. Hence line 3–8 of ESTIMATE takes time $O((M + m \log m + |L|) \log k) = O(k \log k \log(k/\epsilon))$ time. Thus estimation takes time $O(k \log k \log(k/\epsilon)) + |L| \log k + |L| \log |L|) = O(k \log k \log(k/\epsilon))$.

The total running time is domimated by that of isolation, which is proportional to the number of samples taken.

Output Evaluation Metric. Since we do not expect to recover the frequencies exactly, the typical approximation error of the form

$$\left\| \sum_j a_j e^{i\omega_j t} - a'_j e^{i\omega'_j t} + \nu(t) \right\|_p$$

contains both the amplitude approximation error $\|a - a'\|$ and a term of the form $\sum |a_j||\omega_j - \omega'_j|$, rather than the more usual bound in terms of the noise alone $\|\nu\|_p$ in the discrete case. Given bounds on both the amplitudes $|a_j - a'_j|$ and the frequencies $|\omega_j - \omega'_j|$, it is possible to compute the two terms in the error. This is standard in the literature of polynomial-time algorithms to recover real frequencies (e.g., [9], with which our result is comparable).

4 Conclusion

In this paper, we define a mathematically rigorous and practical signal model for sampling sparse Fourier signals with continuously placed frequencies and devise a sublinear time algorithm for recovering such signals. There are a number of technical difficulties in this model with directly applying the discrete sublinear Fourier sampling techniques, both algorithmic and mathematical. In particular, several direct techniques incur the penalty of extra measurements. We do not know if these additional measurements are necessary, if they are inherent in the model. Furthermore, unlike the discrete case, the "duration" of the sampling or the extent of the samples is a resource for which we have no lower bounds.

Acknowledgements. The authors would like to thank an anonymous reviewer for a suggestion that improves the running time.

References

1. Kushilevitz, E., Mansour, Y.: Learning decision trees using the Fourier spectrum. In: STOC, pp. 455–464 (1991)
2. Gilbert, A.C., Guha, S., Indyk, P., Muthukrishnan, M., Strauss, M.: Near-optimal sparse fourier representations via sampling. In: STOC, pp. 152–161 (2002)
3. Gilbert, A.C., Muthukrishnan, S., Strauss, M.: Improved time bounds for near-optimal sparse Fourier representations. In: Proceedings of Wavelets XI Conference (2005)
4. Iwen, M.: Combinatorial sublinear-time Fourier algorithms. Foundations of Computational Mathematics 10(3), 303–338 (2009)
5. Hassanieh, H., Indyk, P., Katabi, D., Price, E.: Simple and practical algorithm for sparse Fourier transform. In: SODA, pp. 1183–1194 (2012)
6. Hassanieh, H., Indyk, P., Katabi, D., Price, E.: Nearly optimal sparse Fourier transform. In: STOC, pp. 563–578 (2012)
7. Vetterli, M., Marziliano, P., Blu, T.: Sampling signals with finite rate of innovation. IEEE Transactions on Signal Processing 50(6), 1417–1428 (2002)
8. Gilbert, A.C., Li, Y., Porat, E., Strauss, M.: Approximate sparse recovery: Optimizing time and measurements. SIAM J. Comput. 41(2), 436–453 (2012)
9. Peter, T., Potts, D., Tasche, M.: Nonlinear approximation by sums of exponentials and translates. SIAM J. Sci. Comput. 33(4), 1920–1947 (2011)

Improved Hardness Results for Profit Maximization Pricing Problems with Unlimited Supply

Parinya Chalermsook[1,⋆], Julia Chuzhoy[2,⋆⋆], Sampath Kannan[3,⋆⋆⋆], and Sanjeev Khanna[3,†]

[1] University of Chicago, Chicago, IL and IDSIA, Lugano, Switzerland
[2] Toyota Technological Institute, Chicago, IL
[3] Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA

Abstract. We consider profit maximization pricing problems, where we are given a set of m customers and a set of n items. Each customer c is associated with a subset $S_c \subseteq [n]$ of items of interest, together with a budget B_c, and we assume that there is an unlimited supply of each item. Once the prices are fixed for all items, each customer c buys a subset of items in S_c, according to its buying rule. The goal is to set the item prices so as to maximize the total profit.

We study the unit-demand min-buying pricing ($\mathsf{UDP_{MIN}}$) and the single-minded pricing (SMP) problems. In the former problem, each customer c buys the cheapest item $i \in S_c$, if its price is no higher than the budget B_c, and buys nothing otherwise. In the latter problem, each customer c buys the whole set S_c if its total price is at most B_c, and buys nothing otherwise. Both problems are known to admit $O(\min\{\log(m+n), n\})$-approximation algorithms. We prove that they are $\log^{1-\epsilon}(m+n)$ hard to approximate for any constant ϵ, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\log^{\delta} n})$, where δ is a constant depending on ϵ. Restricting our attention to approximation factors depending only on n, we show that these problems are $2^{\log^{1-\delta} n}$-hard to approximate for any $\delta > 0$ unless $\mathsf{NP} \subseteq \mathsf{ZPTIME}(n^{\log^{\delta'} n})$, where δ' is some constant depending on δ. We also prove that restricted versions of $\mathsf{UDP_{MIN}}$ and SMP, where the sizes of the sets S_c are bounded by k, are $k^{1/2-\epsilon}$-hard to approximate for any constant ϵ.

We then turn to the Tollbooth Pricing problem, a special case of SMP, where each item corresponds to an edge in the input graph, and each set

⋆ Supported in part by NSF CAREER grant CCF-0844872, Swiss National Science Foundation project 200020-122110/1, and Hasler Foundation Grant 11099. Part of this work was done while at University of Chicago.

⋆⋆ Supported in part by NSF CAREER grant CCF-0844872 and Sloan Research Fellowship.

⋆⋆⋆ Supported in part by the National Science Foundation grant CCF-1137084.

† Supported in part by the National Science Foundation grants CCF-1116961 and IIS-0904314.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 73–84, 2012.
© Springer-Verlag Berlin Heidelberg 2012

S_c is a simple path in the graph. We show that Tollbooth Pricing is at least as hard to approximate as the Unique Coverage problem, thus obtaining an $\Omega(\log^{\epsilon} n)$-hardness of approximation, assuming NP $\not\subseteq$ BPTIME$(2^{n^{\delta}})$, for any constant δ, and some constant ϵ depending on δ.

1 Introduction

We study profit maximization pricing problems in the unlimited supply model. In these problems, we are given a set of m customers and a set of n items, where each customer c is associated with a budget B_c, and a subset $S_c \subseteq [n]$ of items it is interested in. Our goal is to set a price $p(i)$ for each item $i \in [n]$, so as to maximize the total revenue. Once the prices for the items are set, each customer c chooses a subset of items in S_c to buy, using its *buying rule*. We assume that we are given an unlimited supply of each item.

One of the most natural buying rules is the unit-demand min-buying rule, where each customer $c \in [m]$ buys the cheapest item $i \in S_c$ (breaking ties arbitrarily), provided that the price $p(i) \leq B_c$. We refer to the corresponding pricing problem as $\mathsf{UDP_{MIN}}$. This problem was first introduced by Rusmevichientong et al. [18,19], and subsequently Aggarwal et al. [1] have shown an $O(\log m + \log n)$-approximation algorithm for it.

The second problem that we consider is Single-Minded Pricing (SMP). Here, each customer c buys the whole set S_c of items if its total price does not exceed its budget B_c, and buys nothing otherwise. This problem was introduced by Guruswami et al. [14], who also show that the techniques of [1] can be used to obtain an $O(\log m + \log n)$-approximation algorithm for SMP. Hartline and Koltun [15] gave a $(1+\epsilon)$-approximation algorithm for both $\mathsf{UDP_{MIN}}$ and SMP when the number of items n is constant.

We remark that for pricing problems, it is natural to assume that the number of customers is much higher than the number of items, that is, $m >> n$. Even though both $\mathsf{UDP_{MIN}}$ and SMP admit logarithmic approximation algorithms in terms of $(m+n)$, if we restrict ourselves to approximation factors depending only on n, nothing better than the trivial $O(n)$-approximation is known.

On the negative side, Briest [3] has shown that both $\mathsf{UDP_{MIN}}$ and SMP are $\max\left\{n^{\delta}, \log^{\delta}(m+n)\right\}$-hard to approximate for some (small) $\delta > 0$, assuming that no randomized polynomial-time algorithms can approximate constant-degree Balanced Bipartite Independent Set to within arbitrarily small constant factors. He also showed similar results under an assumption that slightly strengthens Feige's Random 3SAT hypothesis [11].

In this paper, we show that both $\mathsf{UDP_{MIN}}$ and SMP are $\log^{1-\epsilon}(m+n)$ hard to approximate for any constant ϵ, unless NP $\subseteq$ DTIME$(n^{\log^{\epsilon'} n})$ for some constant ϵ' depending only on ϵ. If we restrict our attention to approximation factors as a function of n, then we show that both these problems are $2^{\log^{1-\delta} n}$ hard to approximate for any constant δ, under the assumption that NP $\not\subseteq$ ZPTIME$(n^{\log^{\delta'} n})$, for some constant δ' depending only on δ.

We next turn to restricted versions of $\mathsf{UDP_{MIN}}$ and SMP, denoted by $\mathsf{kUDP_{MIN}}$ and kSMP respectively, where the sizes of the sets S_c are bounded by k. The kSMP problem is known to be APX-hard even for $k = 2$ [14], and Balcan and Blum [2] have shown an $O(k)$-approximation for $\mathsf{kUDP_{MIN}}$, improving on an independent work of Briest and Krysta [4], who achieved an $O(k^2)$-approximation for the problem. As for negative results, Briest [3] has proved that kSMP is k^ϵ-hard to approximate for some constant ϵ, assuming Feige's random 3SAT hypothesis [11], and Khandekar et al. [16] showed that the problem is $\Omega(k)$ hard to approximate for constant k, assuming the Unique Games Conjecture of Khot [17]. We show that both $\mathsf{kUDP_{MIN}}$ and kSMP are $k^{1/2-\epsilon}$-hard to approximate for any constant ϵ unless $\mathsf{P} = \mathsf{NP}$.

Finally, we consider a special case of the SMP problem called the Tollbooth Pricing problem, where we are given a graph G, and items correspond to the edges of G. The item set S_c of every customer c is some simple path in graph G, and the goal is to set the prices of the edges, so as to maximize the revenue. Since the Tollbooth Pricing problem is a special case of SMP, it admits an $O(\log m + \log n)$ approximation [14]. The problem is APX-hard [14], and from the results of Khandekar et al. [16], it is $(2 - \epsilon)$ hard to approximate even on star graphs, assuming the Unique Games Conjecture. We show that the Tollbooth Pricing problem is at least as hard to approximate as the Unique Coverage problem (to within a constant factor). In the Unique Coverage problem, we are given a collection U of n elements, and a family $\mathcal{S}$ of subsets of elements of U. The goal is to find a family $\mathcal{S}' \subseteq \mathcal{S}$ of element subsets, maximizing the number of elements that are covered by exactly one subset in $\mathcal{S}'$. The problem was introduced and studied by Demaine et al. [8], who showed that for any arbitrarily small constant δ, if $\mathsf{NP} \not\subseteq \mathsf{BPTIME}(2^{n^\delta})$, then Unique Coverage is hard to approximate to within a factor of $\Omega(\log^\epsilon n)$, where ϵ is some constant depending on δ. They also showed that the problem is hard to approximate to within $\Omega(\log^{1/3-\epsilon} n)$ for any ϵ assuming the Random 3SAT Hypothesis of Feige [11], and proved additional hardness results using a hypothesis about Balanced Bipartite Independent Set. Our reduction immediately implies similar hardness results for the Tollbooth Pricing problem.

Related Work. Briest and Krysta [4] considered a more general version of $\mathsf{UDP_{MIN}}$, where customers are allowed to have different budgets (valuations) for different items. They show an $\Omega(\log^\epsilon n)$-hardness for this problem for some constant ϵ, unless $\mathsf{NP} \subseteq \mathsf{DTIME}\left(n^{O(\log \log n)}\right)$, and an n^ϵ-hardness for some constant $\epsilon > 0$, unless $\mathsf{NP} \subseteq \mathsf{DTIME}\left(2^{O(n^\delta)}\right)$ for all $\delta > 0$.

A special case of the Tollbooth Pricing problem, called the Highway Problem, where the input graph is restricted to be a path, has received a significant amount of attention. Elbassioni et al. [9] showed that the problem is strongly NP-hard. On the algorithmic side, Balcan and Blum [2] have shown an $O(\log n)$-approximation algorithm, and Elbassioni et al. [10] have proposed a QPTAS. Subsequently, Grandoni and Rothvoss [13] have shown a PTAS for the problem. For the special case of the Tollbooth Pricing problem where the input graph is

a tree, the best known approximation ratio is $O(\log n/\log\log n)$, due to Gamzu and Segev [12]. However, when the number of leaves in the tree is bounded by a constant, the problem admits a PTAS [13].

Pricing problems with limited supply have also received a considerable amount of attention; Please refer to, e.g., [5,7,6] and references therein.

Our Results. We start by formally stating the pricing problems we consider. We are given a set of m customers and a set of n items, where each customer $c \in [m]$ is associated with a set $S_c \subseteq [n]$ of items and a budget B_c. Given a setting $\{p(i)\}_{i\in[n]}$ of item prices, every customer selects a subset $S'_c \subseteq S_c$ of items to buy according to its buying rule, and our goal is to maximize the total profit, $\sum_{c\in[m]} \sum_{i\in S'_c} p(i)$. In the $\mathsf{UDP_{MIN}}$ problem, the buying rule of the customers is defined as follows. Each customer $c \in [m]$ buys the cheapest item $i \in S_c$, breaking ties arbitrarily, if $p(i) \leq B_c$, and buys nothing otherwise.

In the SMP problem, each customer $c \in [m]$ purchases the whole set S_c if $\sum_{i\in S_c} p(i) \leq B_c$, and purchases nothing otherwise. Our main result is summarized in the following theorem.

Theorem 1. $\mathsf{UDP_{MIN}}$ *and* SMP *are* $\log^{1-\epsilon}(m+n)$*-hard to approximate for any constant* $\epsilon > 0$*, unless* $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{(\log n)^{\epsilon'}})$*, where* ϵ' *is some constant depending only on* ϵ*. Moreover, assuming that* $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}(n^{(\log n)^{\delta'}})$*, both problems are hard to approximate to within a factor of* $2^{\log^{1-\delta} n}$ *for any constant* δ*, where* δ' *is some constant depending only on* δ*.*

We next turn to special cases of both problems, denoted by $\mathsf{kUDP_{MIN}}$ and kSMP respectively, where the sizes of the sets S_c are bounded by k and prove the following theorem.

Theorem 2. *Let* $\epsilon > 0$ *be any constant. Then for infinitely many constants* k*, both* $\mathsf{kUDP_{MIN}}$ *and* kSMP *are* $k^{1/2-\epsilon}$*-hard to approximate unless* $\mathsf{P} = \mathsf{NP}$*.*

Finally we turn to the Tollbooth Pricing problem. In this problem, we are given a graph $G = (V, E)$, and a set of m simple paths $P_1, \ldots, P_m$, where each path P_c is associated with a customer c and a budget B_c. Once the price function $p : E \to \mathbb{R}$ on the edges is set, each customer c buys all edges on the path P_c if $\sum_{e\in P_c} p(e) \leq B_c$, and buys nothing otherwise. The goal is to compute the edge prices $p(e)$ so as to maximize the total profit. It is clear that Tollbooth Pricing is a special case of SMP, and notice that the number of items is $n = |E(G)|$.

We perform a reduction from the Unique Coverage problem to the Tollbooth Pricing. In the Unique Coverage problem, we are given a set U of elements and a family $\mathcal{S}$ of subsets of U as input. A solution is a sub-collection $\mathcal{S}' \subseteq \mathcal{S}$ of the input sets. We say that element $u \in U$ is *satisfied* by the solution if and only if it belongs to exactly one set in $\mathcal{S}'$. Our goal is to choose $\mathcal{S}'$ so as to maximize the number of satisfied elements. Demaine et. al. [8] have shown that for any arbitrarily small constant δ, if $\mathsf{NP} \not\subseteq \mathsf{BPTIME}(2^{n^\delta})$, then Unique Coverage is hard to approximate to within a factor of $\Omega(\log^\epsilon n)$, for some constant ϵ depending

on δ. They also showed that, under the assumption of Feige [11] that refuting random instances of 3SAT is hard, Unique Coverage is hard to approximate to within a factor of $\Omega(\log^{1/3-\epsilon} n)$ for any $\epsilon > 0$. We prove the following theorem:

Theorem 3. *If there is a factor α-approximation algorithm for the* Tollbooth Pricing *problem, for any approximation factor $\alpha \leq O(\log n)$, then there is a randomized $O(\alpha)$-approximation algorithm for the* Unique Coverage *problem.*

Combining this with the result of [8], we obtain the following corollary.

Corollary 1. *For any arbitrarily small constant δ, if* NP $\not\subseteq$ BPTIME(2^{n^δ}), Tollbooth Pricing *is hard to approximate to within a factor of $\Omega(\log^{\epsilon} n)$ for some constant ϵ depending on δ. Moreover, under Feige's random 3SAT assumption, this problem is hard to approximate to within a factor of $\Omega(\log^{1/3-\epsilon} n)$ for any $\epsilon > 0$.*

2 Hardness of $\mathsf{UDP}_{\mathsf{MIN}}$ and SMP

In this section we prove Theorems 1 and 2. We focus here on the $\mathsf{UDP}_{\mathsf{MIN}}$ problem only. The hardness results for SMP are obtained using similar ideas and appear in the full version of the paper.

We start with the following theorem, due to Trevisan [20]. Since we use slightly different parameters, we provide the proof in the full version.

Theorem 4. *Given an n-variable 3SAT formula φ, any sufficiently small constant $\epsilon > 0$ and any integer $\lambda > 0$, there is a randomized algorithm to construct a graph G with maximum degree at most $\Delta = 2^{\lambda \operatorname{poly}(\frac{1}{\epsilon})}$ such that w.h.p.:*

- *(*Yes-Instance*:) If φ is satisfiable, then G has an independent set of size $|V(G)|/\Delta^{\epsilon}$.*
- *(*No-Instance*:) If φ is not satisfiable, then G has no independent set of size $|V(G)|/\Delta^{1-\epsilon}$.*

The construction size is $|V(G)| = n^{\lambda \operatorname{poly}(\frac{1}{\epsilon})}$, and the reduction runs in time $n^{\lambda \operatorname{poly}(\frac{1}{\epsilon})}$. Moreover, the algorithm can be made deterministic with running time $2^{O(\Delta)} n^{\lambda \operatorname{poly}(\frac{1}{\epsilon})}$.

We remark that this theorem allows us to adjust parameter λ. To prove Theorem 1, we will use $\lambda = O(\log \log n)$, while we set $\lambda = O(1)$ for Theorem 2.

2.1 The Construction

Let $G = (V, E)$ be the instance of Maximum Independent Set obtained from Theorem 4, where the value of λ (and Δ) will be fixed later. We first define an intermediate instance of $\mathsf{UDP}_{\mathsf{MIN}}$, which is then converted into a final instance.

The intermediate instance is defined as follows. The set of items contains, for each vertex $v \in V$, for each index $y \in [\Delta]$, an item $i(v, y)$. That is, the set of items is $\mathcal{I} = \{i(v, y) \mid v \in V, y \in [\Delta]\}$.

Similarly, the set of customers contains, for each vertex $v \in V$, for each index $x \in [\Delta]$, a customer $c(v,x)$. That is, the set of customers is $\mathcal{C} = \{c(v,x) \mid v \in V, x \in [\Delta]\}$.

The item set $S_{c(v,x)}$ for the customer $c(v,x)$, contains the item $i(v,x)$, and additionally, for each neighbor u of vertex v in graph G, for each index $y \in [\Delta]$, item $i(u,y)$ belongs to $S_{c(v,x)}$. Formally, $S_{c(v,x)} = \{i(u,y) \mid (u,v) \in E, y \in [\Delta]\} \cup \{i(v,x)\}$. Notice that $\left|S_{c(v,x)}\right| \leq \Delta^2 + 1$ for all customers $c(v,x) \in \mathcal{C}$. Moreover for each item $i(v,y) \in \mathcal{I}$, there are at most $\Delta^2 + 1$ customers $c' \in \mathcal{C}$ such that $i(v,y) \in S_{c'}$.

We partition the set $\mathcal{C}$ of customers into Δ subsets $\mathcal{C}_1, \ldots, \mathcal{C}_\Delta$, such that for each $1 \leq h \leq \Delta$, set $\mathcal{C}_h$ contains customers $c(v,h)$ for all $v \in V$. Finally, for each $1 \leq h \leq \Delta$, each customer $c \in \mathcal{C}_h$ is assigned budget $B_c = 1/2^h$.

This finishes the definition of the intermediate instance. For convenience, we call the customers in set $\mathcal{C}$ *virtual customers*. In our final instance, we replace each virtual customer with a number of new customers.

In order to define the final instance, for each $1 \leq h \leq \Delta$, we replace each virtual customer $c \in \mathcal{C}_h$ with a set $\mathcal{G}(c) = \left\{c(1), \ldots, c(2^h)\right\}$ of 2^h identical new customers. Each new customer $c(h')$, for $1 \leq h' \leq 2^h$ has budget $B_{c(h')} = B_c$ and $S_{c(h')} = S_c$. The final set of customers is $\mathcal{C}' = \bigcup_{c \in \mathcal{C}} \mathcal{G}(c)$ and the final set of items remains unchanged, $\mathcal{I}' = \mathcal{I}$. The number of customers in the final instance is $\tilde{m} = |\mathcal{C}'| = O(2^\Delta |\mathcal{C}|) = |V| \cdot \Delta \cdot 2^{O(\Delta)} = |V| \cdot 2^{O(\Delta)}$, while the number of items is $\tilde{n} = |V| \cdot \Delta$. Moreover, for each customer $c \in \mathcal{C}'$, we have $|S_c| \leq \Delta^2 + 1$. This completes the construction description.

2.2 Analysis

We analyze the construction in the following two lemmas.

Lemma 1. *In the* YES-INSTANCE*, there is a solution to the* $\mathsf{UDP_{MIN}}$ *problem instance whose value is at least* $|V|\Delta^{1-\epsilon}$.

Proof. Let $U \subseteq V$ be a maximum independent set of size $|V|/\Delta^\epsilon$ in G. We set the prices of the items $i(u,y) \in \mathcal{I}'$ as follows. If $u \notin U$, then the price of $i(u,y)$ is set to ∞. Otherwise, if $u \in U$, then we set the price of $i(u,y)$ to $1/2^y$. Notice that, since $|U| \cdot \Delta \geq |V| \cdot \Delta^{1-\epsilon}$, there are $|V| \cdot \Delta^{1-\epsilon}$ items of finite prices. We now show that this solution has value at least $|V| \cdot \Delta^{1-\epsilon}$.

Indeed, for each vertex $u \in U$ and an index $y \in [\Delta]$, consider the virtual customer $c' = c(v,y) \in \mathcal{C}_y$. Notice that $S_{c'}$ contains item $i(v,y)$ whose price is $1/2^y$, but all other items in $S_{c'}$ have price ∞. Therefore, each customer $c \in \mathcal{G}(c')$ buys the item $i(v,y)$, and pays $1/2^y$ for it. The total profit collected from customers in $\mathcal{G}(c')$ is 1, and so the total profit collected from all customers is at least $|U|\Delta \geq |V| \cdot \Delta^{1-\epsilon}$.

Lemma 2. *In the* NO-INSTANCE*, the value of the optimal solution is at most* $O(|V| \cdot \Delta^\epsilon)$.

Proof. Let p^* be an optimal solution, and let r^* be its revenue. We first argue that we can assume w.l.o.g. that for each item $i \in \mathcal{I}'$, either $p^*(i) \in \left\{1/2^h \mid 1 \leq h \leq \Delta\right\}$, or $p^*(i) = \infty$.

Indeed, suppose there is an item $i \in \mathcal{I}'$ with $p^*(i) \in (1/2^h, 1/2^{h-1})$. Then any customer who buys item i must have budget at least $1/2^{h-1}$, so increasing $p^*(i)$ to $1/2^{h-1}$ does not affect these customers, and may only increase the revenue. Therefore, from now on we assume that for each item $i \in \mathcal{I}'$, $p^*(i) \in \left\{1/2^h \mid 1 \leq h \leq \Delta\right\} \cup \{\infty\}$.

Notice that for each virtual customer $c \in \mathcal{C}$, all customers in $\mathcal{G}(c)$ contribute the same amount to the total revenue. Let k_c denote this amount. We now let $\mathcal{C}^* \subseteq \mathcal{C}$ be the set of virtual customers for which $k_c = B_c$. Equivalently,

$$\mathcal{C}^* = \left\{ c \in \mathcal{C} : \min_{i \in S_c} \{p^*(i)\} = B_c \right\}$$

Claim. The customers in $\bigcup_{c' \in \mathcal{C}^*} \mathcal{G}(c')$ contribute at least $r^*/2$ to the total revenue.

Due to space limitation, the proof of this claim appears in the full version. Notice that $|\mathcal{C}^*| \geq r^*/2$, since for each virtual customer $c \in \mathcal{C}^*$, the total budget of all customers in $\mathcal{G}(c)$ is 1.

From now on, we focus on finding an independent set U in graph G of size at least $(r^*/2 - |V|)/\Delta$ from $\mathcal{C}^*$. Since in the No-Instance, G does not contain an independent set of size more than $|V|/\Delta^{1-\epsilon}$, this implies that $(r^*/2 - |V|)/\Delta \leq |V|/\Delta^{1-\epsilon}$, and hence $r^* \leq O(|V|\,\Delta^{\epsilon})$.

We construct an independent set $U \subseteq V(G)$, together with a partition $(\mathcal{C}^1, \mathcal{C}^2)$ of $\mathcal{C}^*$, as follows. Start with $U, \mathcal{C}^1, \mathcal{C}^2 = \emptyset$. We then perform Δ iterations, where in iteration y, we consider each virtual customer $c(v, y)$ in $\mathcal{C}^* \cap \mathcal{C}_y$, and do the following:

- If vertex v is already in U, we add virtual customer $c(v, y)$ into $\mathcal{C}^1$.
- If vertex v is not in U and $U \cup \{v\}$ remains an independent set, we add vertex v to set U and add $c(v, y)$ to $\mathcal{C}^1$. We say that $c(v, y)$ is *responsible* for adding vertex v into U.
- Otherwise, $v \notin U$, but there is a vertex $u \in U$ such that $(u, v) \in E(G)$. We add $c(v, y)$ to $\mathcal{C}^2$ in this case and say that vertex u prevents the algorithm from adding v into U.

In the end, when all customers in $\mathcal{C}^*$ are processed, each virtual customer in $\mathcal{C}^*$ is added to either $\mathcal{C}^1$ or $\mathcal{C}^2$, so $\mathcal{C}^* = \mathcal{C}^1 \cup \mathcal{C}^2$. Moreover, for each virtual customer $c(v, y)$ in $\mathcal{C}^1$, the corresponding vertex v belongs to U, so $|U| \geq |\mathcal{C}^1|/\Delta$. The following claim will complete the proof of the lemma.

Claim. $|\mathcal{C}^2| \leq |V|$, and so $|U| \geq |\mathcal{C}^* \setminus \mathcal{C}^2|/\Delta \geq (r^*/2 - |V|)/\Delta$.

Proof. It is sufficient to show that for each vertex $v \in V$, no two virtual customers $c(v, y), c(v, y')$ with $y \neq y'$ belong to $\mathcal{C}^2$. Assume otherwise, and let $c(v, y), c(v, y') \in \mathcal{C}^2$. By our construction, we have $c(v, y) \in \mathcal{C}_y$ and $c(v, y') \in \mathcal{C}_{y'}$. Assume w.l.o.g. that $y < y'$, so $c(v, y)$ was processed before $c(v, y')$.

Let $u \in U$ be a vertex such that $(u, v) \in E(G)$, and vertex u prevents the algorithm from adding v to set U. Let $c(u, x)$ be the customer responsible for adding u to U. Then $c(u, x)$ was processed before $c(v, y)$, and so $x \leq y < y'$.

Notice that the item $i(v, y')$ belongs to $S_{c(u,x)}$. The price of $i(v, y')$ then must be set to at least $B_{c(u,x)} = 1/2^x > 1/2^{y'} = B_{c(v,y')}$, since otherwise the customers in $\mathcal{G}(c(u, x))$ would have paid below $B_{c(u,x)}$ for item $i(v, y')$, contradicting the fact that $c(u, x) \in \mathcal{C}^*$. But then customer $c(v, y')$ must buy some item $i' \neq i(v, y')$. Assume that $i' = i(w, z)$. Then w must be a neighbor of v in G, $w \neq v$, and so $i' \in S_{c(v,y)}$ must hold. But the price of i' must be $B_{c(v,y')} = 1/2^{y'} < 1/2^y = B_{c(v,y)}$, and so the customers in $\mathcal{G}(c(v, y))$ should have paid below $B_{c(v,y)}$ for item i', contradicting the fact that $c(v, y) \in \mathcal{C}^*$.

Hardness factors: The gap between Yes-Instance and No-Instance costs is $\Delta^{1-2\epsilon}$, while the number of customers in the instance is $\tilde{m} = |V(G)| \cdot 2^{O(\Delta)}$, and the number of items is $\tilde{n} = |V(G)| \cdot \Delta$.

We first prove Theorem 1. We choose the parameter $\lambda = O(\log \log n)$ such that $\Delta = (\log n)^b$, where $b > \frac{1}{2\epsilon}$. The hardness factor then becomes $g = \Delta^{1-2\epsilon} \geq \log^{b-1} n$, while $\tilde{m} + \tilde{n} = |V(G)|2^{O(\Delta)} \leq 2^{O(\Delta \log n)} \leq 2^{\log^{b+2} n} \leq 2^{g^{1+O(\epsilon)}}$. Taking logarithm on both sides will give $g = \log^{1-O(\epsilon)}(\tilde{m} + \tilde{n})$. The deterministic reduction takes time $2^{O(\Delta)} = n^{(\log n)^{f(\epsilon)}}$ for some function f, so we have proved the first part of Theorem 1.

To prove the second part, we use the randomized version of Theorem 4, and choose $\lambda = (\log n)^b$ for some large constant b, while ϵ is set to be any small enough constant for which Theorem 4 works. In this case, we have $\Delta = 2^{O((\log n)^b)}$ and $\tilde{n} \leq |V(G)|\Delta \leq 2^{(\log n)^{b+2}}$, while $g = \Delta^{1-2\epsilon} \geq 2^{O((\log n)^b)}$. It is easy to check that $g \geq 2^{\log^{1-O(1/b)} \tilde{n}}$, as desired. Since we use the randomized reduction, the running time of the reduction is $2^{(\log n)^{O(b)}}$, and so the result holds under the assumption that $\mathsf{NP} \not\subseteq \mathsf{ZPTIME}(n^{(\log n)^{O(b)}})$.

To prove Theorem 2, we choose λ in Theorem 4 to be any sufficiently large constant. Denote by $k = \max_{c \in \mathcal{C}'} |S_c|$. Since the construction guarantees that $k \leq 2\Delta^2$, we have the hardness factor of $\Delta^{1-2\epsilon} \geq k^{1/2-\epsilon}$. In this case, the deterministic reduction only takes polynomial time, so this hardness result holds under the assumption that $\mathsf{P} \neq \mathsf{NP}$.

3 Tollbooth Pricing

In this section we prove Theorem 3. It will be useful to introduce the notion of fractional coverage and show how to convert fractional coverage to an integral one. Given an instance of Unique Coverage and a fractional solution that assigns a non-negative weight $w(S)$ to every set $S \in \mathcal{S}$, we say that an element $u \in U$ is *fractionally covered* if and only if $1/4 \leq \sum_{S:u \in S} w(S) \leq 1$. We argue that any good fractional coverage can be converted into a good integral coverage with a constant loss in the solution value. The proof of the following lemma appears in the full version of the paper.

Lemma 3. *There is an efficient randomized algorithm, that, given a fractional solution of value βn to any instance of the* Unique Coverage *problem, w.h.p. finds an integral solution of value $\Omega(\beta n)$ to the* Unique Coverage *instance.*

3.1 Construction

Let $(U, \mathcal{S})$ be an instance of Unique Coverage, where $|U| = n$ and $|\mathcal{S}| = m$. We construct an instance of Tollbooth Pricing as follows. Graph $G = (V, E)$ consists of $m+1$ vertices $v_0, \ldots, v_m$. Let $h = \lceil \log m \rceil$. For each consecutive pair (v_{i-1}, v_i) of vertices, $0 < i \leq m$, we add $h+1$ parallel edges $e_0^i, \ldots, e_h^i$. These edges are viewed as representing the set $S_i \in \mathcal{S}$. We now define the set of paths (or customers) in the graph. All paths start from v_0 and end at v_m. For each element $u \in U$, for each $j : 1 \leq j \leq h$, we have a set $\mathcal{P}(u, j)$ of 2^{h-j} paths. The budget of each path in $\mathcal{P}(u, j)$ is 2^j, the source vertex is v_0, and the sink is v_m. Each path in $\mathcal{P}(u, j)$ consists of edges $e_{i_1}^1, e_{i_2}^2, \ldots, e_{i_m}^m$, where for all $1 \leq \ell \leq m$, if $u \in S_\ell$ then $i_\ell = j$, or otherwise $i_\ell = 0$. This completes the description of the construction. Notice that the total budget is $\mathcal{B} = nh2^h$. Let $\tilde{m}$ and $\tilde{n}$ denote the number of customers (i.e. the number of paths) and items, respectively. Notice that $\tilde{m} \leq O(nm \log m)$, and $\tilde{n} \leq nh \leq O(n \log m) \leq O(n^2)$, since we can assume w.l.o.g. that $|\mathcal{S}| \leq 2^n$.

3.2 Analysis

The analysis consists of two parts. First we show that if there is a solution to Unique Coverage that satisfies a β-fraction of the elements, then there is a solution to Tollbooth Pricing of value at least $\beta \cdot \mathcal{B}$. In the second part, we show an efficient randomized algorithm, that, given any solution to Tollbooth Pricing instance G of value $\alpha \cdot \mathcal{B}$, w.h.p. finds a solution to the Unique Coverage problem that satisfies $\Omega(\alpha n)$ elements.

Lemma 4. *If there is a solution to the* Unique Coverage *instance $(U, \mathcal{S})$ that satisfies at least βn-elements, then there is a solution to the* Tollbooth Pricing *instance of value $\beta\mathcal{B}$.*

Proof. Let $\mathcal{S}' \subseteq \mathcal{S}$ be a solution to the Unique Coverage problem, and let $U' \subseteq U$ be the set of elements uniquely covered by $\mathcal{S}'$, $|U'| \geq \beta n$. For each $S_i \in \mathcal{S}'$, for each $j : 1 \leq j \leq h$, we set the price of the edge e_j^i to 2^j. The prices of all other edges (including the edges e_0^i for all i) are set to 0. For each $u \in U'$ and $j : 1 \leq j \leq h$, we consider the revenue collected from the paths in $\mathcal{P}(u, j)$. Let S_i be the set that uniquely covers u in the solution. Then for each path in $\mathcal{P}(u, j)$, exactly one edge e_j^i on the path has a non-zero price. This price is 2^j - the same as the budget of the path, while all other edges have price 0. Therefore, each such path contributes 2^j to the solution value, and the total contribution of the paths in $\mathcal{P}(u, j)$ is 2^h. This implies the lemma.

Lemma 5. *There is an efficient randomized algorithm, that, given any solution to the* Tollbooth Pricing *instance G of value $\alpha\mathcal{B}$, w.h.p. finds a solution to the* Unique Coverage *instance $(U, \mathcal{S})$ that satisfies $\Omega(\alpha n)$ of the elements.*

Proof. Let $p^* : E \to \mathbb{R}_{\geq 0}$ be any solution of value $\alpha\mathcal{B}$ to the Tollbooth Pricing problem. Let $\mathcal{P}_1$ be the set of paths, such that each $P \in \mathcal{P}_1$ contributes at least

half of its budget to the solution. Our first observation is that the profit collected from the paths in $\mathcal{P}_1$ must be at least $\alpha\mathcal{B}/2$ (otherwise, we can multiply the price of each edge by a factor of two and get a better solution). From now on, we will only focus on paths in $\mathcal{P}_1$ and we will discard all other paths. We say that a path $P \in \mathcal{P}_1$ is of type 1 if at least half the cost it pays goes to edges in set $E_0 = \{e_0^i : 1 \le i \le m\}$, and it is of type 2 otherwise. Let $\mathcal{P}'$ and $\mathcal{P}''$ denote the set of paths of type 1 and 2 respectively. We distinguish between two cases.

Case 1: Paths of type 1 contribute at least $\alpha\mathcal{B}/4$ to the solution value. We claim that in this case the solution value is at most $O(\mathcal{B}/\log m)$, and therefore it is sufficient to find a solution to Unique Coverage instance that satisfies a $\Omega(1/\log m)$-fraction of the elements. We then show an algorithm to find such a solution.

Indeed, consider some element $u \in U$. Recall that, for all j, every path in the sets $\mathcal{P}(u,j)$ traverses all edges in the set $E_0(u) = \{e_0^i : u \notin S_i\}$, and these are the only edges from E_0 traversed by these paths. Let $C_u = \sum_{e \in E_0(u)} p^*(e)$ be the total price of these edges. A path $P \in \mathcal{P}(u,j)$ can belong to $\mathcal{P}'$ only if $2^j/4 \le C_u \le 2^j$. This means that there are at most 3 values of $j : 1 \le j \le h$ for which $\mathcal{P}(u,j) \cap \mathcal{P}' \neq \emptyset$, so for each $u \in U$, the paths in set $\bigcup_{j=1}^{h} \mathcal{P}(u,j)$ only contribute at most an $O(1/h) = O(1/\log m)$-fraction of their total budget to the solution. Therefore, the solution value is at most $O(\mathcal{B}/h) = O(\mathcal{B}/\log m)$. Now we show an algorithm for the Unique Coverage problem instance that satisfies an $\Omega(1/\log m)$-fraction of the elements. From Lemma 3, it is enough to construct a fractional solution of value $\Omega(n/\log m)$. For each element $u \in U$, let $\delta(u)$ be the number of sets in $\mathcal{S}$ to which element u belongs. We partition the elements into $h = \lceil \log m \rceil$ classes $C_1, \ldots, C_h$ where class C_j contains elements u with $2^j \le \delta(u) \le 2^{j+1}$. Let j^* be the class containing the maximum number of elements, so $|C_{j^*}| \ge \Omega(n/\log m)$. We set the weight of every set S to be $w(S) = 1/2^{j^*+1}$. This ensures that all elements in C_{j^*} are fractionally covered. Applying Lemma 3, we obtain an integral solution of value $\Omega(n/\log m)$.

Case 2: Assume now that the paths in $\mathcal{P}''$ contribute at least $\alpha\mathcal{B}/4$ to the solution value. Let r'' denote the total revenue collected from these paths by edges in $E_1 = E \setminus E_0$. Then we have that $r'' \ge \Omega(\alpha\mathcal{B}) = \Omega(\alpha n h 2^h)$. Notice that by the definition of set $\mathcal{P}''$, each path $P \in \mathcal{P}''$ pays at least $1/4$ of its budget for the edges in set E_1 that lie on path P.

We now partition the paths in $\mathcal{P}''$ into sets $\mathcal{P}''_1, \ldots, \mathcal{P}''_h$ where set $\mathcal{P}''_j$ contains all type-2 paths whose budget is 2^j. Let j^* be the index for which the profit contributed by the paths in $\mathcal{P}''_{j^*}$ is maximized. This profit is at least $\alpha n 2^h$.

We say that element u is good if $2^{j^*}/4 \le \sum_{i:u \in S_i} p^*(e^i_{j^*}) \le 2^{j^*}$. From the above arguments, for each path $P \in \mathcal{P}''$, if $P \in \mathcal{P}(u,j^*)$, then the corresponding element u must be good. Moreover, if u is good, then all paths in $\mathcal{P}(u,j^*)$ belong to $\mathcal{P}''_{j^*}$. Therefore, at least $\Omega(\alpha n)$ of the elements in U must be good. We now define a fractional solution to the Unique Coverage problem, where every the weight of every set $S_i \in \mathcal{S}$ is set to $w(S_i) = p(e^i_{j^*})/2^{j^*}$. Notice that all

good elements are fractionally covered, thus giving us a fractional solution where $\Omega(\alpha n)$ elements are satisfied. We finally invoke Lemma 3 to complete the proof.

Acknowledgement. The first author thanks Khaled Elbassioni for introducing him to pricing problems and for explaining the differences between various pricing models. He also thanks Danupon Nanongkai and Khaled Elbassioni for useful discussions about $\mathsf{UDP}_{\mathsf{MIN}}$ and SMP.

References

1. Aggarwal, G., Feder, T., Motwani, R., Zhu, A.: Algorithms for Multi-product Pricing. In: Díaz, J., Karhumäki, J., Lepistö, A., Sannella, D. (eds.) ICALP 2004. LNCS, vol. 3142, pp. 72–83. Springer, Heidelberg (2004)
2. Balcan, M.-F., Blum, A.: Approximation algorithms and online mechanisms for item pricing. Theory of Computing 3(1), 179–195 (2007)
3. Briest, P.: Uniform Budgets and the Envy-Free Pricing Problem. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) ICALP 2008, Part I. LNCS, vol. 5125, pp. 808–819. Springer, Heidelberg (2008)
4. Briest, P., Krysta, P.: Buying cheap is expensive: hardness of non-parametric multi-product pricing. In: SODA, pp. 716–725 (2007)
5. Chen, N., Deng, X.: Envy-Free Pricing in Multi-item Markets. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010, Part II. LNCS, vol. 6199, pp. 418–429. Springer, Heidelberg (2010)
6. Chen, N., Ghosh, A., Vassilvitskii, S.: Optimal envy-free pricing with metric substitutability. SIAM J. Comput. 40(3), 623–645 (2011)
7. Cheung, M., Swamy, C.: Approximation algorithms for single-minded envy-free profit-maximization problems with limited supply. In: FOCS, pp. 35–44. IEEE Computer Society (2008)
8. Demaine, E.D., Feige, U., Hajiaghayi, M., Salavatipour, M.R.: Combination can be hard: Approximability of the unique coverage problem. SIAM J. Comput. 38(4), 1464–1483 (2008)
9. Elbassioni, K., Raman, R., Ray, S., Sitters, R.: On Profit-Maximizing Pricing for the Highway and Tollbooth Problems. In: Mavronicolas, M., Papadopoulou, V.G. (eds.) SAGT 2009. LNCS, vol. 5814, pp. 275–286. Springer, Heidelberg (2009)
10. Elbassioni, K., Sitters, R., Zhang, Y.: A Quasi-PTAS for Profit-Maximizing Pricing on Line Graphs. In: Arge, L., Hoffmann, M., Welzl, E. (eds.) ESA 2007. LNCS, vol. 4698, pp. 451–462. Springer, Heidelberg (2007)
11. Feige, U.: Relations between average case complexity and approximation complexity. In: STOC, pp. 534–543 (2002)
12. Gamzu, I., Segev, D.: A Sublogarithmic Approximation for Highway and Tollbooth Pricing. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010, Part I. LNCS, vol. 6198, pp. 582–593. Springer, Heidelberg (2010)
13. Grandoni, F., Rothvoß, T.: Pricing on paths: A ptas for the highway problem. In: SODA, pp. 675–684 (2011)
14. Guruswami, V., Hartline, J.D., Karlin, A.R., Kempe, D., Kenyon, C., McSherry, F.: On profit-maximizing envy-free pricing. In: SODA, pp. 1164–1173 (2005)

15. Hartline, J.D., Koltun, V.: Near-Optimal Pricing in Near-Linear Time. In: Dehne, F., López-Ortiz, A., Sack, J.-R. (eds.) WADS 2005. LNCS, vol. 3608, pp. 422–431. Springer, Heidelberg (2005)
16. Khandekar, R., Kimbrel, T., Makarychev, K., Sviridenko, M.: On Hardness of Pricing Items for Single-Minded Bidders. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 202–216. Springer, Heidelberg (2009)
17. Khot, S.: On the power of unique 2-prover 1-round games. In: Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing, STOC 2002, pp. 767–775. ACM, New York (2002)
18. Rusmevichientong, P.: A non-parametric approach to multi-product pricing: Theory and application. Ph. D. thesis, Stanford University (2003)
19. Rusmevichientong, P., Van Roy, B., Glynn, P.W.: A nonparametric approach to multiproduct pricing. Oper. Res. 54, 82–98 (2006)
20. Trevisan, L.: Non-approximability results for optimization problems on bounded degree instances. In: STOC, pp. 453–461 (2001)

Online Flow Time Scheduling in the Presence of Preemption Overhead

Ho-Leung Chan*, Tak-Wah Lam**, and Rongbin Li

The University of Hong Kong, Hong Kong
{hlchan,twlam,rbli}@cs.hku.hk

Abstract. This paper revisits the online problem of preemptive scheduling to minimize the total flow time. Previous theoretical results often assume that preemption is free, which is not true for most systems. This paper investigates the complexity of the problem when a processor has to perform a certain amount of overhead (extra work) before it resumes the execution of a job preempted before. Such overhead causes delay to all unfinished jobs. In this paper we first consider single-processor scheduling. We show that no online algorithm can be competitive for total flow time in the presence of preemption overhead (note that the well-known online algorithm SRPT is 1-competitive when preemption overhead is zero). We then consider resource augmentation and show a simple algorithm that is $(1+\epsilon)$-speed $(1+\frac{1}{\epsilon})$-competitive for minimizing total flow time on a single processor. We also extend the result to the multiprocessor setting.

1 Introduction

This paper is concerned with online scheduling of jobs that can be preempted but with a certain overhead, which is modeled as extra work for a processor that delays the execution of jobs. This paper studies how to use preemption effectively while minimizing the delay. Specifically, we consider schedules that minimize the total flow time of jobs, where the flow time of a job is defined as the completion time of the job minus its arrival time. An algorithm is said to be c-competitive if for any input sequence, its total flow time is at most c times that of the optimal schedule.

It is well-known that preemption is undesirable because it incurs overhead due to context switching. Online flow time scheduling has been studied intensively for two decades; it is perhaps surprising that not much theoretical results have been known on the effect of preemption overhead. Most previous work assumes preemption is free, i.e., a job can be interrupted and resumed without any extra cost (e.g., [8, 11]). In this setting, the simple algorithm SRPT, which always schedules the job with the minimum remaining size, is 1-competitive and always minimizes the total flow time. Bartal et al. [3] considered a model where the flow time and preemption overhead are accounted separately, i.e., a processor

* The research is partially supported by GRF Grant HKU710210E.
** The research is partially supported by HKU Grant 201109176197.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 85–97, 2012.
© Springer-Verlag Berlin Heidelberg 2012

can switch from one job to another job in zero time but it costs $k > 0$ units of "abstract" overhead. The objective is to minimize the total flow time plus the total preemption overhead. A competitive online algorithms has been given for this objective. However, this model might have oversimplified the charging of preemption overhead. In reality, when a job is preempted for later execution, the overhead increases the flow time of all unfinished jobs, and the actual "cost" of the overhead is sensitive to the number of unfinished jobs.

In this paper, we adopt a more natural model studied by Heydari et al. [10], in which preemption overhead is modeled as extra h units of work for a processor and the time incurred directly increases the flow time of all unfinished jobs. The objective is simply to minimize the total flow time of all jobs. More specifically, the h units of overhead is due to the re-configuration of the runtime environment before a job can be processed; when a processor runs a job for the first time or resumes a job preempted before, the processor has to perform h units of extra work. We assume that the processing of overhead can also be interrupted, but the entire overhead needs to be processed again upon next resumption.

Based on the above model that charges preemption overhead into flow time, Heydari et al. [10] proposed an online algorithm for minimizing the total flow time and they evaluated it by simulations and experiments. Their algorithm is based on a simple greedy strategy that recomputes the currently best schedule whenever a new job arrives, assuming no more job will arrive in the future. It is easy to see that this algorithm is not competitive. In this paper we indeed show that any online algorithm is $\Omega(n^{1/4})$-competitive. Yet, with resource augmentation, we give a simple algorithm QSRPT that is $(1+\epsilon)$-speed $(1+\frac{1}{\epsilon})$-competitive for any $\epsilon > 0$. Note that the algorithm of Heydari et al. [10] is not s-speed $O(1)$-competitive for any $s < 1.5$.

It is worth-mentioning that, for other objectives, online scheduling with preemption overhead is relatively easier. Liu and Cheng [13] considered the same online setting as in this paper but the objective of minimizing the total completion time. They gave a 1.5625-competitive algorithm. We can in fact adapt the analysis of QSRPT to show that, without resource augmentation, QSRPT is $(1+\epsilon+O(1/n))$-competitive for total completion time, for any $\epsilon > 0$ (Details are left to the full paper). Hence, the competitive ratio can be arbitrarily close to 1 with sufficient jobs. Another interesting objective is maximum flow time. Note that FIFO, which is non-preemptive in nature, is 1-competitive for minimizing the maximum flow time no matter the preemption overhead is zero or finite.

This paper presents three results on flow time scheduling with preemption overhead, the first two are based on single processor, while the third for multiprocessors. Details are as follows:

• We show that the problem has unbounded competitive ratio even if the preemption overhead is small relative to the job size. In particular, we show that any algorithm, without resource augmentation, is $\Omega((\frac{\delta}{1+\delta})^{1/4}n^{1/4})$-competitive, where n is the number of jobs in the input and δ is the ratio of preemption overhead to the minimum job size. Notice that in Bartal et al'.s model, the

separation of flow time and overhead accounting makes scheduling easier and allows an $O(\sqrt{\delta})$-competitive algorithm without extra resource [3].

• Given the strong lower bound, we consider the resource augmentation model where the online algorithm is given an s-speed processor, for some $s \geq 1$. Formally, an s-speed processor can process s units of preemption overhead or the work of the jobs in one unit of time. Since preemption incurs overhead, a natural idea is to process a job for at least a *quantum* of work before preempting it. We integrate this idea into SRPT, that is, we select the job with the minimum remaining size after completing each quantum. We call the resulting algorithm QSRPT. Notice that at selection time, if a job not being processed currently has a size slightly smaller than the job being processed, QSRPT will still preempt the current job and switch to the smaller job. This is true even if we can complete the current job earlier than the smaller job which requires preemption overhead. While this looks unnatural, insisting on the smaller job at selection time keeps the analysis simple. In fact, we are not able to analyze the algorithm which selects the job with the earliest completion time. We show that QSRPT is $(1+\epsilon)$-speed $(1+\frac{1}{\epsilon})$-competitive, for any $\epsilon > 0$. Notice that if the jobs are non-preemptive, no algorithm can be $O(1)$-competitive with an $O(1)$-speed processor [15].

• When there is more than one processor, resuming a job may require different amount of overhead depending on where the job is resumed. In this case, we assume that h units of overhead is needed for starting a job for the first time or resuming a preempted job on the same processor; and $h' \geq h$ units of overhead for resuming a job on a different processor. To distinguish the two types of overhead, we call h the *preemption overhead* and h' the *migration overhead*. Interestingly, we notice that there is a non-migratory algorithm IMD which is $(1+\epsilon)$-speed $O(\frac{1}{\epsilon})$-competitive on total flow time when preemption overhead is assumed to be zero [2]. IMD assigns a job to a processor immediately when a job is released. When preemption overhead is non-zero, we use IMD to assign jobs to the processors and then schedule each processor by QSRPT. We can easily show that IMD plus QSPRT is $(1+\epsilon)^2$-speed $O(\frac{1}{\epsilon^2})$-competitive for total flow time. Interestingly, we can improve the ratio by a more careful and direct analysis of QSRPT (instead of extending the previous analysis of IMD). This shows that QSRPT is indeed $(1+\epsilon)$-speed $O(\frac{1}{\epsilon})$-competitive.

Related Work. Scheduling with preemption overhead was also studied in the offline context. Some approximation results have been obtained. Liu and Cheng [12] considered single processor scheduling where the jobs have different release times, preemption overheads and delivery times. The objective is to minimize the maximum completion time plus delivery time. They showed the problem is NP-hard and gave a PTAS for the problem. Schuurman and Woeginger [16] considered parallel processor scheduling where all jobs are released at time 0. The objective is to minimize the maximum completion time. They gave a 4/3-approximate algorithm for jobs with different preemption overheads, and a PTAS for jobs with identical preemption overhead. Chen [4] and Monma and Potts [14] studied parallel processor scheduling where jobs have different types and

preemption overhead is incurred only when the processor resumes a job with a type different from the current job. They gave $O(1)$-approximate algorithms to minimize the maximum completion time.

A perhaps related problem is non-preemptive scheduling with setup overhead depending on the types of the jobs. Divakaran and Saks [7] considered online algorithms to minimize the maximum flow time on a single processor. They gave an $O(1)$-competitive algorithm. Other results are based on the offline setting (e.g., [5,6,9]). See the survey by Allahverdi et. al. [1].

Notations. We consider online scheduling on $m \geq 1$ processors. Jobs arrive online. Each job j has a release time $r(j)$ and size $p(j)$. Recall that h denotes the preemption overhead, and h' the migration overhead. Note that each time before starting or resuming a job j, a preemption (or migration) overhead must be performed. For the analysis reason, this preemption (or migration) overhead is called a preemption (or migration) overhead of the job j. Following, the *work* of j always refers to the $p(j)$ units work of j. The *overhead* of j refers to preemption or migration overhead of j. We say that an algorithm is processing the work of j if it is processing the $p(j)$ units work of j. It is processing j if it is processing work or overhead of j.

2 Lower Bound

This section shows that even on a single processor, no algorithm is competitive without resource augmentation.

Theorem 1. *Consider single processor scheduling to minimize total flow time with preemption overhead $h > 0$. Any algorithm is $\Omega((\frac{\delta}{1+\delta})^{1/4} n^{1/4})$-competitive, where n is the number of jobs and $\delta = \frac{h}{p_{min}}$, where p_{min} is the minimum job size.*

Let Alg be any online algorithm. Below we focus on input instances of which the minimum job size is 1 and $\delta = \frac{h}{p_{min}} = h$. For any integer $P \geq 1$, we give an instance with $n = O(P^4)$ jobs such that the total flow time of Alg is at least P times that of the optimal schedule. To obtain a lower bound result for arbitrary minimum job size (Theorem 1), we can simply scale the instance accordingly.

The instance is based on a simple idea. We release a big job and a stream of small jobs. If Alg continues processing the big job, many small jobs are delayed and the flow time is immediately bad. Otherwise, if Alg preempts the big job, it incurs some extra preemption overhead which can be avoided by the optimal schedule. This extra overhead can be accumulated and the flow time of Alg becomes $w(1)$ times the optimal. Furthermore, the rate at which the extra overhead is accumulated decreases with $\delta = \frac{h}{p_{min}}$, and the result follows.

Lower Bound Instance. Assume the minimum job size is 1 and $\delta = \frac{h}{p_{min}} = h$. Let $P \geq 1$ be any integer. Let $w = (1+\delta) + (1+\delta) + \frac{\delta}{P}$. We release the following two *streams* of jobs. At each time $t = 0, w, 2w, \ldots$, we release a *small* job of size 1. At each time $t = 0, Pw, 2Pw, \ldots$, we release a *big* job of size $P(1+\delta)$. Let

t_o be the earliest time such that Alg processes a big job non-preemptively for $\delta+\frac{P(1+\delta)}{2}$ units of time. If no such event occurs until $\frac{P^3(1+\delta)w}{\delta}$, let $t_o = \frac{P^3(1+\delta)w}{\delta}$. We terminate the above two steams at time t_o. Starting from t_o, we release a small job of size 1 at time $t_o + (i-1)(1+\delta)$ for $i = 1, 2, \ldots, \frac{P^4(1+\delta)}{\delta}$.

Fact 1. *Let n be the number of jobs released. Then $n \leq 3\frac{P^4(1+\delta)}{\delta}$.*

Proof. During $[0, t_o)$, we release at most $\frac{P^3(1+\delta)w}{\delta}/w$ small jobs and $\frac{P^3(1+\delta)w}{\delta}/(Pw)$ big jobs. From t_o, we release $\frac{P^4(1+\delta)}{\delta}$ small jobs. □

Below we show that the flow time of Alg is $\Omega(P)$ times that of the optimal, hence it is $\Omega((\frac{\delta}{1+\delta})^{1/4}n^{1/4})$-competitive.

Consider the jobs released by the two streams during $[0, Pw)$. By first processing the big job without preemption and then each small jobs, the total amount of time required is $(\delta + P(1+\delta)) + P(1+\delta) = Pw$. Hence, it is possible to complete all the jobs by time Pw. In particular, at time t_o, let t'_o be the largest multiple of Pw that is less than or equal to t_o. At time t'_o, the optimal schedule can complete all jobs released before time t'_o. From time t'_o to time t_o, it can simply schedule the new released small jobs. Hence, the optimal schedule can have at most one big job and one small job unfinished at time t_o. However, we can show that Alg has many jobs remaining at time t_o.

Lemma 1. *Consider all jobs remaining in Alg at time t_o. Either (i) the total amount of remaining work for small jobs is at least $\frac{P}{6}$, or (ii) the total amount of remaining work for big jobs is at least $\frac{P^2(1+\delta)}{6}$.*

Proof. There are two cases depending on whether t_o is the first time that a big job is processed non-preemptively for at least $\delta + \frac{P(1+\delta)}{2}$ units of time. If it is the case, the number of small jobs released during this period is at least $(\delta + \frac{P(1+\delta)}{2})/w \geq \frac{P(1+\delta)}{2}/(2+3\delta) \geq P/6$. Since all these small jobs are not processed, the total remaining work for small jobs is at least $P/6$.

Otherwise, we have $t_o = \frac{P^3(1+\delta)w}{\delta}$. Let R_1 and R_2 be the amount of remaining work for small and big jobs, respectively. Then, during $[0, t_o]$, Alg has processed $\frac{P^2(1+\delta)}{\delta} \cdot P(1+\delta) - R_2$ units of work for big jobs. Alg processes δ units of preemption overhead before processing at most $\frac{P(1+\delta)}{2}$ units of work. Hence, the amount of time that Alg is processing a big job is at least $(\frac{P^3(1+\delta)^2}{\delta} - R_2) + (\frac{P^3(1+\delta)^2}{\delta} - R_2)/\frac{P(1+\delta)}{2} \cdot \delta = \frac{P^3(1+\delta)^2}{\delta} + 2P^2(1+\delta) - R_2(1+\frac{2\delta}{P(1+\delta)})$. Similarly, the amount of time that Alg is processing a small job is at least $(\frac{P^3(1+\delta)}{\delta} - R_1)(1+\delta)$.

The total amount of time spent on small and big jobs is at most t_o. Hence, the sum of the above two terms is at most $t_o = \frac{P^3(1+\delta)w}{\delta} = 2\frac{P^3(1+\delta)^2}{\delta} + P^2(1+\delta)$. Rearranging the terms, we have $P^2(1+\delta) - R_2(1+\frac{2\delta}{P(1+\delta)}) - R_1(1+\delta) \leq 0$. If $R_1 \geq P/6$, then (i) is true. Otherwise, $P^2(1+\delta) \leq R_2(1+\frac{2\delta}{P(1+\delta)}) + R_1(1+\delta) \leq R_2(1+2) + P(1+\delta)/6$. Hence, $R_2 \geq \frac{5}{18}P^2(1+\delta) > \frac{1}{6}P^2(1+\delta)$ and (ii) is true. □

Lemma 2. *The total flow time of Alg is $\Omega(P)$ times the optimal total flow time.*

Proof. Recall that the optimal schedule Opt can complete all except two jobs by time t_o. At any time during $[0, t_o]$, Opt has at most P unfinished jobs. During $[t_o, t_o + \frac{P^4(1+\delta)^2}{\delta}]$, Opt has at most 3 unfinished jobs. Opt has at most 2 unfinished jobs at time $t_o + \frac{P^4(1+\delta)^2}{\delta}$, which can be completed in $2(\delta + P(1+\delta))$ units of time. Hence, the total flow time of Opt is at most $t_o P + 3\frac{P^4(1+\delta)^2}{\delta} + 2 \cdot 2(\delta + P(1+\delta)) \leq 3\frac{P^4(1+\delta)^2}{\delta} + 3\frac{P^4(1+\delta)^2}{\delta} + 8\frac{P^4(1+\delta)^2}{\delta} \leq 14\frac{P^4(1+\delta)^2}{\delta}$.

By Lemma 1, at time t_o, Alg has at least $P/6$ units of work from small jobs or at least $\frac{P^2(1+\delta)}{6}$ units of work from big jobs. It is easy to see that Alg has at least $P/6$ jobs remaining during $[t_o, t_o + \frac{P^4(1+\delta)^2}{\delta}]$. The total flow time of Alg is at least $P/6 \cdot \frac{P^4(1+\delta)^2}{\delta} = \frac{P^5(1+\delta)^2}{6\delta}$, which is at least $P/84$ times that of Opt. □

3 A $(1+\epsilon)$-Speed $(1+\frac{1}{\epsilon})$-Competitive Algorithm

This section gives a $(1+\epsilon)$-speed $(1+\frac{1}{\epsilon})$-competitive algorithm for the single processor setting. We call the algorithm QSRPT. Intuitively, to minimize total flow time, it is natural to follow SRPT and give priority to the job with the minimum remaining work. But since there is preemption overhead, we should process a job for at least a *quantum* amount of work before preempting it. QSRPT divides the work of a job j into quantums each of fixed size $\ell > 0$, except the final quantum which has size less than or equal to ℓ. We assume that when QSRPT finishes the current quantum of a certain job, an interrupt will occur immediately to trigger selecting the next job.

> **Algorithm QSRPT(ℓ).** Let $\ell \geq 0$ be a parameter. At any time t, if the processor is idle or if a *selection interrupt* occurs, QSRPT(ℓ) selects a job for processing if there is some job unfinished. It selects the job j with the minimum remaining work, and it processes the next ℓ units of work of j (or all remaining work of j if the remaining work of j is at most ℓ). We call this work a *quantum*. Note that if j is not being processed just before t, QSRPT(ℓ) first processes h units of preemption overhead for j. QSRPT(ℓ) sets the next selection interrupt to the time at which it finishes processing this quantum.

If a quantum is processed immediately after the preemption overhead, we call it an *inefficient* quantum. Otherwise, we call it an *efficient* quantum. Note that an efficient quantum is processed immediately after another quantum of the same job. The final quantum of each job may be efficient or inefficient.

When QSRPT selects a job j at time t, j is the job with the minimum remaining work at time t. However, new jobs with smaller size may arrive while

QSRPT is processing j. Notice that QSRPT will process j for only a bounded amount of time before the next selection occurs. Precisely, QSRPT re-selects in at most $\frac{h+\ell}{s}$ units of time, where h is the size of the preemption overhead and s is the speed of the processor of QSRPT.

Our main result is that when QSRPT(ℓ) is given a $(1+\epsilon)$-speed processor, we can set ℓ to $\frac{h}{\epsilon}$. Then, QSRPT($\frac{h}{\epsilon}$) is $(1+\epsilon)$-speed $(1+\frac{1}{\epsilon})$-competitive. The analysis is divided into the following two subsections. To ease the discussion, for the rest of this section, we assume QSRPT is given a $(1+\epsilon)$-speed processor and $\ell = \frac{h}{\epsilon}$. We also omit the parameter ℓ from the notation.

3.1 Reduction to Setting without Preemption Overhead

Consider any input instance I. In this section, we transform both the schedule of QSRPT and the optimal schedule OPT into other schedules where there is no preemption overhead. We show that the performance of QSRPT can be implied based on the performance of the new schedules.

We define another input instance I^* based on I as follows. Jobs in I^* have no preemption overhead. Whenever a job j is released at time t in I, we release a corresponding job j^* in I^* and the size of j^* is $p(j^*) = p(j) + h$.

Let OPT* be a schedule for I^* which always processes the job in I^* with the minimum remaining size. Note that OPT* minimizes the total flow time for input I^*. We observe that OPT* is a lower bound for OPT as follows.

Lemma 3. *The total flow time incurred by OPT on I is at least the total flow time incurred by OPT* on I^*.*

Proof. For any job j in I, the total amount of time that OPT spent on processing j is at least $p(j) + h$. Let S be a schedule for I^* which processes j^* whenever OPT processes the corresponding job j in I. The flow time of S on I^* is at most the flow time of OPT on I. OPT* is running SRPT on I^*, which minimizes the total flow time. The total flow time of OPT* is at most that of S. □

To analyze QSRPT, the main difficulty is that QSRPT may sometimes work on preemption overhead which can be avoided by OPT, e.g., by avoiding preemption of the jobs. Hence, there may be times during which QSRPT is wasting its processing power. Intuitively, since QSRPT has a $(1+\epsilon)$-speed processor, we can charge these wasted processing power to the period that QSRPT is processing useful work. Formally speaking, we define a schedule QSRPT* on I^* which selects jobs identically as QSRPT, but processes the job at an adjusted speed depending on what kind of work QSRPT is processing. Details are as follows.

Definition 1 (Definition of schedule QSRPT*). *At any time t, QSRPT* processes a job j^* if and only if QSRPT is processing the corresponding job j at time t. The speed at which QSRPT* processes j^* varies depending on which part of j QSRPT is processing.*

1. *If QSRPT is processing the preemption overhead of j, then QSRPT* processes j^* at speed 1.*
2. *If QSRPT is processing the work of j but is not processing the final quantum, there are two cases. If this quantum is inefficient, QSRPT* processes j^* at speed 1; otherwise, QSRPT* proceeses j^* at speed $(1+\epsilon)$.*
3. *If QSRPT is processing the work of j and is processing the final quantum, there are two cases. Let f be the size of this final quantum. If this quantum is inefficient, QSRPT* processes j^* at speed $(1+\epsilon)+\frac{\epsilon h}{f}$; otherwise, i.e., this quantum is efficient, QSRPT* proceeses j^* at speed $(1+\epsilon)+\frac{(1+\epsilon)h}{f}$.*

Lemma 4. *For any job j^*, the work done on j^* by QSRPT* is $p(j^*) = p(j)+h$.*

Proof. Note that the length of time for QSRPT to process a preemption overhead is $\frac{h}{1+\epsilon}$. The length of time for QSRPT to process $\ell = \frac{h}{\epsilon}$ units of work is $\frac{h}{\epsilon(1+\epsilon)}$. The length of time for QSRPT to process the final quantum (of size f) is $\frac{f}{1+\epsilon}$.

For a maximal period during which QSRPT is processing the preemption overhead, the work done on j^* is $\frac{h}{1+\epsilon}$. For a period that QSRPT processes a non-final inefficient quantum, the work done on j^* is $\frac{h}{\epsilon(1+\epsilon)}$. For a period that QSRPT processes a non-final efficient quantum, the work done on j^* is $\frac{h}{\epsilon}$.

Assume that j has x inefficient quantums and y efficient quantums excluding the final quantum. Hence, the size of j is $p(j) = (x+y)\frac{h}{\epsilon} + f$. The size of j^* is $p(j^*) = p(j) + h$. We consider two cases depending on the final quantum of j. If the final quantum is efficient, then QSRPT processes x preemption overhead of j. The total work done on j^* equals

$$x \cdot \left(\frac{h}{\epsilon(1+\epsilon)} + \frac{h}{1+\epsilon}\right) + y \cdot \frac{h}{\epsilon} + \left((1+\epsilon) + \frac{(1+\epsilon)h}{f}\right) \cdot \frac{f}{1+\epsilon} = (x+y)\frac{h}{\epsilon} + f + h.$$

Similarly, if the final quantum is inefficient, then QSRPT processes $(x+1)$ preemption overhead of j. The total work done on j^* equals

$$x \cdot \frac{h}{\epsilon(1+\epsilon)} + (x+1) \cdot \frac{h}{1+\epsilon} + y \cdot \frac{h}{\epsilon} + \left((1+\epsilon) + \frac{\epsilon h}{f}\right) \cdot \frac{f}{1+\epsilon} = (x+y)\frac{h}{\epsilon} + f + h. \qquad \square$$

Lemma 5. *The total flow time incurred by QSRPT on I equals the total flow time incurred by QSRPT* on I^*.*

Proof. By Lemma 4, the jobs j and j^* complete at the same time. $\square$

In Theorem 3 in the next subsection, we show that the flow time of QSRPT* is at most $(1+\frac{1}{\epsilon})$ times that of OPT*. Hence, together with Lemma 3 and 5, we conclude with the following theorem.

Theorem 2. *When $\ell = \frac{h}{\epsilon}$, QSRPT($\ell$) is $(1+\epsilon)$-speed $(1+\frac{1}{\epsilon})$-competitive.*

Before analyzing QSRPT*, we first notice three properties of it. we say that QSRPT* selects job j^* at time t if QSRPT selects the corresponding job j at time t. We say a selection interrupt occurs in QSRPT* at time t if a selection interrupt occurs in QSRPT at time t. Property 1 is obvious.

Property 1. **(i)** QSRPT* never idles if there is some job unfinished.
(ii) QSRPT* always processes the selected job at speed at least 1.
(iii) After QSRPT* selects a job j^*, it processes j^* for at most $(h+\frac{h}{\epsilon})/(1+\epsilon) = \frac{h}{\epsilon}$ units of time before the next selection interrupt occurs.

Property 2. Assume QSRPT* selects a job j^* at time t, then j^* is the unfinished job with the minimum remaining work at time t.

Proof. Consider any unfinished job j_o^* in QSRPT* and the corresponding job j_o in QSRPT. Since a selection interrupt occur at time t and j_o is unfinished, QSRPT has processed c quantums of j_o, where c is an integer. Furthermore, these c quantums are non-final. Let $p(j_o, t)$ and $p(j_o^*, t)$ be the remaining size of j_o and j_o^* at time t, respectively. Then $p(j_o, t) = p(j_o) - c\frac{h}{\epsilon}$. Among those c quantums, suppose there are x inefficient quantums and $(c - x)$ efficient quantums. Then QSRPT processes x preemption overheads for j_o by time t. The amount of work done on j_o^* by QSRPT* equals $x\frac{h}{1+\epsilon} + x\frac{h}{\epsilon(1+\epsilon)} + (c - x)\frac{h}{\epsilon} = c\frac{h}{\epsilon}$. Hence, $p(j_o^*, t) = p(j_o^*) - c\frac{h}{\epsilon} = h + p(j_o, t)$.

Hence, for any unfinished jobs in QSRPT*, its remaining size at time t is h plus the remaining size of the corresponding job in QSRPT. At time t, QSRPT* selects the job j^* because QSRPT selects the corresponding job j. j is the unfinished job with the minimum remaining size in QSRPT, hence j^* is the unfinished job with the minimum remaining size in QSRPT*. □

Property 3. At any time, each unfinished job that is not being processed by QSRPT* has remaining size at least h.

Proof. At any time t, consider any unfinished job j^* that is not being processed by QSRPT*. Let $p(j^*, t)$ and $p(j, t)$, respectively, be the remaining size of j^* and the corresponding job j in QSRPT at time t. Note that QSRPT has processed c non-final quantums of j, where c is an integer. By the same calculation as in the proof of Property 2, $p(j^*, t) = p(j, t) + h \geq h$. □

3.2 Analysis of QSRPT* and OPT*

Recall that I^* is an input where jobs have no preemption overhead. The schedule OPT* always processes the job with the minimum remaining size using a 1-speed processor. QSRPT* is a schedule satisfying Property 1, 2 and 3. This section shows that the flow time of QSRPT* is at most $(1 + \frac{1}{\epsilon})$ times that of OPT*.

We first define some notations. Consider any time t. Let N_t be the total number of jobs released by t. Consider the schedule of OPT*. We list the N_t jobs in increasing order of their remaining sizes in OPT* (ties broken by job IDs). Note that the list may start with jobs with zero remaining size, which are jobs already completed by OPT* by time t. We call this list the *profile* of OPT* at time t, and denote it as OPT_t^*. Note that OPT* is working on the first job in

OPT_t^* with non-zero remaining size. For any $k = 1, \ldots, N_t$, we denote $OPT_t^*[k]$ as the remaining size of the k-th job in OPT_t^*. Similarly, we list the N_t jobs in increasing order of their remaining sizes in QSRPT* (ties broken by job IDs). Then, we define the profile $QSRPT_t^*$ and $QSPRT_t^*[k]$ similarly.

At any time t, let $prefix_q(k,t) = \sum_{i=1}^{k} QSRPT_t^*[i]$, and let $prefix_o(k,t) = \sum_{i=1}^{k} OPT_t^*[i]$. We can prove the following relationship between the two profiles (Lemma 6), which will imply our desired result (Theorem 3).

Lemma 6. *At any time t, for any $k = 1, \ldots, N_t$, $prefix_q(k,t) \le prefix_o(k,t) + \frac{h}{\epsilon}$.*

Theorem 3. *The total flow time of QSRPT* on I^* is at most $(1+\frac{1}{\epsilon})$ times the total flow time of OPT* on I^*.*

Proof. At any time t, let $n_q(t)$ and $n_o(t)$ be the number of jobs with non-zero remaining size in the profiles of QSRPT* and OPT*, respectively. Let T be the union of all time intervals during which QSRPT* is processing a job. Let $T_1 \subseteq T$ be those time intervals that all unfinished jobs in QSRPT* have remaining size at least h. Let $T_2 = T \setminus T_1$. By definition, at any time $t \in T_2$, QSRPT* has at least one unfinished job with remaining size less than h. Together with Property 3, we know that at any time $t \in T_2$, QSRPT* has exactly one unfinished job with remaining size less than h and this job is being processed at t.

At any time $t \in T_1 \cup T_2$, OPT* has $n_o(t)$ unfinished jobs and $prefix_o(N_t - n_o(t), t) = 0$. Assume among the the first $(N_t - n_o(t))$ jobs in the profile of QSRPT*, there are y jobs with non-zero remaining size. By Lemma 6, the total remaining size of these y jobs is $prefix_q(N_t - n_o(t), t) \le \frac{h}{\epsilon}$. Note that $n_q(t) \le n_o(t) + y$. For any $t \in T_1$, all unfinished jobs in QSRPT* has remaining size at least h, hence $y \le \frac{1}{\epsilon}$ and $n_q(t) \le n_o(t) + \frac{1}{\epsilon}$. For any $t \in T_2$, there is exactly one unfinished job in QSRPT* with remaining size less than h, hence $y \le 1 + \frac{1}{\epsilon}$ and $n_q(t) \le n_o(t) + 1 + \frac{1}{\epsilon}$.

Let $|T|, |T_1|$ and $|T_2|$ be the total length of T, T_1 and T_2, respectively. The flow time of QSRPT* is $\int_{T_1} n_q(t)dt + \int_{T_2} n_q(t)dt \le \int_{T_1} (n_o(t) + \frac{1}{\epsilon})dt + \int_{T_2} (n_o(t) + 1 + \frac{1}{\epsilon})dt = \int_{T_1 \cup T_2} n_o(t)dt + \frac{1}{\epsilon}(|T_1| + |T_2| + \epsilon|T_2|) = \int_T n_o(t)dt + \frac{1}{\epsilon}(|T| + \epsilon|T_2|)$.

Consider any job j^*, let $\alpha(j^*)$ the amount of time that QSRPT* is processing the last h units of work of j^*. We want to show that $\alpha(j^*) \le \frac{h}{1+\epsilon}$. Consider the corresponding job j in QSRPT. QSRPT takes $\frac{f}{1+\epsilon}$ units of time to complete the final quantum of size f. During this period, QSRPT* process j^* at speed at least $(1+\epsilon) + \frac{\epsilon h}{f}$ and the work done by QSPRT* is at least $f + \frac{h\epsilon}{1+\epsilon}$. If $f + \frac{h\epsilon}{1+\epsilon} \ge h$, we conclude that QSPRT* processes the last h units of work with speed at least $1+\epsilon$ and $\alpha(j^*) \le \frac{h}{1+\epsilon}$. Else if $f + \frac{h\epsilon}{1+\epsilon} < h$, since QSRPT* always has speed at least 1 and the last $f + \frac{h\epsilon}{1+\epsilon}$ units of work takes at most $\frac{f}{1+\epsilon}$ units of time, $\alpha(j^*) \le (h - (f + \frac{h\epsilon}{1+\epsilon})) + \frac{f}{1+\epsilon} = \frac{h}{1+\epsilon} - \frac{\epsilon f}{1+\epsilon} \le \frac{h}{1+\epsilon}$.

For each job j^*, let $\beta(j^*)$ be the amount of time that QSRPT* is processing the first $p(j^*) - h$ units of work. Obviously, $\beta(j^*) \le p(j^*) - h$. Hence,

$|T| = \sum_{j^* \in I^*}(\alpha(j^*) + \beta(j^*)) \leq \sum_{j^* \in I^*}(p(j^*) - h + \frac{h}{1+\epsilon})$. Also, $|T_2| = \sum_{j^* \in I^*} \alpha(j^*) \leq \sum_{j^* \in I^*} \frac{h}{1+\epsilon}$. The flow time of QSPRT* is at most $\int_T n_o(t)dt + \frac{1}{\epsilon}(|T| + \epsilon|T_2|) \leq \int_T n_o(t)dt + \frac{1}{\epsilon}\sum_{j^* \in I^*} p(j^*)$. Note that $\int_T n_o(t)dt$ and $\sum_{j^* \in I^*} p(j^*)$ are both lower bounds for the flow time of OPT*, the theorem follows. □

It remains to prove Lemma 6. We show that $prefix_q$ and $prefix_o$ actually satisfy a stronger relationship (see Lemma 7). At any time t, we say that a job j^* has *rank* i in the profile of QSRPT* if it appears as the i-th entry in the profile QSRPT^*_t. Let $index(t)$ be the rank of the job that QSRPT* is processing at time t. With Lemma 7, proving Lemma 6 is straightforward.

Lemma 7. *Consider any time t that QSRPT* is processing a job. Let $last(t)$ be the latest time on or before t when a job selection is performed. For any $k = 1, \ldots, N_t$,*
(i) *if $k \geq index(t)$, then $prefix_q(k,t) - prefix_o(k,t) \leq \frac{h}{\epsilon}$; and*
(ii) *if $k < index(t)$, then $prefix_q(k,t) - prefix_o(k,t) \leq$*
$$\max\left\{t - last(t), \tfrac{h}{\epsilon} - (QSRPT^*_t[index(t)] - QSRPT^*_t[k])\right\}.$$

Proof of Lemma 6. Notice that Lemma 6 is true if QSRPT* is idle at time t, since QSRPT* is idle only when all jobs are completed by QSRPT*. When QSRPT* is not idle at time t, Lemma 7 gives bounds on $prefix_q(k,t) - prefix_o(k,t)$. We only need to check that $t - last(t) \leq \frac{h}{\epsilon}$ and $\frac{h}{\epsilon} - (QSRPT^*_t[index(t)] - QSRPT^*_t[k])\} \leq \frac{h}{\epsilon}$. The first bound is true by Property 1(iii). The second bound is true because when $k < index(t)$, $QSRPT^*_t[index(t)] \geq QSRPT^*_t[k]$. □

We prove Lemma 7 by induction on time. Consider a maximal interval $[t_1, t_2]$ during which QSRPT* is processing some job. Just after the first job j is released at time t_1, $index(t_1) = N_{t_1}$, $last(t_1) = t_1$, and all jobs in QSRPT* have remaining size zero except that $QSRPT^*_{t_1}[N_{t_1}] = p(j)$. Hence, $QSRPT^*_{t_1}[i] \leq OPT^*_{t_1}[i]$ for all i and we can easily check that the lemma is true at t_1.

We partition $[t_1, t_2]$ into intervals by following *discrete events*: job arrival, job completion, selection of a job and change of $index(t)$. We show that if the lemma is true at some time t, it remains true after a period without any discrete event. We also show that if the lemma is true before a discrete event occurs, it remains true after that discrete event. Hence, by induction on time, the lemma is true at any time during $[t_1, t_2]$. The induction involves a careful case analysis and is non-trivial, but due to space limit, we leave the details to the full paper.

4 Multiprocessor Scheduling

When there are $m > 1$ processors, we let h to denote the preemption overhead and h' to denote the migration overhead. This section gives a $(1+\epsilon)$-speed $O(\frac{1}{\epsilon})$-competitive algorithm.

Consider any input I. Let OPT be the optimal (migratory) schedule. We define another input I^* for the setting without preemption and migration overhead. In particular, whenever a job j is released in I, we release a corresponding job $j^* \in I^*$ where $r(j^*) = r(j)$ and $p(j^*) = p(j) + h$. Let OPT* be the (migratory) schedules that minimizes the flow time for I^*. It is obvious that the flow time of OPT* is at most the flow time of OPT. For the setting without preemption and migration overhead, there is a known competitive algorithm.

Lemma 8. *[2] For minimizing flow time without preemption and migration overhead, the non-migratory algorithm IMD is $(1+\epsilon)$-speed $O(\frac{1}{\epsilon})$-competitive. Furthermore, IMD assigns each job to a processor once the job arrives and on each processor IMD schedules jobs by SRPT.*

Given any input I in the setting with preemption and migration overhead, our algorithm follows the same job assignment as IMD and schedules each individual processors by QSRPT. Since QSRPT is competitive for single processor scheduling and there is no job migration, the resulting algorithm has flow time comparable to that of IMD, which in turns is competitive with OPT*.

Algorithm IMD-QSRPT(ℓ). Let $\ell \geq 0$ be a parameter. We assume IMD-QSRPT(ℓ) is given $(1+\epsilon)^2$-speed processors.

Job assignment. We simulate a copy of IMD with $(1+\epsilon)$-speed processors and input I^*. Let q_i and q_i^* denotes the processors in IMD-QSRPT and IMD, respectively, where $i = 1, \ldots, m$. At any time, if a new job j arrives, we assign j to q_i if IMD assigns the corresponding job j^* to q_i^*.

Job execution. At any time, each processor q_i processes the jobs assigned to it by QSRPT(ℓ) with a $(1+\epsilon)^2$-speed processor.

Theorem 4. *When $\ell = \frac{h}{\epsilon}$, IMD-QSRPT($\ell$) is $(1+\epsilon)^2$-speed $O(\frac{1}{\epsilon})$-competitive for minimizing flow time with preemption and migration overhead in the multi-processor setting.*

Proof. Consider any processor q_i and q_i^*. Let $I_i \subseteq I$ and $I_i^* \subseteq I^*$ be the sets of jobs assigned to q_i and q_i^*, respectively. Let $\overline{I_i}$ (resp., $\overline{I_i^*}$) be the set of jobs obtained by scaling down the preemption overhead and size of each job in I_i (resp., $\overline{I_i^*}$) by a factor of $1+\epsilon$. For any algorithm A, denote $F(A, s, I)$ as the total flow time incurred when algorithm A is given an s-speed processor and input I. Notice that $F(QSRPT(\frac{h}{\epsilon}), (1+\epsilon)^2, I_i) = F(QSRPT(\frac{\overline{h}}{\epsilon}), (1+\epsilon), \overline{I_i})$, where $\overline{h} = \frac{h}{1+\epsilon}$ is the preemption overhead for $\overline{I_i}$. The proof of Theorem 3 actually shows a stronger bound that $F(QSRPT(\frac{\overline{h}}{\epsilon}), (1+\epsilon), \overline{I_i}) \leq F(SRPT, 1, \overline{I_i^*}) + \frac{1}{\epsilon}\sum_{\overline{j^*} \in \overline{I_i^*}} p(\overline{j^*})$. Also, $F(SRPT, 1, \overline{I_i^*}) = F(SRPT, 1+\epsilon, I_i^*)$. Combining these equalities and inequality, we conclude that the flow time incurred by IMD-QSRPT on I_i is at most the flow time incurred by IMD on I_i^* plus $\frac{1}{\epsilon}\sum_{\overline{j^*} \in \overline{I_i^*}} p(\overline{j^*}) \leq \frac{1}{\epsilon}\sum_{j^* \in I_i^*} p(j^*)$.

Summing up over all processors, we have that the total flow time of IMD-QSRPT is at most the total flow time of IMD plus $\frac{1}{\epsilon}\sum_{j^* \in I^*} p(j^*)$. Since the total flow time of IMD is at most $O(\frac{1}{\epsilon})$ times that of OPT* and $\sum_{j^* \in I^*} p(j^*)$ is at most the total flow time of OPT*, the theorem follows. □

By setting $\epsilon = \frac{1}{3}\epsilon'$, IMD-QSRPT is $(1 + \frac{2}{3}\epsilon' + (\frac{1}{3}\epsilon')^2) \leq (1 + \epsilon')$-speed $O(\frac{3}{\epsilon'})$-competitive.

References

1. Allahverdi, A., Ng, C.T., Cheng, T.C.E., Kovalyov, M.Y.: A survey of scheduling problems with setup times or costs. European Journal of Operational Research 187(3), 985–1032 (2008)
2. Avrahami, N., Azar, Y.: Minimizing total flow time and total completion time with immediate dispatching. Algorithmica 47(3), 253–268 (2007)
3. Bartal, Y., Leonardi, S., Shallom, G., Sitters, R.A.: On the Value of Preemption in Scheduling. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX and RANDOM 2006. LNCS, vol. 4110, pp. 39–48. Springer, Heidelberg (2006)
4. Chen, B.: A better heuristic for preemptive parallel machine scheduling with batch setup times. SIAM J. Comput. 22(6), 1303–1318 (1993)
5. Crauwels, H.A.J., Potts, C.N., Oudheusden, D.V., Wassenhove, L.N.V.: Branch and bound algorithms for single machine scheduling with batching to minimize the number of late jobs. J. Scheduling 8(2), 161–177 (2005)
6. Divakaran, S., Saks, M.E.: Approximation algorithms for problems in scheduling with set-ups. Discrete Applied Mathematics 156(5), 719–729 (2008)
7. Divakaran, S., Saks, M.E.: An online algorithm for a problem in scheduling with set-ups and release times. Algorithmica 60(2), 301–315 (2011)
8. Fox, K., Moseley, B.: Online scheduling on identical machines using srpt. In: Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, USA, January 23-25, pp. 120–128 (2011)
9. Hariri, A., Potts, C.: Single machine scheduling with batch set-up times to minimize maximum lateness. Annals of Operations Research 70, 75–92 (1997)
10. Heydari, M., Sadjadi, S., Mohammadi, E.: Minimizing total flow time subject to preemption penalties in online scheduling. The International Journal of Advanced Manufacturing Technology 47, 227–236 (2010), doi:10.1007/s00170-009-2190-9
11. Leonardi, S., Raz, D.: Approximating total flow time on parallel machines. J. Comput. Syst. Sci. 73(6), 875–891 (2007)
12. Liu, Z., Cheng, T.C.E.: Scheduling with job release dates, delivery times and preemption penalties. Inf. Process. Lett. 82(2), 107–111 (2002)
13. Liu, Z., Cheng, T.C.E.: Minimizing total completion time subject to job release dates and preemption penalties. J. Scheduling 7(4), 313–327 (2004)
14. Monma, C.L., Potts, C.N.: Analysis of heuristics for preemptive parallel machine scheduling with batch setup times. Oper. Res. 41, 981–993 (1993)
15. Phillips, C.A., Stein, C., Torng, E., Wein, J.: Optimal time-critical scheduling via resource augmentation. Algorithmica 32(2), 163–200 (2002)
16. Schuurman, P., Woeginger, G.J.: Preemptive scheduling with job-dependent setup times. In: SODA, pp. 759–767 (1999)

Prize-Collecting Survivable Network Design in Node-Weighted Graphs*

Chandra Chekuri, Alina Ene, and Ali Vakilian

Dept. of Computer Science, University of Illinois, Urbana, IL 61801, USA
{chekuri,ene1,vakilia2}@illinois.edu

Abstract. We consider node-weighted network design problems, in particular the survivable network design problem (SNDP) and its prize-collecting version (PC-SNDP). The input consists of a node-weighted undirected graph $G = (V, E)$ and integral connectivity requirements $r(st)$ for each pair of nodes st. The goal is to find a minimum node-weighted subgraph H of G such that, for each pair st, H contains $r(st)$ edge-disjoint paths between s and t. PC-SNDP is a generalization in which the input also includes a penalty $\pi(st)$ for each pair, and the goal is to find a subgraph H to minimize the sum of the weight of H and the sum of the penalties for all pairs whose connectivity requirements are not fully satisfied by H. Let $k = \max_{st} r(st)$ be the maximum requirement. There has been no non-trivial approximation for node-weighted PC-SNDP for $k > 1$, the main reason being the lack of an LP relaxation based approach for node-weighted SNDP. In this paper we describe multiroute-flow based relaxations for the two problems and obtain approximation algorithms for PC-SNDP through them. The approximation ratios we obtain for PC-SNDP are similar to those that were previously known for SNDP via combinatorial algorithms. Specifically, we obtain an $O(k^2 \log n)$-approximation in general graphs and an $O(k^2)$-approximation in graphs that exclude a fixed minor. The approximation ratios can be improved by a factor of k but the running times of the algorithms depend polynomially on n^k.

1 Introduction

In this paper we consider the survivable network design problem (SNDP) and its prize-collecting version (PC-SNDP). In SNDP the input consists of an undirected graph $G = (V, E)$ and a connectivity requirement function specified in terms of an integer $r(st)$ for each unordered pair of nodes st. The goal is to find a minimum-weight subgraph H of G that contains $r(st)$ disjoint paths for each pair st. We use EC-SNDP and VC-SNDP to refer to the versions of SNDP depending on whether the desired paths are edge or node disjoint. In this paper we focus on EC-SNDP; for notational convenience we use SNDP when we mean EC-SNDP. A parameter of interest is the maximum requirement $k = \max_{st} r(st)$. Special cases of SNDP include well-studied problems such as the Steiner tree and Steiner forest problems (here $k = 1$). The weight of the chosen subgraph H can depend both on the edges and nodes in H. In the edge-weighted version, each edge has a weight $w(e)$ and the weight of H is the sum of the weights of the edges

* Partially supported by NSF grant CCF-1016684. A longer version containing the omitted proofs will be made available on the authors' webpages.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 98–109, 2012.
© Springer-Verlag Berlin Heidelberg 2012

in H; Jain [14] obtained a 2-approximation for this problem via the influential iterated rounding technique that he introduced. The focus of this paper is the more general *node-weighted* case where each node v has a weight $w(v)$; the weight of H is the sum of the weights of the nodes in it[1]. The node-weighted version is provably harder to approximate. In contrast to the constant factor approximation for edge-weighted SNDP, the node-weighted Steiner tree problem is already $\Omega(\log n)$-hard to approximate via a simple reduction from the Set Cover problem [17].

Klein and Ravi [17] were the first to study node-weighted network design from an approximation point of view. They showed the hardness result mentioned above and described algorithms that achieved an $O(\log n)$-approximation for the Steiner tree and Steiner forest problems. Their algorithms are based on finding a structure called *spider*. Nutov examined the approximability of node-weighted SNDP [19] and obtained an $O(k \log n)$-approximation via the augmentation framework of Williamson et al. [21] (the connectivity requirements are met in k stages with each stage increasing the connectivity of every unsatisfied pair by 1). His algorithm is based on a non-trivial structural result on spiders for covering an arbitrary 0-1 uncrossable requirement function[2]. Further, Nutov gave evidence, via a reduction from the k-densest subgraph problem, that a dependence on k is necessary in the approximation ratio when k is large. The algorithms of Klein and Ravi [17] and that of Nutov [19] are combinatorial. Mathematical programming relaxation based algorithms are powerful and flexible and it is natural to ask about their efficacy for node-weighted network design, and in particular for SNDP. Guha et al. [9] considered a natural LP relaxation for node-weighted Steiner tree and forest and showed that its integrality gap is $O(\log n)$, matching the bound obtained via the combinatorial algorithm; in fact, their proof uses a nice dual-fitting argument via spiders. In more recent work Demaine, Hajiaghayi, and Klein [8] demonstrated the advantage of the LP relaxation by describing a primal-dual algorithm that achieves an $O(1)$-approximation for node-weighted Steiner tree and forest when the underlying graph is planar.

In recent work [5] we generalized the work of Demaine et al. [8] and described an $O(k)$-approximation for node-weighted SNDP in planar graphs. A technical point of interest is that the algorithm is not based on a single LP relaxation. It uses the augmentation framework in which the connectivity requirements are incrementally satisfied in k phases; a separate LP relaxation (**Augment-LP**) for each stage (that depends on the solution for the previous stages) is used[3].

This paper is motivated by two questions. Is there a natural LP relaxation for node-weighted SNDP? Is there a non-trivial approximation for node-weighted PC-SNDP?

[1] The version where both edges and nodes have weights can be easily reduced to the node-weighted version by sub-dividing each edge e and placing a weight of $w(e)$ on the new node.

[2] A 0-1 set function $f : 2^V \rightarrow \{0, 1\}$ is said to be uncrossable if $f(A) = f(B) = 1$ implies that $f(A \cap B) = f(A \cup B) = 1$ or $f(A - B) = f(B - A) = 1$.

[3] There is some subtlety to understanding the integrality gap of **Augment-LP** since it only applies to a certain restricted class of uncrossable functions that arise from proper functions; in particular, each uncrossable function is a residual function of a node-induced subgraph of the original graph. This is in contrast to the edge-weighted case where there is a natural cut relaxation for covering an arbitrary uncrossable function whose integrality gap is at most 2. We refer the reader to Subsection 2.1 and [5] for more details.

We now give some background on prize-collecting network design problems and then discuss our results.

Prize-Collecting SNDP (PC-SNDP): In PC-SNDP the input, in addition to that for SNDP, consists of penalties $\pi(st)$ for each pair of nodes. The goal is to find a subgraph H of G to minimize the weight of H plus the sum of the penalties for pairs whose connectivity requirement is not satisfied by H; a pair st is not satisfied if the number of disjoint paths in H between s and t is strictly less than $r(st)$; this is the all-or-nothing penalty model and is the most interesting one from a technical point of view. The prize-collecting version of Steiner tree and Steiner forest have been studied extensively and have several theoretical and practical applications [15, 11, 20, 10]. A simple technique, introduced by Bienstock et al. [2], shows how one can use an LP relaxation based ρ-approximation algorithm for Steiner tree (and Steiner forest) to obtain an $O(\rho)$ approximation algorithm for the prize-collecting version. PC-SNDP for higher connectivity has been recently studied [18, 13, 12]. In [12] a technique similar to that of Bienstock et al. is used for edge-weighted SNDP (and also for Elem-SNDP and VC-SNDP). However, [12] shows that a straightforward and natural LP relaxation has a large integrality gap, and introduce a stronger LP relaxation. In this paper we are concerned with node-weighted PC-SNDP. For node-weighted Steiner tree and Steiner forest there is a natural LP relaxation with $O(\log n)$ integrality gap (and $O(1)$ gap for planar graphs), and one can use this to obtain a corresponding approximation for the prize-collecting version. However, as we already remarked, the algorithms for node-weighted SNDP for $k > 1$ have not been based on a single LP relaxation.

Our Contribution: Our first contribution is to formulate an LP relaxation for node-weighted SNDP and PC-SNDP via *multi-route flows* [16, 1]. We give two relaxations, one for arbitrary k and a different relaxation that is more suited for fixed k. The multi-route flow based relaxation easily allows us to apply the basic idea of Bienstock et al. [2] to reduce the PC-SNDP problem to the SNDP problem. Our second contribution is to analyze the integrality gap of these relaxations for node-weighted SNDP. We obtain an upper bound on the integrality gap by relating the optimum value of the relaxation to that of the **Augment-LP** relaxation [5] in each phase of the augmentation framework. For planar graphs we can use the result from [5] that showed that the integrality gap of the **Augment-LP** is $O(1)$. In this paper we show that **Augment-LP** has an integrality gap of $O(\log n)$ for general graphs. These ingredients give us the following theorem that summarizes our results.

Theorem 1. *There is an $O(k^2 \log n)$-approximation for node-weighted PC-SNDP in undirected graphs which improves to an $O(k^2)$-approximation for planar graphs. There is an algorithm with running time that is polynomial in n^k that achieves an $O(k \log n)$ approximation for general graphs and an $O(k)$ approximation for planar graphs.*

Discussion, Related Work and Extensions: We start with the question as to why it is non-trivial to find a natural LP relaxation for the node-weighted SNDP problem. Consider the problem where the requirement is only for a single pair st; that is, we wish to find a minimum weight subgraph that has k edge-disjoint paths from s to t. If the weights are on the edges then this problem can be solved easily via min-cost flow. However, if the weights are on the nodes the edge-disjoint paths from s to t may use

a node v multiple times, yet the weight of the node v counts only once. (This is the same issue that is also present in the *capacitated* SNDP (Cap-SNDP) problem [3,4].) The inability to solve the single pair problem exactly is at the heart of the difficulty of finding a relaxation for node-weighted SNDP. We write a multi-route flow based LP that we cannot solve in polynomial time because the separation oracle for the dual requires us to solve the single pair problem. However, this relaxation can be solved approximately within a factor of k. This is the reason that our approximation ratios depend on k^2, one factor of k from approximating the relaxation, and another factor of k from the augmentation framework. We write a different relaxation that can be solved in time that is polynomial in n^k. This relaxation is inspired by the formulation of the **Augment-LP** and allows us to improve the approximation when k is a fixed constant. Multi-route flows and cuts are useful concepts when considering higher connectivity. Their applications and properties are not as widely known as they could be, and we hope our work helps highlight their usefulness.

One can also consider the node-weighted versions of element-connectivity SNDP (Elem-SNDP) and vertex-connectivity SNDP (VC-SNDP). Multi-route flow based relaxations can be written in the same fashion. The same difficulty present in EC-SNDP for solving the single-pair node-weighted problem extends to the Elem-SNDP problem. Interestingly, it is easy to write a multi-route LP relaxation for VC-SNDP and solve it in polynomial time! The reason for this is that in VC-SNDP the paths are required to be node-disjoint and hence the capacitated aspect goes away. However, the only non-trivial algorithmic technique for VC-SNDP at this point is via a (randomized) reduction from VC-SNDP to Elem-SNDP [7]. We believe that our algorithms and analysis will extend from EC-SNDP to Elem-SNDP as well and hence indirectly also to VC-SNDP.

The multi-route flow based LP relaxations can be solved in polynomial time for edge-weighted problems. For the prize-collecting version the relaxation is in fact equivalent to that in the work of Hajiaghayi et al. [12]; the multi-route flow view makes the cut-based relaxation in [12] easier to understand. Previous work on prize-collecting SNDP has considered submodular penalty functions [20,12]; here the penalty for not connecting a set of pairs is a monotone submodular function of those pairs. It is easy to extend our algorithms and analysis to this more general case by simply replacing the linear penalty in the objective function of the relaxation by a Lovász-extension based convex penalty function; this is in the same fashion as in the work of Chudak and Nagano [6]. We omit the details in this version of the paper.

Finally, the advantage of having an LP relaxation based algorithm (for node-weighted SNDP) is the flexibility it affords in incorporating additional constraints and solving related problems. For instance, problems such as k-MST can be solved via relaxations for the Steiner tree. Guha et al. [9] studied an LP relaxation approach for node-weighted Steiner tree motivated by such considerations. Similar applications can now be derived for higher connectivity.

Organization: The rest of the paper is organized as follows. Section 2 discusses the multi-route flow based relaxations and relates their integrality gap to that of **Augment-LP**. In Section 3 we bound the integrality gap of **Augment-LP** by $O(\log n)$ for general graphs.

2 LP Relaxations for Node-Weighted PC-SNDP

Let s and t be two vertices of the graph and let ℓ be an integer. Consider a tuple $\overline{p} = (p_1, p_2, \cdots, p_\ell)$ such that each p_i is a path from s to t and the paths in $\overline{p}$ are edge-disjoint; we refer to such a tuple $\overline{p}$ as an ℓ-route tuple connecting s to t. In the following, we ignore the order in which the paths appear in the tuple; more precisely, two tuples consisting of the same collection of paths are considered to be the same tuple. A vertex v intersects $\overline{p}$ if there exists *some* path in $\overline{p}$ that contains v; we use $v \in \overline{p}$ to denote the fact that v intersects $\overline{p}$. Similarly, an edge e intersects $\overline{p}$ if there exists some path in $\overline{p}$ that contains e; we use $e \in \overline{p}$ to denote the fact that e intersects $\overline{p}$.

Consider an instance of the node-weighted PC-SNDP problem. For each unordered pair st of nodes, we let $\mathcal{P}_{st}^{r(st)}$ denote the collection of all $r(st)$-tuples that connect s to t, where $r(st)$ is the requirement of the pair. We can write a relaxation for the problem as follows. We have a variable $x(v)$ for each vertex v and a variable $z(st)$ for each pair st of nodes with the interpretation that $x(v) = 1$ if v is in the solution and $z(st) = 1$ if the requirement of st is *not* satisfied by the solution. We also have variables $f(\overline{p})$, where $\overline{p} \in \mathcal{P}_{st}^{r(st)}$, with the interpretation that $f(\overline{p}) = 1$ if the paths connecting s to t are the paths of $\overline{p}$.

PC-Multiroute-LP

$$\min \sum_{v \in V} w(v)x(v) + \sum_{st \in V \times V} \pi(st)z(st)$$

$$\text{s.t.} \sum_{\overline{p} \in \mathcal{P}_{st}^{r(st)}} f(\overline{p}) = 1 - z(st) \quad \forall st$$

$$\sum_{\overline{p} \in \mathcal{P}_{st}^{r(st)},\, v \in \overline{p}} f(\overline{p}) \leq x(v) \quad \forall v, \forall st$$

$$0 \leq x(v) \leq 1 \quad \forall v$$

$$0 \leq z(st) \leq 1 \quad \forall st$$

$$f(\overline{p}) \geq 0 \quad \forall \overline{p}$$

Multiroute-LP

$$\min \sum_{v \in V} w(v)x(v)$$

$$\text{s.t.} \sum_{\overline{p} \in \mathcal{P}_{st}^{r(st)}} f(\overline{p}) = 1 \quad \forall st$$

$$\sum_{\overline{p} \in \mathcal{P}_{st}^{r(st)},\, v \in \overline{p}} f(\overline{p}) \leq x(v) \quad \forall v, \forall st$$

$$0 \leq x(v) \leq 1 \quad \forall v$$

$$f(\overline{p}) \geq 0 \quad \forall \overline{p}$$

Proposition 1. ***PC-Multiroute-LP*** *is a valid relaxation for the node-weighted PC-SNDP problem. Moreover if there is a single pair st with non-zero requirement then the relaxation is exact.*

We summarize at a high-level our theorems about **PC-Multiroute-LP** and **Multiroute-LP** below.

- Given a feasible solution (x, f, z) to **PC-Multiroute-LP** it is easy to obtain another feasible solution (x', f', z'), via the scaling trick of Bienstock et al. [2], such that z' is integral and the cost of (x', f', z') is at most 2 times the cost of (x, f, z).
- The integrality gap of **Multiroute-LP** is $O(k \log n)$ for general graphs and $O(k)$ for graphs from a minor-closed family of graphs.

- **PC-Multiroute-LP** and **Multiroute-LP** are **NP**-hard to solve when k is part of the input. However, one can find in polynomial time a feasible solution to them with cost at most k times the optimum solution value. This is done by solving a compact relaxation. Combining the above three ingredients gives an $O(k^2 \log n)$ approximation for node-weighted PC-SNDP and the ratio improves to $O(k^2)$ for minor-closed families of graphs.
- There is a different relaxation that leads to an improvement in the approximation ratio for PC-SNDP to $O(k \log n)$ in general graphs and to $O(k)$ in minor-closed families of graphs respectively. The running time is, however, polynomial in n^k.

Remark 1. For edge-weighted problems the multi-route formulation will have a variable $x(e)$ for each edge and the total multi-route flow on each edge e for any pair will be bounded by $x(e)$. This relaxation can be solved in polynomial time since the separation oracle for the dual is the min-cost flow problem. This relaxation for PC-SNDP is equivalent (in the sense of having the same optimal value for each instance) to the cut-based relaxation from [12].

We sketch the rounding step in the first item above that reduces the PC-SNDP problem to the SNDP problem, since it demonstrates the naturalness of the multi-route LP for higher connectivity. Let (x, f, z) be a feasible fractional solution to **PC-Multiroute-LP**. Let $I = \{st \mid z(st) > 1/2\}$. Consider the SNDP instance that we get from the prize-collecting instance by setting the requirements of all the pairs in I to zero. Let J be the set of all pairs not in I. Let x' and f' be the following vectors. For each vertex $v \in V$, we set $x'(v) = \min\{1, 2x(v)\}$. For each pair $st \in J$ and each $\overline{p} \in \mathcal{P}_{st}^{r(st)}$, we set $f'(\overline{p}) = f(\overline{p})/(1 - z(st))$. (Note that, for each $st \in J$, $z(st) \leq 1/2$.) It is straightforward to show that (x', f') is a feasible solution to **Multiroute-LP** for the pairs in J. Further, the penalty incurred for pairs in I is at most twice the penalty that the fractional solution (x, f, z) already paid for them. The factor of 2 loss here can be improved slightly via an idea of Goemans as was done in prior work, but we omit the improvement in this version.

In Subsection 2.1 we show an upper bound on the integrality gap of **Multiroute-LP** via the augmentation framework and **Augment-LP** from [5].

A Different Relaxation: Consider a solution H that satisfies the requirement of the pair st. If we remove less than $r(st)$ of the edges of H then there will be at least one path from s to t in the resulting graph. With this observation in mind, we can write an LP relaxation as follows. As before, we have a variable $x(v)$ for each vertex v and a variable $z(st)$ for each pair st. We introduce the following constraints for each pair st and each set $F \subseteq E$ such that $|F| < r(st)$. Consider the network $G_F = (V, E - F)$ with node capacities given by the values $x(v)$. We impose the valid constraint that the network G_F supports at least $1 - z(st)$ units of flow from s to t subject to the node capacity constraints given by x. The resulting LP has $O(|E|^k)$ constraints and can be solved in time that is polynomial in n^k. When k is a fixed constant, this relaxation leads to an improvement in the approximation ratio.

We refer to this LP as **PC-Cut-LP** and to its non-prize-collecting counterpart as **Cut-LP**. We note that **Multiroute-LP** is strictly stronger than **Cut-LP**; on instances of the problem in which there is a single requirement pair with requirement k,

Multiroute-LP is exact whereas **Cut-LP** has an $\Omega(k)$ integrality gap. Nevertheless, we can show that the integrality gap of **Cut-LP** is $O(k \log n)$ for general graphs and $O(k)$ for graphs from a minor-closed family. The approach for upper bounding the integrality gap of **PC-Cut-LP** and **Cut-LP** is very similar to the approach described in Subsection 2.1 for upper bounding the integrality gap of **PC-Multiroute-LP** and **Multiroute-LP**.

2.1 Integrality Gap of Multiroute-LP via Augment-LP

In this section, we show that the integrality gap of **Multiroute-LP** is $O(k \log n)$ for general graphs and $O(k)$ for minor-closed families of graphs.

Theorem 2. *Let* OPT *be the value of the optimal fractional solution to* ***Multiroute-LP****. There is a polynomial time algorithm that constructs a subgraph H of G such that H is a feasible solution for the node-weighted SNDP instance and the weight of H is $O(k \log n) \cdot$ OPT.*

Theorem 3. *Let* OPT *be the value of the optimal fractional solution to* ***Multiroute-LP****. If the input graph G belongs to a minor-closed family $\mathcal{G}$, there is a polynomial time algorithm that constructs a subgraph H of G such that H is a feasible solution for the node-weighted SNDP instance and the weight of H is $O(k) \cdot$ OPT, where the constant depends only on the family $\mathcal{G}$.*

In order to prove Theorem 2 and Theorem 3, we use the augmentation framework that was introduced by Williamson et al. [21] for the edge-weighted SNDP problem. Note that the theorems only upper bound the integrality gap of the relaxations; the algorithms for SNDP are not based on solving them. The relaxations need to be solved for PC-SNDP to identify the pairs to connect and reduce to SNDP.

We start by introducing some notation. A set S *separates* a pair st iff S contains exactly one of s, t. Let $r : 2^V \to \mathbb{Z}_+$ be the function such that $r(S)$ is the maximum requirement of a pair separated by S. Let $r_\ell : 2^V \to \mathbb{Z}_+$ be the function such that $r_\ell(S) = \min\{r(S), \ell\}$ for all sets $S \subseteq V$. Let $\delta_H(S)$ be the set of all edges of H with an endpoint in S and the other in $V - S$ (note that H may not contain all the vertices of S). A graph H covers r iff $|\delta_H(S)| \geq r(S)$ for all sets S. By Menger's theorem, a graph H is a feasible solution to the SNDP instance iff H covers r.

The algorithm selects a cover H of r in k phases. The algorithm maintains the invariant that the first ℓ phases have selected a graph H_ℓ that covers r_ℓ. During phase ℓ, the algorithm adds a new set of nodes to $H_{\ell-1}$ in order to get a graph H_ℓ that covers r_ℓ. More precisely, in phase ℓ, we solve the following augmentation problem. It is convenient to assume that all the nodes in $H_{\ell-1}$ have weight zero; since we have already paid for the nodes, we can set their weight to zero at the beginning of phase ℓ. Let $h_\ell : 2^V \to \{0, 1\}$ be the function such that $h_\ell(S) = 1$ iff $|\delta_{H_{\ell-1}}(S)| = \ell - 1$ and $r(S) \geq \ell$. Let $G'_\ell = (V, E - E(H_{\ell-1}))$. The goal is to select a minimum weight subgraph K_ℓ of G'_ℓ that covers h_ℓ; once we have K_ℓ, we let H_ℓ be the subgraph of G induced by $V(H_{\ell-1}) \cup V(K_\ell)$.

In the following, we show that, in each phase ℓ, we can select a subgraph K_ℓ that covers h_ℓ such that the node weight of K_ℓ is at most $O(\log n) \cdot \mathrm{OPT}$ for general graphs and $O(1) \cdot \mathrm{OPT}$ for minor-closed families of graphs, where OPT is the value of the optimal solution to **Multiroute-LP**. It will then follow that the algorithm described above constructs a subgraph H such that H covers r and the weight of H is $O(k \log n) \cdot \mathrm{OPT}$ for general graphs and $O(k) \cdot \mathrm{OPT}$ for minor-closed families of graphs.

Consider a phase ℓ. Recall that the goal is to cover h_ℓ using a subgraph of G'_ℓ. Let $\Gamma_{G'_\ell}(S)$ be the vertex neighborhood of S; that is, the set of vertices $v \in V - S$ such that there is an edge $uv \in E(G'_\ell)$, where $u \in S$. We have the following relaxation for the augmentation problem of phase ℓ.

Augment-LP(G'_ℓ, h_ℓ)

$$\min \sum_{v \in V} w(v)x(v)$$

$$\text{s.t.} \sum_{v \in \Gamma_{G'_\ell}(S)} x(v) \geq h_\ell(S) \quad \forall S \subseteq V$$

$$x(v) \geq 0 \quad \forall v \in V$$

As shown in Lemma 1, for each phase of the algorithm, the optimal value of **Augment-LP** is at most the optimal value of **Multiroute-LP**.

Lemma 1. *Let (x, f) be a feasible solution to **Multiroute-LP**. For any phase ℓ, x is a feasible solution to **Augment-LP**(G'_ℓ, h_ℓ).*

Corollary 1. *Let ρ be such that, for each phase ℓ, the integrality gap of **Augment-LP**(G'_ℓ, h_ℓ) is at most ρ. Then the integrality gap of **Multiroute-LP** is at most $k\rho$.*

Therefore it suffices to upper bound the integrality gap of **Augment-LP**. We prove Theorem 4 in Section 3. Theorem 5 was shown in [5].

Theorem 4. *For each ℓ, the integrality gap of **Augment-LP**(G'_ℓ, h_ℓ) is $O(\log n)$. Moreover, there is a polynomial time algorithm that selects a subgraph K_ℓ of G'_ℓ such that K_ℓ covers h_ℓ and the weight of K_ℓ is at most $O(\log n)$ times the weight of the optimal fractional solution to **Augment-LP**(G'_ℓ, h_ℓ).*

Theorem 5 ([5]). *Suppose that G belongs to a minor-closed family $\mathcal{G}$. For each ℓ, the integrality gap of **Augment-LP**(G'_ℓ, h_ℓ) is a constant that depends only on the family $\mathcal{G}$. Moreover, there is a polynomial time algorithm that selects a subgraph K_ℓ of G'_ℓ such that K_ℓ covers h_ℓ and the weight of K_ℓ is at most $O(1)$ times the weight of the optimal fractional solution to **Augment-LP**(G'_ℓ, h_ℓ).*

Remark 2. The integrality gap of **Augment-LP** is unbounded when the function h_ℓ is an arbitrary uncrossable function. However, the functions h_ℓ that arise from instances of the node-weighted SNDP problem via the augmentation framework have additional properties that are exploited by Theorem 2 and Theorem 3. We refer the reader to [5] for more details.

Theorem 2 and Theorem 3 follow from Corollary 1 and Theorem 4 and Theorem 5.

3 Integrality Gap of Augment-LP

In this section, we prove Theorem 4 that upper bounds the integrality gap of **Augment-LP** in general graphs. We refer the reader to Subsection 2.1 for the relevant definitions and notation.

In order to simplify notation, we let $G' = G'_\ell$ and $h = h_\ell$; our goal is to select a minimum-weight subgraph K of G' that covers h. As we have already seen in Subsection 2.1, we have the following LP relaxation for this problem.

Augment-LP(G', h)

$$\min \sum_{v \in V} w(v)x(v)$$

$$\text{s.t.} \sum_{v \in \Gamma_{G'}(S)} x(v) \geq h(S) \quad \forall S \subseteq V$$

$$x(v) \geq 0 \quad \forall v \in V$$

Dual of **Augment-LP**(G', h)

$$\max \sum_{S \subseteq V} y(S)h(S)$$

$$\text{s.t.} \sum_{S: v \in \Gamma_{G'}(S)} y(S) \leq w(v) \quad \forall v \in V$$

$$y(S) \geq 0 \quad \forall S \subseteq V$$

Our proof uses the concept of a (generalized) *spider* that was introduced by Nutov [19] which we will define shortly. While Nutov uses a combinatorial algorithm to find a spider we find one via a primal-dual algorithm and relate its density to that of the LP relaxation. We start with some notation and some definitions that are based on [19,21].

Preliminaries: Recall that we are working with a 0-1 uncrossable function $h : 2^V \to \{0, 1\}$. We can also view h as a family consisting of all sets S such that $h(S) = 1$. Following Nutov, we let $\mathcal{F} = \{S \mid h(S) = 1\}$ be the family corresponding to h. We refer to each set in $\mathcal{F}$ as a **violated set** and we refer to the inclusion-wise minimal sets of $\mathcal{F}$ as **min-cores**. Let $\mathcal{C}$ be the set of all min-cores of $\mathcal{F}$. The sets in $\mathcal{C}$ are disjoint and we can compute the collection $\mathcal{C}$ in polynomial time for the function h that arises in SNDP [21]. Additionally, if S is a violated set and C is a min-core, either C is contained in S or C and S are disjoint.

A set $S \in \mathcal{F}$ is a **core** of $\mathcal{F}$ iff S contains exactly one min-core C; we refer to a core S that contains the min-core $C \in \mathcal{C}$ as a C-core. Let $\mathcal{A} \subseteq \mathcal{C}$ and let u be a vertex. Let $\mathcal{S}(\mathcal{A}, u) \subseteq \mathcal{F}$ be the family consisting of all sets $S \in \mathcal{F}$ such that S is an A-core for some $A \in \mathcal{A}$ and $u \notin S$. We refer to the family $\mathcal{S}(\mathcal{A}, u)$ as a **spider family**. We refer to the min-cores in $\mathcal{A}$ as the **feet** of $\mathcal{S}(\mathcal{A}, u)$ and we refer to u as the **center** of $\mathcal{S}(\mathcal{A}, u)$. A set $F \subseteq E(G')$ of edges covers a family $\mathcal{F}'$ of sets iff, for each set $S \in \mathcal{F}'$, there is at least one edge of F leaving S; more precisely, we have $|\delta_F(S)| \geq 1$ for each set $S \in \mathcal{F}'$. If $\mathcal{F}'$ is a spider family, we refer to F as a **spider cover**. Nutov [19] introduced the notions of spider families and covers as a generalization to the concept of spiders that play an important role in the algorithm of Klein and Ravi [17] for the node-weighted Steiner tree problem; we refer the reader to [19] for more details. We remark that there are subtleties when thinking about spiders for uncrossable functions since a spider cover F can be disconnected.

The Algorithm for Covering $\mathcal{F}$: Nutov extended the algorithm of Klein and Ravi to the problem of covering an uncrossable family $\mathcal{F}$ as follows. We find a spider family

$\mathcal{S}(\mathcal{A}, u)$ and a cover F of $\mathcal{S}(\mathcal{A}, u)$. Let $\mathcal{F}' = \{S \mid S \in \mathcal{F}, \delta_F(S) = \emptyset\}$ be the subfamily of $\mathcal{F}$ that is not covered by F; the residual family $\mathcal{F}'$ is uncrossable as well. Let $G'' = (V, E(G') - F)$. We recursively construct a cover $F' \subseteq E(G'')$ for $\mathcal{F}'$ and we return $F \cup F'$ as our cover of $\mathcal{F}$.

Nutov gave a polynomial time algorithm to find a spider cover whose weight (in terms of nodes) is "comparable" to the weight of the optimal *integral* solution; here the comparison is in the sense of density which is the weight divided by the number of min-cores that are removed by the addition of the cover. We show that we can find a spider cover whose weight is "comparable" to the weight of the optimal *fractional* solution for **Augment-LP**(G', h). More precisely, we show the following theorem.

Theorem 6. *There is a spider family $\mathcal{S}(\mathcal{A}, u)$ of $\mathcal{F}$ and a cover F of $\mathcal{S}(\mathcal{A}, u)$ with the following properties. Let $\mathcal{F}' = \{S \mid S \in \mathcal{F}, \delta_F(S) = \emptyset\}$ be the subfamily of $\mathcal{F}$ that is not covered by F, and let $\mathcal{C}'$ be the collection of all minimal sets of $\mathcal{F}'$. We have $|\mathcal{C}'| < |\mathcal{C}|$ and $w(V(F))$ (total weight of the nodes in F) is $O\big((|\mathcal{C}| - |\mathcal{C}'|)/|\mathcal{C}|\big)$ times the value of the optimal fractional solution to **Augment-LP**(G', h). Moreover, we can find the feet $\mathcal{A}$, the center u, and the cover F of $\mathcal{S}(\mathcal{A}, u)$ in polynomial time.*

Once we have Theorem 6, we can find a cover of h using a greedy algorithm. If the collection $\mathcal{C}$ of all minimal violated components is empty, we return an empty cover. Otherwise, let $\mathcal{S}(A, u)$ and F be the spider family and spider cover guaranteed by Theorem 6. Let H' and h' be as in the statement of Theorem 6, and let $G'' = (V, E - E(H'))$. We recursively find a cover F' of h' and we return $F \cup F'$.

It is straightforward to verify that the weight of the optimal fractional solution to **Augment-LP**(G'', h') is at most the weight of the optimal fractional solution to **Augment-LP**(G', h). This observation together with a standard set cover analysis gives us that the total weight of the cover constructed by the algorithm above is $O(\log |\mathcal{C}|)$ times the weight of the optimal fractional solution to **Augment-LP**(G', h).

Therefore, in order to complete the proof of Theorem 4, it suffices to prove Theorem 6. In the following, we give the algorithm for constructing the spider family $\mathcal{S}(\mathcal{A}, u)$.

Primal-Dual Algorithm for Constructing the Spider Family: Consider the dual of the **Augment-LP**(G', h) (see above). The algorithm selects a set $X \subseteq V(G')$ of nodes as follows. The algorithm also maintains a solution y that is feasible for the dual of **Augment-LP**(G', h); the solution y is implicitly initialized to zero.

We proceed in iterations. Consider iteration i and let X_{i-1} be the nodes selected in the first $i-1$ iterations; X_0 is the set of all zero-weight nodes. A set S is violated with respect to a set Z of nodes iff $h(S) = 1$ and $\delta_{G'[Z]}(S)$ is empty. Recall that $\mathcal{C}$ is the collection of all minimal violated components of h; note that $\mathcal{C}$ is also the collection of all minimal sets that are violated with respect to X_0. Let $\mathcal{C}_{i-1}$ be the collection of minimal violated sets with respect to X_{i-1}. For each component $C \in \mathcal{C}_{i-1}$, we have $C \subseteq X_{i-1}$ [5]. Since the components of $\mathcal{C}_{i-1}$ are disjoint and two components $C \in \mathcal{C}$ and $C' \in \mathcal{C}_{i-1}$ do not properly intersect, we have $|\mathcal{C}_{i-1}| \leq |\mathcal{C}|$. If $|\mathcal{C}_{i-1}|$ is strictly less than $|\mathcal{C}|$, we return the set $X = X_{i-1}$ and the dual solution y, and we terminate the algorithm. In other words we stop the algorithm when at least two of the min-cores in $\mathcal{C}$ "merge" and are part of the same minimal violated set of $\mathcal{C}_{i-1}$. Otherwise, we

increase the dual variables $\{y(C)\}_{C \in \mathcal{C}_{i-1}}$ uniformly until a dual constraint for a vertex becomes tight. (Note that it is possible that the increase was zero if there was already a tight vertex at the beginning of the iteration; any vertex that was already tight is not in X_{i-1}.) Let v be a vertex that became tight; if there are several such vertices, we pick one of them arbitrarily. We add v to X and we proceed to the next iteration (note that we have $X_i = X_{i-1} \cup \{v\}$).

Let X be the set of nodes selected by the algorithm. Let i^* denote the last iteration of the algorithm which adds a node. Let $\hat{\mathcal{C}} = \cup_{i \leq i^*} \mathcal{C}_{i-1}$ be the collection of all sets that were minimal violated sets throughout the history of the primal-dual algorithm before merging happens at the end of iteration i^*. Let u be the node that was added to X in iteration i^*. Intuitively, the addition of u merged some of the cores. We formally identify the min-cores associated with the merged cores as follows. Let $\mathcal{A} = \{C \in \mathcal{C} \mid \text{there is } D \in \mathcal{C}_{i^*-1} \text{ such that } C \subseteq D \text{ and } u \in \Gamma_{G'}(D)\}$. The family $\mathcal{S}(\mathcal{A}, u)$ is the desired spider family.

Finally, we perform the following *reverse-delete* step on the set X of nodes in order to identify a subset of nodes that cover $\mathcal{S}(\mathcal{A}, u)$. We let Y_C be the set of all nodes in $X - (X_0 \cup \{u\})$ that are adjacent to some C-core in $\hat{\mathcal{C}}$. The sets $\{Y_C\}_{C \in \mathcal{C}}$ are disjoint and their union is $X - (X_0 \cup \{u\})$. We consider each foot $A \in \mathcal{A}$ separately. An important observation is that $G'[Y_A \cup X_0 \cup \{u\}]$ covers $\mathcal{S}(\{A\}, u)$. For each foot A, we select a set $Z_A \subseteq Y_A$ such that $G'[Z_A \cup X_0 \cup \{u\}]$ covers $\mathcal{S}(\{A\}, u)$ as follows. We start with $Z_A = Y_A$. We consider the nodes of Z_A in the *reverse* of the order in which they were added to X. Let v be the current node. If the graph $G'[(Z_A \cup X_0 \cup \{u\}) - \{v\}]$ covers the spider family $\mathcal{S}(\{A\}, u)$, we remove v from Z_A. We set $Z = \cup_{A \in \mathcal{A}} Z_A$ and we output the family $\mathcal{S}(\mathcal{A}, u)$ and the cover $G'[Z \cup X_0 \cup \{u\}]$.

The spider family $\mathcal{S}(\mathcal{A}, u)$ and the cover $G'[Z \cup X_0 \cup \{u\}]$ have the properties required by Theorem 6; we defer the proof to a longer version of this paper.

References

1. Aggarwal, C.C., Orlin, J.B.: On multiroute maximum flows in networks. Networks 39(1), 43–52 (2002)
2. Bienstock, D., Goemans, M.X., Simchi-Levi, D., Williamson, D.: A note on the prize collecting traveling salesman problem. Mathematical Programming 59(1), 413–420 (1993)
3. Carr, R.D., Fleischer, L.K., Leung, V.J., Phillips, C.A.: Strengthening integrality gaps for capacitated network design and covering problems. In: Proc. of ACM-SIAM SODA, pp. 106–115 (2000)
4. Chakrabarty, D., Chekuri, C., Khanna, S., Korula, N.: Approximability of Capacitated Network Design. In: Günlük, O., Woeginger, G.J. (eds.) IPCO 2011. LNCS, vol. 6655, pp. 78–91. Springer, Heidelberg (2011)
5. Chekuri, C., Ene, A., Vakilian, A.: Node-Weighted Network Design in Planar and Minor-Closed Families of Graphs. In: Czumaj, A., Mehlhorn, K., Pitts, A., Wattenhofer, R. (eds.) ICALP 2012, Part I. LNCS, vol. 7391, pp. 206–217. Springer, Heidelberg (2012)
6. Chudak, F.A., Nagano, K.: Efficient solutions to relaxations of combinatorial problems with submodular penalties via the Lovász extension and non-smooth convex optimization. In: Proc. of ACM-SIAM SODA, pp. 79–88 (2007)
7. Chuzhoy, J., Khanna, S.: An $O(k^3 \log n)$-approximation algorithm for vertex-connectivity survivable network design. In: Proc. of IEEE FOCS, pp. 437–441 (2009)

8. Demaine, E.D., Hajiaghayi, M.T., Klein, P.: Node-Weighted Steiner Tree and Group Steiner Tree in Planar Graphs. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555, pp. 328–340. Springer, Heidelberg (2009)
9. Guha, S., Moss, A., Naor, J.S., Schieber, B.: Efficient recovery from power outage. In: Proc. of ACM STOC, pp. 574–582 (1999)
10. Gutner, S.: Elementary approximation algorithms for prize collecting Steiner tree problems. Information Processing Letters 107(1), 39–44 (2008)
11. Hajiaghayi, M.T., Jain, K.: The prize-collecting generalized Steiner tree problem via a new approach of primal-dual schema. In: Proc. of ACM-SIAM SODA, pp. 631–640 (2006)
12. Hajiaghayi, M.T., Khandekar, R., Kortsarz, G., Nutov, Z.: Prize-Collecting Steiner Network Problems. In: Eisenbrand, F., Shepherd, F.B. (eds.) IPCO 2010. LNCS, vol. 6080, pp. 71–84. Springer, Heidelberg (2010)
13. Hajiaghayi, M.T., Nasri, A.A.: Prize-Collecting Steiner Networks via Iterative Rounding. In: López-Ortiz, A. (ed.) LATIN 2010. LNCS, vol. 6034, pp. 515–526. Springer, Heidelberg (2010)
14. Jain, K.: A factor 2 approximation algorithm for the generalized Steiner network problem. Combinatorica 21(1), 39–60 (1998); Preliminary version in FOCS 1998
15. Johnson, D.S., Minkoff, M., Phillips, S.: The prize collecting Steiner tree problem: theory and practice. In: Proc. of ACM-SIAM SODA, pp. 760–769 (2000)
16. Kishimoto, W.: A method for obtaining the maximum multiroute flows in a network. Networks 27(4), 279–291 (1996)
17. Klein, P., Ravi, R.: A nearly best-possible approximation algorithm for node-weighted Steiner trees. J. Algorithms 19(1), 104–115 (1995); Preliminary version in IPCO 1993
18. Nagarajan, C., Sharma, Y., Williamson, D.P.: Approximation Algorithms for Prize-Collecting Network Design Problems with General Connectivity Requirements. In: Bampis, E., Skutella, M. (eds.) WAOA 2008. LNCS, vol. 5426, pp. 174–187. Springer, Heidelberg (2009)
19. Nutov, Z.: Approximating Steiner networks with node-weights. SIAM Journal of Computing 39(7), 3001–3022 (2010); Preliminary version in LATIN 2008
20. Sharma, Y., Swamy, C., Williamson, D.P.: Approximation algorithms for prize collecting forest problems with submodular penalty functions. In: Proc. of ACM-SIAM SODA, pp. 1275–1284 (2007)
21. Williamson, D.P., Goemans, M.X., Mihail, M., Vazirani, V.V.: A primal-dual approximation algorithm for generalized Steiner network problems. Combinatorica 15(3), 435–454 (1995); Preliminary version in STOC 1993

Approximating Minimum-Cost Connected T-Joins

Joseph Cheriyan, Zachary Friggstad, and Zhihan Gao

Department of Combinatorics and Optimization,
University of Waterloo, Waterloo, Ontario N2L3G1, Canada
{jcheriyan,zfriggstad,z9gao}@uwaterloo.ca

Abstract. We design and analyse approximation algorithms for the *minimum-cost connected T-join* problem: given an undirected graph $G = (V, E)$ with nonnegative costs on the edges, and a set of nodes $T \subseteq V$, find (if it exists) a spanning connected subgraph H of minimum cost such that T is the set of nodes of odd degree; H may have multiple copies of any edge of G. Recently, An, Kleinberg, and Shmoys (STOC 2012) improved on the long-standing $\frac{5}{3}$ approximation guarantee for the s, t path TSP (the special case where $T = \{s, t\}$) and presented an algorithm based on LP rounding that achieves an approximation guarantee of $\frac{1+\sqrt{5}}{2} \approx 1.618$. We show that the methods of An et al. extend to the minimum-cost connected T-join problem to give an approximation guarantee of $5/3 - 1/(9|T|) + O\left(|T|^{-2}\right)$ when $|T| \geq 4$; our approximation guarantee is 1.625 when $|T| = 4$, and it is ≈ 1.642 when $|T| = 6$. Finally, we focus on a prize-collecting version of the problem, and present a primal-dual algorithm that is "Lagrangian multiplier preserving" and that achieves an approximation guarantee of $3-2/(|T|-1)$ when $|T| \geq 4$.

Keywords: approximation algorithms, LP rounding, primal-dual method, prize-collecting problems, T-joins, TSP, s, t-path TSP.

1 Introduction

The Traveling Salesman Problem (TSP) and its variants, especially the s, t path TSP, are currently attracting substantial research interest. We focus on a generalization that captures the TSP and the s, t path TSP.

Let $G = (V, E)$ be an undirected graph with nonnegative costs c_e on the edges $e \in E$ and let T be a subset of V. A *T-join* is a *multiset* of edges J of G such that the set of nodes with odd degree in the graph $H = (V, J)$ is precisely T, that is, a node $v \in V$ has $\deg_J(v)$ odd if and only if $v \in T$, [8]. A (spanning) *connected T-join* is a *multiset* of edges F of G such that the graph $H = (V, F)$ is connected and T is the set of nodes with odd degree in H. Clearly, we may (and we shall) assume that G is connected and that $|T|$ is even, otherwise, no connected T-join exists; moreover, we may assume that each edge of G occurs with multiplicity zero, one, or two in H, otherwise, we may remove two copies of an edge from H while preserving the connected T-join property. In the *minimum-cost connected*

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 110–121, 2012.
© Springer-Verlag Berlin Heidelberg 2012

T-join problem, the goal is to find a connected T-join of minimum cost. Two well-known special cases are the TSP ($T = \emptyset$), and the s,t path TSP ($T = \{s, t\}$).

By a *metric graph* G we mean a complete graph on $V(G)$ such that the edge costs satisfy the triangle inequality. The *metric completion* of a graph G is given by the complete graph on $V(G)$ with the cost of any edge vw equal to the cost of a shortest v, w path of G. It can be seen that G has a connected T-join of cost γ if and only if the metric completion has a connected T-join of cost γ. Thus, we may assume that the given graph G is a metric graph.

Christofides presented an algorithm for the (metric) TSP that achieves an approximation guarantee of $\frac{3}{2}$, see [8], and this is the best result known. Hoogeveen [6] extended the algorithm and its analysis to the s, t path TSP, and proved an approximation guarantee of $\frac{5}{3}$. Recently, An, Kleinberg, and Shmoys [1] improved on this long-standing $\frac{5}{3}$ approximation guarantee and presented an algorithm that achieves an approximation guarantee of $\frac{1+\sqrt{5}}{2} \approx 1.61803$. To the best of our knowledge, there is only one previous result on approximating min-cost connected T-joins: Sebő and Vygen [7] present a very nice $\frac{3}{2}$-approximation algorithm for *unweighted* graphs (each edge has unit cost); in this context, we mention that the input graph cannot be assumed to be a metric graph.

All of our algorithms follow the plan of Christofides' algorithm: first, compute an appropriate tree, then, compute a D-join of minimum cost, where D denotes the set of nodes that have the "wrong degree" in the tree; finally, return the union of the tree and the D-join. (Here, a D-join means a multiset of edges E' such that D is the set of nodes of odd degree in (V, E'); throughout the paper, we use "T" and "T-join" as in the abstract, that is, T denotes a set of nodes specified in the input; we use a symbol different from T for a join with respect to some auxiliary set of nodes.)

We show that the methods of An et al. extend to the minimum-cost connected T-join problem. They presented a new proof for a $\frac{5}{3}$ approximation guarantee for the s, t path TSP; in Section 3, we show that their proof extends easily to the minimum-cost connected T-join problem. More interestingly, in Section 4, we generalize the main result of An et al. to obtain an approximation guarantee of $\frac{5}{3} - \sigma(|T|)$, where $\sigma(|T|) = \frac{4|T|-11}{36|T|^2-156|T|+168} = \frac{1}{9|T|} - O(|T|^{-2})$ when $|T| \geq 4$. For sets T whose size is bounded by a constant, this approximation guarantee is better than $\frac{5}{3}$ by a constant. For example, the approximation guarantee is 1.625 when $|T| = 4$ and is less than 1.6421 when $|T| = 6$. Our analysis uses some new methods over that of An et al. and we elaborate in the next subsection.

Our second batch of results pertain to the following prize-collecting version of the problem: in addition to the graph $G = (V, E)$ and the edge costs c, there is a nonnegative penalty $\pi(v)$ for each node $v \in V \setminus T$; the goal is to find $I \subseteq V \setminus T$ and a connected T-join F of the graph $G \setminus I$ such that $c(F) + \pi(I)$ is minimized. The special case of the prize-collecting TSP ($T = \emptyset$) has been extensively studied for over 20 years, starting with Balas [3], and an approximation guarantee of 1.91457 has been presented by Goemans [5]; also see Archer et al. [2]. The special case of the prize-collecting s, t path TSP ($T = \{s, t\}$) has also been studied, and An et al. [1] present an approximation guarantee of 1.9535.

We focus on the general problem (prize-collecting connected T-join) and present a primal-dual algorithm that achieves an approximation guarantee of $3 - \frac{2}{|T|-1}$ when $|T| \geq 4$. Our primal-dual algorithm may be viewed as a generalization of the known primal-dual 2-approximation for the prize-collecting s,t path TSP by Chaudhuri et al. [4], and we also match their approximation guarantee of 2 for $|T| = 2$. Our algorithm has the "Lagrangian Multiplier Preserving" property; this property is useful for the design of approximation algorithms for cardinality-constrained versions of problems. Furthermore, we show that our analysis is tight by presenting instances with $|T| \geq 4$ such that the cost of the solution found by the algorithm is exactly $3 - \frac{2}{|T|-1}$ times the cost of the constructed dual solution.

Our algorithm and analysis follow well-known methods for the prize-collecting Steiner tree problem, see [4,5]. One key difference comes from the cost analysis for the D-join, where D denotes the set of nodes that have the wrong degree in the tree computed by the algorithm. A simple analysis of the cost of this D-join results in an approximation guarantee of $4 - O(|T|^{-1})$. To improve on this approximation guarantee, our analysis has to go beyond the standard methods used for analysing the approximation guarantee of primal-dual algorithms.

Most of our notation is standard, and follows Schrijver [8]; Section 2 has a summary of most of our notation.

1.1 New Contributions on Min-Cost Connected T-Joins

This subsection discusses the main points of difference between our analysis and that of An et al.

Our algorithm and analysis follow that of An et al. at a high level. The algorithm solves an LP relaxation, and using the optimal solution x^* of the LP, it samples a random spanning tree J, and then computes a min-cost D-join, where D is the set of nodes of the wrong degree in J. The analysis hinges on constructing a fractional D-join (a solution to an LP formulation of the D-join problem) of low cost to "fix" the wrong-degree nodes in J.

We construct the fractional D-join as $y := \alpha \cdot \chi(J) + \beta \cdot x^* + z$ where $\chi(J)$ is the 0-1 indicator vector for the edges of J, z is some "correction" vector (described in Section 4.3), and α and β are carefully chosen values. By integrality of the D-join polytope, the cheapest D-join has cost at most the cost of y. By linearity of expectation, the expected cost of y is less than or equal to $\alpha + \beta$ times the cost of x^* plus the expected cost of z. It turns out that the correction vector z is needed *only* for a special type of cut, the so-called τ-narrow cuts: these are given by T-odd sets U such that $x^*(\delta(U)) < 1 + \tau$. When $|T| = 2$, as in An et al. [1], it turns out that (the node sets of) the τ-narrow cuts form a nested family $U_1 \subset U_2 \subset \cdots \subset U_i \subset \ldots$. This is no longer true for $|T| \geq 4$, and hence, the analysis of the correction vectors by An et al. does not apply when $|T| \geq 4$.

We prove that the τ-narrow cuts form a laminar family when $|T| \geq 4$. Moreover, in contrast with An et al., our analysis hinges on the "partition inequalities" that are satisfied by spanning trees and fractional spanning trees such as x^*, namely, every partition $\mathcal{P} = \{P_1, \ldots, P_k\}$ of the node set into nonempty sets

satisfies $x^*(\delta(P_1, \ldots, P_k)) \geq k - 1$. In our application, we are given a subfamily of τ-narrow cuts from the laminar family of τ-narrow cuts, and we have to obtain a partition of the nodeset V into *nonempty* sets that correspond to the given subfamily. It is not clear that this holds for τ close to 1, but, we prove that it holds for $\tau = O(|T|^{-1})$. This is one reason why our approximation guarantee approaches 5/3 (from below) as $|T|$ increases.

To complete the analysis, we have to fix α, β and τ subject to several constraints, and we have to minimize the expected cost of the fractional D-join. We chose the optimal values for the above constants, and thus our approximation guarantee (in terms of $|T|$) is optimal for our methods.

2 Preliminaries

We first establish some notation. Given a multiset of edges F, we use $c(F)$ to denote the cost of F; thus, $c(F) = \sum_e \mu_e^F c_e$; here, μ_e^F denotes the number of copies of the edge e in F.

For any set of edges F of G, we use $\chi(F)$ to denote the zero-one indicator vector of F, thus, $\chi(F) \in \{0,1\}^{|E|}$, and we use $V(F)$ to denote the set of incident nodes. For any set of edges F of G and any subset of nodes S, we use $F(S)$ to denote the set of edges of F that have both endpoints in S, and we use $\delta_F(S)$ to denote the set of edges of F that have exactly one endpoint in S. We use the same notation for a multiset of edges.

For any set of nodes S, let $\overline{S}$ denote the complement $V \setminus S$. A set of nodes S is called *T-even* if $|S \cap T|$ is even, and it is called *T-odd* if $|S \cap T|$ is odd. Also, we say that a cut $\delta_F(S)$ is *T-even* (respectively, *T-odd*) if S is *T-even* (respectively, S is *T-odd*).

We say that two subsets of nodes R and S *cross* if $R \cap S$, $R \cup S$, $R \setminus S$ and $S \setminus R$ are all non-empty, proper subsets of V. A family of subsets of V is called *laminar* if no two of the subsets in the family cross.

Let $\mathcal{P} = \{P_1, \ldots, P_k\}$ be a partition of the nodes of G into nonempty sets $P_1, \ldots, P_k$. Then $\delta(\mathcal{P})$ denotes the set of edges that have endpoints in different sets in $\mathcal{P}$.

For ease of notation, we often identify a tree with its edge-set, e.g., we may use $J \subseteq E(G)$ to denote a spanning tree. Moreover, we may use relaxed notation for singleton sets, e.g., for a node t, we may use $V - t$ instead of $V \setminus \{t\}$.

We use the the next fact throughout the paper. It relates the number of odd-degree nodes in a set $U \subseteq V$ and the parity of the cut $\delta(U)$.

Lemma 1. *Let $G = (V, E)$ be a graph, and let $T \subseteq V$ have even size. Let F be a multiset of edges of G, and let D be the set of wrong-degree nodes w.r.t. F, that is, D is the set of nodes v such that either $v \in T$ and $\deg_F(v)$ is even, or $v \in V \setminus T$ and $\deg_F(v)$ is odd. Then, for any $U \subseteq V$ we have*

(i) $|\delta_F(U)| \equiv |U \cap D \cap \overline{T}| + |U \cap \overline{D} \cap T| \pmod 2$;
(ii) moreover, if U is both T-odd and D-odd, then $|\delta_F(U)|$ is even.

2.1 An LP Relaxation

We will assume that G is a metric graph for both the 5/3-approximation and its improvement; moreover, we will assume that $T \neq \emptyset$, except where stated otherwise. The next result is essential for our LP (linear programming) relaxation; the proof follows by generalizing the notion of shortcutting an Eulerian walk.

Proposition 1. *Let $G = (V, E)$ be a metric graph, and let $T \subseteq V$ have even cardinality. Assume that $T \neq \emptyset$. Given a connected T-join F, we can efficiently find a spanning tree of G of cost $\leq c(F)$ that is also a connected T-join.*

Let F be a connected T-join and consider any T-even subset of nodes S. Observe that $|\delta_F(S)|$ is even; this follows from Lemma 1 (since the set of wrong-degree nodes D in F is empty). This fact and Proposition 1 lead to our linear programming relaxation (L.P.1) for the minimum-cost connected T-join problem.

$$\begin{array}{lll} \textbf{(L.P.1)} \quad \text{minimize} : \sum_{e \in E} c_e x_e & & \\ \text{subject to} : \quad x(E(S)) \leq |S| - 1 & & \forall S \subsetneq V, |S| \geq 2 \\ x(E(V)) = |V| - 1 & & \\ x(\delta(S)) \geq 2 & & \forall \emptyset \subsetneq S \subsetneq V, |S \cap T| \text{ even} \\ x_e \geq 0 & & \forall e \in E \end{array}$$

The preceding discussion shows that the optimal value of this linear program is a lower bound for the optimal cost for the connected T-join problem when $T \neq \emptyset$. Using the ellipsoid method, we can solve this linear program efficiently, see [8].

Finally, we recall a linear programming formulation for the minimum cost T-join problem. The extreme points of this LP are integral [8] meaning that the optimal value of this LP is equal to the minimum cost of a T-join (assuming $c \geq 0$). We call any feasible solution to the following linear program a *fractional T-join*.

$$\begin{array}{lll} \textbf{(L.P.2)} \quad \text{minimize} : \sum_{e \in E} c_e x_e & & \\ \text{subject to} : \quad x(\delta(U)) \geq 1 & & \forall U \subseteq V, |U \cap T| \text{ odd} \\ x_e \geq 0 & & \forall e \in E \end{array}$$

3 A $\frac{5}{3}$-Approximation Algorithm

Hoogeveen [6] showed that Christofides' 3/2-approximation algorithm for the TSP (the case when $T = \emptyset$) extends to give a 5/3-approximation algorithm for the s, t path TSP (the case when $T = \{s, t\}$). Later, An, Kleinberg, and Shmoys (AKS) [1] proved that the 5/3-approximation guarantee holds with respect to (the optimal value of) an LP relaxation for the s, t path TSP.

It turns out that Christofides' algorithm generalizes to a 5/3-approximation algorithm for the min-cost connected T-join problem. This was observed in [7] and the arguments in [1] can be used essentially without modification to bound the integrality gap of (L.P.1) by 5/3. The (generalized) algorithm first computes a minimum spanning tree $J \subseteq E(G)$. Then let D denote the set of "wrong

degree" nodes in J. That is, D consists of the nodes in T that have even degree in J and the nodes in $V \setminus T$ that have odd degree in J. Let $M \subseteq E(G)$ be a minimum-cost D-join. Then the multiset $F = J \cup M$ (F has two copies of each edge in $J \cap M$) forms a connected T-join.

Thus the algorithm is combinatorial and does not require solving any linear programs. But, the optimal value of the linear program (L.P.1) serves as a useful lower bound on the minimum cost of a connected T-join.

Theorem 1 (An, Kleinberg, and Shmoys [1]). *Let x^* be an optimal solution for the linear programming relaxation of the connected T-join problem, (L.P.1), and let OPT_{LP} denote the optimal value $\sum_{e \in E} c_e x^*_e$. Then the solution F computed by the algorithm has cost $\leq \frac{5}{3} OPT_{LP}$.*

4 An Improved Approximation for Small T

In this section, we improve on the approximation guarantee of 5/3 for the min-cost connected T-join problem, by extending the approximation algorithm and analysis by An et al. [1], for the s,t path TSP. We assume $|T| \geq 4$, and we prove an approximation guarantee of $\frac{5}{3} - \frac{4|T|-11}{36|T|^2-156|T|+168}$.

Theorem 2. *There is an algorithm (described in Section 4.1) that finds a connected T-join F of cost at most $\frac{5}{3} - \frac{4|T|-11}{36|T|^2-156|T|+168}$ times the optimum value of linear program (L.P.1).*

4.1 The Algorithm

Let x^* denote an optimal solution to the linear programming relaxation for the minimum-cost connected T-join problem. The first two constraints of the LP allow us to decompose x^* as a convex combination of incidence vectors of spanning trees. That is, there exist spanning trees $J_1, \ldots, J_k$ and non-negative values $\lambda_1, \ldots, \lambda_k$ summing to 1 such that $x^* = \sum_{i=1}^{k} \lambda_i \chi(J_i)$. By Caratheodory's theorem, we may assume $k \leq |E| + 1$ and it is possible to find these spanning trees in polynomial time, [8]. For each spanning tree J_i, let D_i denote the set of nodes that have the "wrong" degree in J_i, that is, D_i consists of the nodes in T that have even degree in J_i and the nodes in $V \setminus T$ that have odd degree in J_i. Let M_i be a minimum cost D_i-join and let F_i be the multiset formed by the union of M_i and J_i. Clearly, each F_i is a connected T-join. We output the cheapest of these solutions.

It is easier to analyze a related randomized algorithm. Rather than trying every tree J_i, our algorithm randomly selects a single tree J by choosing J_i with probability λ_i. Since the deterministic algorithm tries all such trees, the cost of the solution found by the deterministic algorithm is at most the expected cost of the solution found by this randomized algorithm. Let D denote the set of nodes of wrong degree in J, M denote the minimum-cost D-join, and F denote the (multiset) union of M and J. The randomized algorithm returns F.

The expected cost of F is the expected cost of J plus the expected cost of the D-join M. The expected cost of the tree J is precisely the cost of x^* since each edge e has probability precisely x^*_e of appearing in J. We will show that the expected cost of M is at most $\frac{2}{3} - \frac{4|T|-11}{36|T|^2-156|T|+168}$ times the cost of x^*.

4.2 Constructing the Fractional D-Join

We will construct the fractional D-join as $y := \alpha \cdot \chi(J) + \beta \cdot x^* + z$, where $x^* \in \mathbb{R}^{|E|}$, z is some "correction" vector in $\mathbb{R}^{|E|}$ to be described below, and α and β are values which will be specified shortly; clearly, $y \in \mathbb{R}^{|E|}$. Again, by the integrality of the T-join polyhedron, the cost of M will be at most the cost of y. By linearity of expectation, the expected cost of y will be exactly $\alpha + \beta$ times the cost of x^* plus the expected cost of z.

The following lemma shows that for certain α and β, the correction vector is not needed for many cuts. The proof is similar to a result in [1].

Lemma 2. *Suppose $\alpha+2\beta \geq 1$. Then $(\alpha\cdot\chi(J)+\beta\cdot x^*)(\delta(U)) \geq 1$ if U is either*
(i) T-even, or
(ii) T-odd and D-odd, with $x^*(\delta(U)) \geq \frac{1-2\alpha}{\beta}$.

It will be convenient to fix a particular node $\widehat{t} \in T$. Unless otherwise specified, when discussing a cut of the graph we will take the set $S \subseteq V$ representing the cut to be such that $\widehat{t} \notin S$; thus the cut will be denoted $\delta(S), S \subseteq V \setminus \{\widehat{t}\}$. As T-odd cuts of the graph that have small x^* capacity will be used frequently in our analysis, we employ the following definition: *Let $\tau \geq 0$. A T-odd subset of nodes S is called* τ-narrow *if $x^*(\delta(S)) < 1 + \tau$.*

Using this definition, Lemma 2 says that if $\alpha + 2\beta \geq 1$ with both $\alpha, \beta \geq 0$, then the vector $\alpha \cdot \chi(J) + \beta \cdot x^*$ satisfies all constraints defining the D-join polyhedron except, perhaps, the constraints corresponding to T-odd, τ-narrow cuts for $\tau \geq \frac{1-2\alpha}{\beta} - 1$.

An et al. in [1], proved that if R and S are distinct T-odd, τ-narrow cuts then either $S \subset R$ or $R \subset S$. A generalization of this result to connected T-joins is the following.

Lemma 3. *If $\tau \leq 1$ and R and S are distinct T-odd, τ-narrow cuts, then R and S do not cross.*

Another way to state Lemma 3 is that the T-odd, τ-narrow cuts of the graph form a laminar family $\mathcal{L}$ of nonempty subsets of $V \setminus \{\widehat{t}\}$.

The correction vector z that we add to $\alpha\cdot\chi(J)+\beta\cdot x^*$ for the T-odd, τ-narrow cuts can be constructed from the following lemma. The main difference from the analogous result in [1] is that we require a further restriction on the size of τ. This is essentially the reason our approximation guarantee degrades to $\frac{5}{3}$ as the size of T increases.

Lemma 4. *Let $\mathcal{L} = \{U_i\}$ be the laminar family of T-odd, τ-narrow cuts. For $\tau \leq \frac{1}{|T|-2}$ there exist non-negative vectors $f^U \in \mathbb{R}^{|E|}$, one for each cut $U \in \mathcal{L}$, such that $\sum_{U\in\mathcal{L}} f^U \leq x^*$ and for each $U \in \mathcal{L}$, $f^U(\delta(U)) \geq 1$.*

The proof of this lemma is deferred to the next section. Assuming this lemma, we will now show how to complete the analysis of the algorithm. We now fix τ to be $\frac{1}{|T|-2}$. We also set $\alpha := \frac{|T|-3}{3|T|-7}$ and $\beta := \frac{|T|-2}{3|T|-7}$. For $|T| \geq 4$ and these choices of parameters, we have $\alpha + 2\beta \geq 1$ and $\tau = \frac{1-2\alpha}{\beta} - 1$.

We construct the correction vector z by including an appropriate multiple of f^U for each D-odd cut $U \in \mathcal{L}$. Formally,

$$z = \sum_{\substack{U \in \mathcal{L} \\ |U \cap D| \text{ odd}}} (1 - 2\alpha - \beta x^*(\delta(U))) \cdot f^U.$$

Since $x^*(\delta(U)) < 1 + \tau$ and $\tau = \frac{1-2\alpha}{\beta} - 1$, we have $1 - 2\alpha - \beta x^*(\delta(U)) \geq 0$ for each $U \in \mathcal{L}$ which shows $z \geq 0$. So, using Lemma 2 plus the contribution of the correction vector to the D-odd, τ-narrow cuts, we can verify that $y(\delta(U)) \geq 1$ for every D-odd set U so y is a fractional D-join.

We conclude the analysis by bounding the expected cost of y. This is given by

$$\mathbf{E}[\text{cost}(y)] = (\alpha+\beta)\ \text{cost}(x^*) + \sum_{U \in \mathcal{L}} (1 - 2\alpha - \beta x^*(U)) \cdot \mathbf{Pr}[|D \cap U| \text{ is odd}] \cdot \text{cost}(f^U).$$

As argued in [1], the probability that a T-odd cut U is also D-odd is at most $x^*(\delta(U)) - 1$. So, for each $U \in \mathcal{L}$ we can bound $(1 - 2\alpha - \beta x^*(\delta(U))) \cdot \mathbf{Pr}[|D \cap U| \text{ is odd}]$ by $(1 - 2\alpha - \beta x^*(\delta(U))) \cdot (x^*(\delta(U)) - 1)$. For $x^*(\delta(U))$ bound between 1 and $1 + \frac{1}{|T|-2}$, the maximum value of this expression is achieved at $x^*(\delta(U)) = 1 + \frac{1}{2} \cdot \frac{1}{|T|-2}$ and its value is $\gamma(|T|) := \frac{1}{12|T|^2 - 52|T| + 56}$.

So, the expected cost of y is at most $(\alpha+\beta) \cdot \text{cost}(x^*) + \gamma(|T|) \cdot \sum_{U \in \mathcal{L}} \text{cost}(f^U)$. Since $\sum_{U \in \mathcal{L}} f^U \leq x^*$, we have the final bound on the expected cost of y being $(\alpha + \beta + \gamma(|T|))\ \text{cost}(x^*)$. Adding this to the expected cost of J, we have that the expected cost of the connected T-join is at most $(1 + \alpha + \beta + \gamma(|T|)) \cdot \text{cost}(x^*)$. In terms of $|T|$, this bound on the integrality ratio is at most $\frac{5}{3} - \frac{4|T|-11}{36|T|^2 - 156|T| + 168}$. We note that for $|T| \geq 4$, this is strictly less than $\frac{5}{3}$.

4.3 The Correction Vector

We complete the analysis by proving Lemma 4. As in [1], we set up a flow network and use the max-flow/min-cut theorem to ensure a flow exists with the desired properties. However, our analysis is complicated by the fact that the sets in $\mathcal{L}$ are laminar rather than simply nested.

Let $\mathcal{L}'$ be a subfamily of $\mathcal{L}$. For $U \in \mathcal{L}'$, let $g_{\mathcal{L}'}(U)$ be the nodes in U that are not found in any smaller subset in $\mathcal{L}'$. That is,

$$g_{\mathcal{L}'}(U) = \{v \in U : v \notin W \text{ for any } W \in \mathcal{L}' \text{ with } W \subsetneq U\}.$$

The following result is the key to generalizing the argument in [1] to our setting.

Lemma 5. *Suppose that* $\tau \leq \frac{1}{|T|-2}$. *Let* $\mathcal{L}'$ *be any subfamily of* $\mathcal{L}$. *The family of subsets* $\{g_{\mathcal{L}'}(U) : U \in \mathcal{L}'\} \cup \{V \setminus \bigcup_{W \in \mathcal{L}'} W\}$ *forms a partition of* V, *and each such subset is nonempty.*

Proof. Each node v in some subset in the family $\mathcal{L}'$ is in $g_{\mathcal{L}'}(U)$ for some $U \in \mathcal{L}'$ since v is "assigned" to the smallest subset of $\mathcal{L}'$ containing v. All other nodes appear in the set $V \setminus \bigcup_{W \in \mathcal{L}'} W$. By construction, the sets are disjoint. It remains to prove that each of the sets is nonempty.

Since $\hat{t}$ is not in any subset in the family $\mathcal{L}'$, it must be that $V \setminus \bigcup_{W \in \mathcal{L}'} W \neq \emptyset$. For a set $U \in \mathcal{L}'$, let $m_{\mathcal{L}'}(U)$ be the maximal proper subsets of U in the subfamily $\mathcal{L}'$ and note that $g_{\mathcal{L}'}(U) = U \setminus \bigcup_{W \in m_{\mathcal{L}'}(U)} W$ and the sets in $m_{\mathcal{L}'}(U)$ are disjoint.

For the sake of contradiction, suppose that $g_{\mathcal{L}'}(U) = \emptyset$. Then U is the disjoint union of the sets in $m_{\mathcal{L}'}(U)$. Since every set in $\mathcal{L}'$ is T-odd, then $|m_{\mathcal{L}'}(U)|$ is also odd and we let $2k+1 = |m_{\mathcal{L}'}(U)|$. Notice that $2k+1 \leq |T|-1$ since each of the disjoint sets in $m_{\mathcal{L}'}(U)$ contain at least one node of $T - \hat{t}$.

Now we examine the quantity $X = (2k-1)x^*(\delta(U)) + \sum_{W \in m_{\mathcal{L}'}(U)} x^*(\delta(W))$. Since U and each $W \in m_{\mathcal{L}'}(U)$ is a T-odd, τ-narrow cut, then $X < (2k-1)(1+\tau)+(2k+1)(1+\tau) = 4k(1+\tau)$. Also, note that $X = \sum_{W \in m_{\mathcal{L}'}(U)} x^*(\delta(U \setminus W)) \geq 2(2k+1)$ where the inequality follows because each set $U \setminus W$ in the sum is T-even. But then $2(2k+1) \leq X < 4k(1+\tau)$, which contradicts $2k \leq |T|-2$ and $\tau \leq \frac{1}{|T|-2}$. This completes the proof of Lemma 5.

Proof (of Lemma 4). We now finish construction of the vectors $f^U, U \in \mathcal{L}$ by describing the flow network. Create a directed graph with 4 layers of nodes, where the first layer has a single source node v_s and the last layer has a single sink node v_t. We have a node v_U for each T-odd, τ-narrow cut $U \in \mathcal{L}$ in the second layer, and a node v_e for each edge $e \in E(G)$ in the third layer. For each $U \in \mathcal{L}$, there is an arc from v_s to v_U with capacity 1. For each edge e of G, there is an arc from v_e to v_t with capacity x^*_e. Finally, for each $U \in \mathcal{L}$ and each $e \in \delta(U)$ we have an arc from v_U to v_e with capacity ∞.

We claim that there is a flow from v_s to v_t that saturates each of the arcs originating from v_s; this is proved below. From such a flow, we construct the vectors f^U for $U \in \mathcal{L}$ by setting f^U_e to be the amount of flow sent on the arc from v_U to v_e (where we use $f^U_e = 0$ if $e \notin \delta(U)$). We have $f^U \geq 0$ and, by the capacities of the arcs entering v_t, $\sum_{U \in \mathcal{L}} f^U \leq x^*$. Finally, since each $U \in \mathcal{L}$ has the arc from v_s to v_U saturated by one unit of flow, we have $f^U(\delta(U)) \geq 1$. Thus, the vectors $f^U, U \in \mathcal{L}$ satisfy the requirements of Lemma 4.

We prove the existence of this flow by the max-flow/min-cut theorem. Let S be any cut with $v_s \in S, v_t \notin S$. If S contains some node v_U for $U \in \mathcal{L}$ but not v_e for some $e \in \delta(U)$, then the capacity of S is ∞. Otherwise, let $\mathcal{L}_S$ denote the subfamily of sets $U \in \mathcal{L}$ such that the node v_U representing U is in S. Then the total capacity of the arcs leaving S is

$$|\mathcal{L}| - |\mathcal{L}_S| + \sum_{\substack{e \in \delta(U) \\ \text{for some } U \in \mathcal{L}_S}} x^*_e = |\mathcal{L}| - |\mathcal{L}_S| + x^*(\mathcal{P}_S)$$

where $\mathcal{P}_S$ is the partition of V given by the sets $\{g_{\mathcal{L}_S}(U), U \in \mathcal{L}_S\} \cup \{V \setminus \bigcup_{W \in \mathcal{L}_S} W\}$. From Lemma 5, each set in $\mathcal{P}_S$ is nonempty so we have $x^*(\mathcal{P}_S) \geq |\mathcal{L}_S|$ because x^* is in the spanning tree polytope and $|\mathcal{P}_S| = |\mathcal{L}_S| + 1$. So, the capacity of this cut is at least $|\mathcal{L}|$.

Since this holds for all v_s,v_t cuts S, then the maximum flow is at least $|\mathcal{L}|$. Finally, the cut $S = \{v_s\}$ has capacity precisely $|\mathcal{L}|$ so the maximum v_s,v_t flow saturates all of the arcs exiting v_s.

5 Prize-Collecting Connected T-Joins

We start with a linear programming relaxation of the prize-collecting problem. For notational convenience, we define a large penalty for each node in T. We also designate an arbitrary node $t^\star \in T$ as the *root* node. The LP has a variable Z_X for each set $X \subseteq V - t^\star$ such that $Z_X = 1$ indicates that X is the set of isolated nodes of an optimal integral solution; moreover, we have a cut constraint for each nonempty subset S of $V - t^\star$; the requirement (r.h.s. value) of a cut constraint is 1 or 2, depending on whether the set S is T-odd or T-even.

Let $\mathcal{Q}$ denote the T-odd subsets of $V - t^\star$, and let $\mathcal{R}$ denote the non-empty T-even subsets of $V - t^\star$. For every solution to the prize-collecting connected T-join problem, observe that there exists an edge in $\delta(Q)$ for each $Q \in \mathcal{Q}$. The LP relaxation we use is the following.

$$\begin{aligned}
\textbf{(L.P.3)} \quad \text{minimize :} \quad & \sum_e c_e x_e + \sum_{X \subseteq V - t^\star} \pi(X) Z_X & & \\
\text{subject to :} \quad & x(\delta(Q)) \geq 1 & & \forall\, Q \in \mathcal{Q} \\
& x(\delta(R)) + \sum_{X : X \supseteq R, X \subseteq V - t^\star} 2 Z_X \geq 2 & & \forall\, R \in \mathcal{R} \\
& x, Z \geq 0 & &
\end{aligned}$$

The dual of (L.P.3) has a variable y_Q for each primal-constraint of the first type, and a variable y_R for each primal-constraint of the second type; thus, each T-odd set $Q \subseteq V - t^\star$ has a dual variable y_Q, and each T-even set $\emptyset \subsetneq R \subsetneq V - t^\star$ has a dual variable y_R.

$$\begin{aligned}
\textbf{(L.P.4)} \quad \text{maximize :} \quad & \sum_{Q \in \mathcal{Q}} y_Q + \sum_{R \in \mathcal{R}} 2 y_R & & \\
\text{subject to :} \quad & \sum_{S \in \mathcal{Q} \cup \mathcal{R} : e \in \delta(S)} y_S \leq c_e & & \forall\, e \in E \\
& \sum_{R \subseteq X, R \in \mathcal{R}} 2 y_R \leq \pi(X) & & \forall\, X \subseteq V - t^\star \\
& y \geq 0 & &
\end{aligned}$$

Consider the dual LP and a feasible solution y; we call an edge e *tight* if the constraint for e holds with equality, and we call a set of nodes $X \subseteq (V - t^\star)$ π-*tight* if the constraint for X holds with equality.

5.1 The Primal-Dual Algorithm

The algorithm proceeds in phases. In each phase, a partition $\mathcal{P}$ of $V(G)$ is maintained; some sets in this partition are *active* and some are *inactive*. Throughout,

the set containing the root, $t^\star$, is taken to be inactive. The initial partition consists of singletons $\{v\}$ for every $v \in V$. Each of the sets $\{v\}, v \in V - t^\star$, is designated as active. We initialize $y_S := 0$ for every subset S of V. Let F denote the set of edges choosen during the growing phase of the algorithm; we initialize $F := \emptyset$.

Each phase proceeds as follows. We simultaneously raise y_S for every active set S in the current partition at a uniform rate. The phase ends when either an edge becomes tight or an active subset of nodes S becomes π-tight. If the former occurs, we add e to F, merge the components in the current partition containing the endpoints of e, and call this new component inactive if it contains the root, otherwise, we call the new component active. If the latter occurs, that is, if an active subset $S \subseteq V$ in the partition becomes π-tight, then S becomes inactive. The algorithm terminates when there are no remaining active sets.

Standard arguments show that the dual solution at the end of the algorithm is feasible and that the set of edges F chosen throughout the algorithm is acyclic. We prune our solution F in the usual way. Namely, we iteratively discard any edge e such that there exists an inclusion-wise maximal set X that was inactive at some point of the algorithm *and* $\delta(X) = \{e\}$; moreover, after this stage of pruning, we discard all remaining edges that are not in the component of $t^\star$. Let J denote the remaining subset of edges. The subgraph that remains after discarding the isolated nodes is a tree J containing the root $t^\star$. Furthermore, since each node in T has a large penalty, then J contains all nodes in T.

Finally, let $D \subseteq V(J)$ denote the set of nodes that have the wrong degree in the tree J. Compute a minimum-cost D-join M and output $J \cup M$ as the connected T-join on $V(J)$. Let I denote the set of nodes not included in J, thus $I = V \setminus V(J)$.

5.2 Analysis of the Primal-Dual Algorithm

Our argument for bounding the cost of the tree J and the penalties of the nodes in I is similar to known arguments. The main contribution here is how we bound the cost of the D-join without simply doubling the edges of J. The following theorem summarizes the cost bounds. For convenience, we define $\rho(|T|) = 2 - \frac{1}{(|T|-1)}$.

Theorem 3. *The penalty of the nodes in I is exactly $2 \sum_{X \subseteq I} y_X$, the cost of the tree J is at most $\rho(|T|) \sum_{Q \in \mathcal{Q}} y_Q + 2 \sum_{R \in \mathcal{R}, R \not\subseteq I} y_R$, and the cost of the minimum-cost D-join M is at most $(\rho(|T|) - 1) \sum_{Q \in \mathcal{Q}} y_Q + 2 \sum_{R \in \mathcal{R}, R \not\subseteq I} y_R$.*

Proof. The bounds on the total penalty and the cost of J follow by standard arguments. The reason we get the slight improvement from 2 to $\rho(|T|)$ in the coefficient of the dual variables for sets in $\mathcal{Q}$ is that the number of T-odd active sets in any step of the algorithm is at most $|T| - 1$.

To bound the cost of the D-join, let $\widehat{J}$ be the set of edges of J whose deletion separates J into two D-even components. One can verify that $M' := J \setminus \widehat{J}$ is

a D-join. Furthermore, using parity arguments, one can show that $|\delta_{\widehat{J}}(Q)| \geq 1$ for every $Q \in \mathcal{Q}$ that was active at some point in the algorithm. This, in turn, implies that the cost of M' is at most the cost of J minus $\sum_{Q \in \mathcal{Q}} y_Q$, thus giving our bound on the cost of the minimum-cost D-join.

Our analysis is tight even up to lower-order terms when $|T| \geq 4$. This is realized by a cycle on T, that is, $G = (T, E)$ consists of an even-length cycle with at least 4 nodes. Let $t^\star \in T$ be a designated node and let the edges incident to it have cost $\frac{1}{2}$ while all other edges have cost one. The dual growth phase grows $y_{\{v\}}$ to $1/2$ for every singleton $v \in T - t^\star$. The algorithm could find a tree of cost $|T| - \frac{3}{2}$ (by picking all edges of G except one of the two edges incident to $t^\star$), and then find a D-join of cost $\frac{|T|-2}{2}$. Observe that the cost of the dual solution is $\frac{|T|-1}{2}$, whereas the connected T-join constructed by the algorithm has cost $\frac{3|T|-5}{2}$; the ratio of these two quantities is exactly $3 - \frac{2}{|T|-1}$.

6 Conclusions

A key open questions is whether the min-cost connected T-join problem can be approximated within a constant strictly smaller than $5/3$, regardless of the size of T. This question has been settled for *unweighted* graphs by Sebő and Vygen via their $\frac{3}{2}$-approximation algorithm [7], but the general question remains open.

Acknowledgements. We thank a number of colleagues for useful discussions; in particular, we thank Jochen Könemann and Chaitanya Swamy. We thank the referees for their comments. The first author is supported by NSERC grant No. OGP0138432.

References

1. An, H.-C., Kleinberg, R., Shmoys, D.B.: Improving Christofides' algorithm for the s-t path TSP. In: Proc. ACM STOC (2012); CoRR, abs/1110.4604v2 (2011)
2. Archer, A., Bateni, M., Hajiaghayi, M., Karloff, H.J.: Improved approximation algorithms for prize-collecting Steiner tree and TSP. SIAM J. Comput. 40(2), 309–332 (2011)
3. Balas, E.: The prize-collecting traveling salesman problem. Networks 19(6), 621–636 (1989)
4. Chaudhuri, K., Godfrey, B., Rao, S., Talwar, K.: Paths, trees, and minimum latency tours. In: Proc. IEEE FOCS, pp. 36–45 (2003)
5. Goemans, M.X.: Combining approximation algorithms for the prize-collecting TSP. CoRR, abs/0910.0553 (2009)
6. Hoogeveen, J.A.: Analysis of Christofides' heuristic: Some paths are more difficult than cycles. Operations Research Letters 10, 291–295 (1991)
7. Sebő, A., Vygen, J.: Shorter tours by nicer ears: 7/5-approximation for graphic TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs. CoRR, abs/1201.1870v3 (2012)
8. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Algorithms and Combinatorics, vol. 24. Springer, Berlin (2003)

iBGP and Constrained Connectivity*

Michael Dinitz[1] and Gordon Wilfong[2]

[1] Weizmann Institute of Science
michael.dinitz@weizmann.ac.il
[2] Alcatel-Lucent Bell Labs
gtw@research.bell-labs.com

Abstract. We initiate the theoretical study of the problem of minimizing the size of an iBGP (Interior Border Gateway Protocol) overlay in an Autonomous System (AS) in the Internet subject to a natural notion of correctness derived from the standard "hot-potato" routing rules. For both natural versions of the problem (where we measure the size of an overlay by either the number of edges or the maximum degree) we prove that it is NP-hard to approximate to a factor better than $\Omega(\log n)$ and provide approximation algorithms with ratio $\tilde{O}(\sqrt{n})$. This algorithm is based on a natural LP relaxation and randomized rounding technique inspired by recent progress on approximating directed spanners. The main technique we use is a reduction to a new connectivity-based network design problem that we call *Constrained Connectivity*, in which we are given a graph $G = (V, E)$ and for every pair of vertices $u, v \in V$ we are given a set $S(u, v) \subseteq V$ called the *safe set* of the pair. The goal is to find the smallest subgraph $H = (V, F)$ of G in which every pair of vertices u, v is connected by a path contained in $S(u, v)$. We show that the iBGP problem can be reduced to the special case of Constrained Connectivity where $G = K_n$. Furthermore, we believe that Constrained Connectivity is an interesting problem in its own right, so provide stronger hardness results and integrality gaps for the general case.

1 Introduction

The Internet consists of a number of interconnected subnetworks called Autonomous Systems (ASes). As described in [1], the way that routes to a given destination are chosen by routers within an AS can be viewed as follows. Routers have a ranking of routes based on economic considerations of the AS. Without loss of generality, in what follows we assume that all routes are equally ranked (it suffices to concentrate on the highest-ranked routes; lower-ranked routes can be ignored without loss of generality). Thus routers must use some tie-breaking scheme in order to choose a route from amongst the equally ranked routes. Tie-breaking is based on traffic engineering considerations and in particular, the goal is to get packets out of the AS as quickly as possible (called *hot-potato routing*).

An AS attempts to achieve hot-potato routing using the Interior Border Gateway Protocol (iBGP), the version of the interdomain routing protocol BGP [15]

* Full version can be found at http://arxiv.org/abs/1107.2299

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 122–133, 2012.
© Springer-Verlag Berlin Heidelberg 2012

used by routers within a subnetwork to announce routes that have been learned from outside the subnetwork. An iBGP configuration is defined by a *signaling graph*, which is supposed to enforce hot-potato routing. Unfortunately, while iBGP has many nice properties that make it useful in practice, constructing a good signaling graph turns out to be a computationally difficult problem. For example, it is not clear *a priori* that it is even possible to check in polynomial time that a signaling graph is correct, i.e. it is not obvious that the problem is even in NP! In this paper we study the problem of constructing small and correct signaling graphs, as well as a natural extension to a more general network design problem that we call *Constrained Connectivity*.

1.1 iBGP

We begin with some definitions. In what follows, when we speak of a route, we mean a route to some fixed destination d in the Internet. We define a *border router* as a router that initially knows of a route that it has been told about by a router outside the AS it is in. The border router that initially knows of a route is said to be the *egress router* of that route. Without loss of generality we can assume that each border router knows of exactly one route to d. Thus an initial set F of routes defines a set X_F of egress routers where there is a one-to-one relationship between routes in F and routers in X_F. The AS has an underlying physical network with edge weights, e.g. IGP (Interior Gateway Protocol) distances or OSPF (Open Shortest Path First) weights. The *distance* between two routers in the AS is then defined to be the length of the shortest path (according to the edge weights) between them. Given a set of routes, a router will choose the one whose egress router is closest according to this definition of distance. The *signaling graph* H is an overlay network whose nodes represent routers and whose edges represent the fact that the two routers at its endpoints use iBGP to inform one another of their current chosen route. The endpoints of an edge in H are called *iBGP neighbors*. A path in H is called a *signaling path*. Note that iBGP neighbors are not necessarily neighbors in the underlying graph, since H is an overlay and can include any possible edge.

The goal of iBGP is to get each router r in the AS to chose a route with a "nearby" egress router E. Then when r has packets to send to d they will be routed from r to E along a shortest path in the AS. Finally, E will forward the packets to the router outside the AS from which it learned about its route.

Intuitively, iBGP can be thought of as working as follows. We can assume that at the beginning each border router has chosen the single route that it has learned about from a router in some other AS. Each border router then tells its iBGP neighbors about its chosen route. Then in an asynchronous fashion, each router is activated: it considers the routes currently chosen by its iBGP neighbors, of these it chooses the route with the closest egress router and finally it tells its iBGP neighbors about this chosen route if the chosen route differs from its previously chosen route. This process continues until no router changes its chosen route. It should be noted that a router cannot choose any route other than one of its neighbors' currently chosen routes. Thus on activation of router

r, if it finds that its previously chosen route R is no longer a chosen route of at least one of its neighbors (implying that the neighbor from which r originally learned of R has since chosen a different route) then r is forced to choose a new route. This is true even if the egress router of R is strictly closer to r than r is to any of the egress routers of the currently chosen routes of its neighbors. It must choose one of its neighbors' currently chosen routes as its new chosen route, even if that route seems worse.

When this process ends the route chosen by router r is denoted by $R(r)$. Let $P(r)$ be the shortest path from r to $E(r)$, the egress router of $R(r)$. When a packet arrives at r, it sends it to the next router r' on $P(r)$, r' in turn sends the packet to the next router on $P(r')$ and so on. Thus if $P(r')$ is not the subpath of $P(r)$ starting at r' then the packet will not get routed as r expected. Note that because of the fact that a router can only choose a route that one of its iBGP neighbors has also chosen, when the process ends there must be a neighbor of r in the signaling graph that also chooses $R(r)$ as its route, but because the signaling graph is an overlay (and not necessarily a subgraph) this neighbor might not be r', and thus r' might have chosen a different route.

A signaling graph H has the *complete visibility property* for a set of egress routers X_F if each router r ultimately chooses as $R(r)$ the route in F whose egress router $E(r)$ is closest to r from among all routers in X_F. It is easy to see that H will achieve hot-potato routing for X_F if and only if it has the complete visibility property for X_F. So we say that a signaling graph is *correct* if it has the complete visibility property for all possible sets X_F.

Clearly if H is the complete graph then H is correct. Because of this, the default configuration of iBGP and the original standard was to maintain a complete graph, also called a full mesh [15]. However the complete graph is not practical and so network managers have adopted various configuration techniques to reduce the size of the signaling graph [2,16]. Unfortunately these methods do not guarantee correct signaling graphs [1,12]. Thus our goal is to determine correct signaling graphs with fewer edges than the complete graph. Slightly more formally, two natural questions are to minimize the number of edges in the signaling graph or to minimize the maximum number of iBGP neighbors for any router, while guaranteeing correctness. We define iBGP-Sum to be the problem of finding a correct signaling graph with the fewest edges, and similarly we define iBGP-Degree to be the problem of finding a correct signaling graph with the minimum possible maximum degree.

1.2 Constrained Connectivity

All we know *a priori* about the complexity of iBGP-Sum and iBGP-Degree is that they are in Σ_2 (the second existential level of the polynomial hierarchy), since the statement of correctness is that "there exists a small graph H such that for all possible subsets X_F each router ultimately chooses the route with the closest egress router". In particular, it is not obvious that these problems are in NP, i.e. that there is a short certificate that a signaling graph is correct. However, it turns out that these problems are actually in NP (see Section 2.1),

and the proof of this fact naturally gives rise to a more general network design problem that we call *Constrained Connectivity*. In this problem we are given an undirected graph $G = (V, E)$ and for each pair of nodes $(u, v) \in V \times V$ we are given a set $S(u, v) \subseteq V$. Each such $S(u, v)$ is called a *safe set* and it is assumed that $u, v \in S(u, v)$. We say that a subgraph $H = (V, E')$ of G is *safely connected* if for each pair of nodes (u, v) there is a path in H from u to v where each node in the path is in $S(u, v)$.

As with iBGP, we are interested in two optimization versions of this problem: Constrained Connectivity-Sum, in which we want to compute a safely connected subgraph H with the minimum number of edges, and Constrained Connectivity-Degree, in which we want to compute a safely connected subgraph H that minimizes the maximum degree over all nodes. It turns out (see Theorem 1) that the iBGP problems can be viewed as Constrained Connectivity problems with $G = K_n$ and safe sets defined in a particular geometric way.

While the motivation for studying Constrained Connectivity comes from iBGP, we believe that it is an interesting problem in its own right. It is an extremely natural and general network design problem that, somewhat surprisingly, seems to have not been considered before. While we only provide negative results for the general problem (hardness of approximation and integrality gaps), a better understanding of Constrained Connectivity might lead to a better understanding of other network design problems, both explicitly via reductions and implicitly through techniques. For example, many of the techniques used in this paper come from recent literature on directed spanners [4,7,3], and given these similarities it is not unreasonable to think that insight into Constrained Connectivity might provide insight into directed spanners.

For a more direct example, there is a natural security application of Constrained Connectivity. Suppose we have n players who wish to communicate with each other, but they do not all trust one another with messages they send to others. That is, when u wishes to send a message to v there is a subset $S(u, v)$ of players that it trusts to see the messages that it sends to v. We can represent this situation as a graph where the nodes are the players and the edges are communication channels. Of course, if for every pair of players there is a direct communication channels between the two players (i.e. the graph is K_n), then there is no problem. But suppose that the graph of communication channels is not K_n, and furthermore that there is a cost to protect communication channels from eavesdropping or other such attacks. Then a goal would be to have as small a network of communication channels as possible (to minimize the cost of security) that would still allow a route from each u to each v with the route completely contained within $S(u, v)$. Thus this problem defines a Constrained Connectivity-Sum problem.

1.3 Summary of Main Results

Due to space constraints we omit all proofs, which can be found in the full version [8]. In Section 3 we give a polynomial approximation for the iBGP problems,

by giving the same approximations for the more general problem of Constrained Connectivity on K_n.

Theorem 2. *There is an $\tilde{O}(\sqrt{n})$-approximation to the Constrained Connectivity problems on K_n, and thus also to the iBGP problems.*

To go along with these theoretical upper bounds, we design a different (but related) algorithm for Constrained Connectivity-Sum on K_n that provides a worse theoretical upper bound (a $\tilde{O}(n^{2/3})$-approximation) but is faster in both practice and theory, and show by simulation on five real AS topologies (Telstra, Sprint, NTT, TINET, and Level 3) that in practice it provides an extremely good approximation. Details of this algorithm and simulations can be found in the full version [8].

To complement these upper bounds, in Section 4 we show that the iBGP problems are hard to approximate, which implies the same hardness for the Constrained Connectivity problems on K_n:

Theorems 3 and 4. *It is NP-hard to approximate the iBGP problems to a factor better than $\Omega(\log n)$.*

We then study the more general Constrained Connectivity problems, and in Section 5 we show that the fully general constrained connectivity problems are hard to approximate:

Theorems 6 and 7. *Assuming NP $\not\subseteq$ DTIME($n^{\text{polylog}(n)}$), the Constrained Connectivity problems do not admit a $2^{\log^{1-\epsilon} n}$-approximation algorithm for any constant $\epsilon > 0$.*

This is basically the same inapproximability factor as for Label Cover, and in fact our reduction is from a minimization version of Label Cover known as Min-Rep. Moreover, we show that the natural LP relaxation has a polynomial integrality gap of $\Omega(n^{\frac{1}{3}-\epsilon})$.

Finally, we consider some other special cases of Constrained Connectivity that turn out to be easier. In particular, we say that a collection of safe sets is *symmetric* if $S(x,y) = S(y,x)$ for all $x, y \in V$ and that it is *hierarchical* if for all $x, y, z \in V$, if $z \in S(x,y)$ then $S(x,z) \subseteq S(x,y)$ and $S(z,y) \subseteq S(x,y)$. It turns out that all of our hardness results and integrality gaps also hold for symmetric instances, but adding the hierarchical property makes things easier. In the full version [8] we show that a reasonably simple greedy algorithm solves symmetric and hierarchical instances optimally in polynomial time:

Theorem. Constrained Connectivity-Sum *with symmetric and hierarchical safe sets can be solved optimally in polynomial time.*

1.4 Related Work

Issues involving eBGP, the version of BGP that routers in different ASes use to announce routes to one another, have recently received significant attention

from the theoretical computer science community, especially stability and game-theoretic issues (e.g., [11,14,9]). However, not nearly as much work has been done on problems related to iBGP. There has been some work on the problem of guaranteeing hot-potato routing in any AS with a route reflector architecture [2]. These earlier papers did not consider the issue of finding small signaling graphs that achieved the hot-potato goal. Instead they either provided sufficient conditions for correctness relating the underlying physical network with the route reflector configuration [12] or they showed that by allowing some specific extra routes to be announced (rather than just the one chosen route) they could guarantee a version of hot-potato routing [1]. The first people to consider the problem of designing small iBGP overlays subject to achieving hot-potato correctness were Vutukuru et al. [17], who used graph partitioning schemes to give such configurations. But while they proved that their algorithm gave correct configurations, they only gave simulated evidence that the configurations it produced were small. Buob et al. [5] considered the problem of designing small correct solutions and gave a mathematical programming formulation, but then simply solved the integer program using super-polynomial time algorithms.

Many of the techniques that we use are adapted from recent work on directed spanners and directed steiner forest. In particular, the LP rounding algorithm that we use is based on a framework of doing both LP-based rounding and independent tree sampling that has been used for both directed spanners [4,7,3] and (in a more complicated form) for directed steiner forest [10]. The exact rounding algorithm that we use is particularly similar to Berman et al. [3] as it uses independent randomized rounding, as opposed to the threshold-based rounding of [4,7].

2 Preliminaries

2.1 Relationship between iBGP and Constrained Connectivity

We will now show that the iBGP problems are just special cases of Constrained Connectivity-Sum and Constrained Connectivity-Degree. This will be a natural consequence of the proof that iBGP-Sum and iBGP-Degree are in NP.

To see this we will need the following definitions. We will assume without loss of generality that there are no ties, i.e. all distances are distinct. For two routers x and y, let $D(x,y) = \{w : d(x,w) > d(x,y)\}$ be the set of routers that are farther from x than y is. Let $S(x,y) = \{w : d(w,y) < d(w, D(x,y))\} \cup \{y\}$ be the set of routers that are closer to y than to any router not in the ball around x of radius $d(x,y)$ (where we slightly abuse notation and define the distance from a node x to a set J of nodes as $d(x,J) = \min\{d(x,j) : j \in J\}$). We will refer to $S(x,y)$ as "safe" routers for the pair (x,y). It turns out that these safe sets characterize correct signaling graphs.

Theorem 1. *An iBGP signaling graph H is correct if and only if for every pair $(x,y) \in V \times V$ there is a signaling path between y and x that uses only routers in $S(x,y)$.*

Note that this condition is easy to check in polynomial time, so we have shown membership in NP. Also this characterization shows that the problems iBGP-Sum and iBGP-Degree are Constrained Connectivity problems where the underlying graph is K_n and the safe sets are defined by the underlying metric space. While the proof of this is relatively simple, we believe that it is an important contribution of this paper as it allows us to characterize the behavior of a protocol (iBGP) using only the static information of the signaling graph and the network distances.

2.2 Linear Programming Relaxations

The obvious linear programming relaxation of the Constrained Connectivity problems (and thus the iBGP problems) is the *flow LP*. For every pair $(u, v) \in V \times V$ let $\mathcal{P}_{uv}$ be the collection of $u - v$ paths that are contained in $S(u, v)$. The flow LP has a variable c_e for every edge $e \in E$ (called the *capacity* of edge e) and a variable $f(P)$ for every $u-v$ path in $\mathcal{P}_{uv}$ for every $(u, v) \in V \times V$ (called the *flow* assigned to path P). The flow LP simply requires that at least one unit of flow is sent between all pairs while obeying capacity constraints:

$$\begin{aligned}
&\min \textstyle\sum_e c_e \\
&\text{s.t} \textstyle\sum_{P \in \mathcal{P}_{uv}} f(P) \geq 1 && \forall (u, v) \in V \times V \\
&\quad \textstyle\sum_{P \in \mathcal{P}_{uv} : e \in P} f(P) \leq c_e && \forall e \in E, (u, v) \in V \times V \\
&\quad 0 \leq c_e \leq 1 && \forall e \in E \\
&\quad 0 \leq f(P) \leq 1 && \forall (u, v) \in V \times V, P \in \mathcal{P}_{uv}
\end{aligned}$$

This is obviously a valid relaxation of Constrained Connectivity-Sum: given a valid solution to Constrained Connectivity-Sum, let P_{uv} denote the required safe $u-v$ path for every $(u, v) \in V \times V$. For every edge e in some P_{uv} set c_e to 1, and set $f(P_{uv})$ to 1 for every $(u, v) \in V \times V$. This is clearly a valid solution to the linear program with the same value. To change the LP for Constrained Connectivity-Degree we can just introduce a new variable λ, change the objective function to $\min \lambda$, and add the extra constraints $\sum_{v:\{v,u\} \in E} c_{\{u,v\}} \leq \lambda$ for all $u \in V$. And while this LP can be exponential in size (since there is a variable for every path), it is also easy to design a compact representation that has only $O(n^4)$ variables and constraints. This compact representation has variables $f_{(u,v)}^{(x,y)}$ instead of $f(P)$, where $f_{(u,v)}^{(x,y)}$ represents the amount of flow from u to v along edge $\{u, v\}$ for the demand (x, y). Then we can write the normal flow conservation and capacity constraints for every demand (x, y) independently, restricted to $S(x, y)$.

3 Algorithms for iBGP and Constrained Connectivity on K_n

In this section we show that there is a $\tilde{O}(\sqrt{n})$-approximation algorithm for both Constrained Connectivity problems as long as the underlying graph is the complete graph K_n. This algorithm is inspired by the recent progress on directed

spanners by Bhattacharyya et al. [4], Dinitz and Krauthgamer [7], and Berman et al. [3]. In particular, we use the same two-component framework that they do: a randomized rounding of the LP and a separate random tree-sampling step. The randomized rounding we do is simple independent rounding with inflated probabilities. The next lemma implies that this works well when the safe sets are small.

Lemma 1. *Let $E' \subseteq E$ be obtained by adding every edge $e \in E$ to E' independently with probability at least $\min\{12c_e \cdot |S(x,y)| \ln n, 1\}$. Then with probability at least $1 - 1/n^3$, E' will have a path between x and y contained in $S(x,y)$.*

Another important part of our algorithm will be random sampling that is independent of the LP. We will use two different types of sampling: star sampling for the sum version and edge sampling for the degree version. First we consider star sampling, in which we independently sample nodes with probability p, and every sampled node becomes the center of a star that spans the vertex set.

Lemma 2. *All pairs with safe sets of size at least s will be satisfied by random star sampling with high probability if $p = (3 \ln n)/s$.*

For edge sampling, we essentially consider the Erdős-Rényi graph $G_{n,p}$, i.e. we just sample every edge independently with probability p. We will actually consider the union of $3 \log n$ independent $G_{n,p}$ graphs, where $p = \frac{(1+\epsilon)\log s}{s}$ for some small $\epsilon > 0$. Let H be this random graph.

Lemma 3. *With probability at least $1 - 1/n$, all pairs with safe sets of size at least s will be connected by a safe path in H.*

We will now combine the randomized rounding of the LP and the random sampling into a single approximation algorithm. Our algorithm is divided into two phases: first, we solve the LP and randomly include every edge e with probability $O(c_e \sqrt{n} \ln n)$. By Lemma 1 this takes care of safe sets of size at most $\sqrt{n}$. Second, if the objective is to minimize the number of edges we do star sampling with probability $(3 \ln n)/\sqrt{n}$, and if the objective is to minimize the maximum degree we do edge sampling using the construction of Lemma 3 with $s = \sqrt{n}$. It is easy to see that this algorithm with high probability results in a valid solution that is a $\tilde{O}(\sqrt{n})$-approximation (details can be found in the full version).

Theorem 2. *This algorithm is a $\tilde{O}(\sqrt{n})$-approximation to both* Constrained Connectivity-Sum *and* Constrained Connectivity-Degree *on K_n.*

4 Complexity of iBGP-Sum and iBGP-Degree

In this section we will show that the iBGP problems are $\Omega(\log n)$-hard to approximate by a reduction from Hitting Set (or equivalently from Set Cover). This is a much weaker hardness than the $2^{\log^{1-\epsilon} n}$ hardness that we prove for the general Constrained Connectivity problems in Section 5, but the iBGP problems

are much more restrictive. We note that this $\Omega(\log n)$ hardness is easy to prove for Constrained Connectivity on K_n; the main difficulty is constructing a metric so that the geometrically defined safe sets of iBGP have the structure that we want.

We begin by giving a useful gadget that encodes a Hitting Set instance as an instance of an iBGP problem in which all we care about is minimizing the degree of a particular vertex. We will then show how a simple combination of these gadgets can be used to prove that iBGP-Degree is hard to approximate, and how more complicated modifications to the gadget can be used to prove that iBGP-Sum is hard to approximate.

Suppose we are given an instance of hitting set with elements $1, 2, \ldots, n$ (note that we are overloading these as both integers and elements) and sets $T_1, T_2, \ldots, T_m$. Our gadget will contain a node x whose degree we want to minimize, a node a_i for all elements $i \in \{1, \ldots, n\}$, and a node b_{T_j} for each set T_j in the instance. We will also have four extra "dummy" nodes: z, y, u, and h. The following table specifies some of the distances between points. All other distances are the shortest path graph distances given these. Let M be some large value (e.g. 20), and let ϵ be some extremely small value larger than 0.

	x	z	y	a_i	b_{T_j}	u	h
x		M			$M + 1.4 + j\epsilon$		
z	M		1.5	$1 + i\epsilon$		2	
y		1.5					
a_i		$1 + i\epsilon$			$1 + (i + j)\epsilon$ (if $i \in T_j$)	1.1	
b_{T_j}	$M + 1.4 + j\epsilon$		$1 + (i + j)\epsilon$ (if $i \in T_j$)				$1 + j\epsilon$
u		2		1.1			
h					$1 + j\epsilon$		

It is easy to check that this is indeed a metric space. Informally, we want to claim that any solution to the iBGP problems on this instance must have an edge from x to a_i nodes such that the associated elements i form a hitting set. Here y, u, and h are nodes that force the safe sets into the form we want, and z is used to guarantee the existence of a small solution.

Lemma 4. *Let E be any feasible solution to the above iBGP instance. For every vertex b_{T_j} there is either an edge $\{x, b_{T_j}\} \in E$ or an edge $\{x, a_i\} \in E$ where $i \in T_j$.*

We now want to use this gadget to prove logarithmic hardness for iBGP-Sum. We will use the basic gadget but will duplicate x. So there will be ℓ copies of x, which we will call $x_1, x_2, \ldots, x_\ell$, and their distances are defined to be $d(x_i, z) = M + i\epsilon$ and $d(x_i, b_{T_j}) = M + 1.4 + (i + j)\epsilon$ with all other distances defined to be the shortest path. Note that all we did was modify the gadget to "break ties" between the x_i's. Let H be the smallest hitting set.

Lemma 5. *Any feasible* iBGP-Sum *solution has at least $\ell|H|$ edges.*

Lemma 6. *There is a feasible* iBGP-Sum *solution with at most* $\ell|H|+\ell+(m+n+4)^2$ *edges.*

Setting $\ell = (m+n+4)^2$ and combining these two lemmas with the known logarithmic hardness of Hitting Set gives us the following theorem:

Theorem 3. *It is NP-hard to approximate* iBGP-Sum *to a factor better than* $\Omega(\log N)$*, where* N *is the number of vertices in the metric.*

It is also fairly simple to modify the basic gadget to prove the same logarithmic hardness for iBGP-Degree. We do this by duplicating everything *other* than x, instead of duplicating x. This will force x to have the largest degree.

Theorem 4. *It is NP-hard to approximate* iBGP-Degree *to a factor better than* $\Omega(\log N)$*, where* N *is the number of vertices in the metric.*

5 Constrained Connectivity

In this section we consider the hardness of the Constrained Connectivity problems and the integrality gaps of the natural LP relaxations.

5.1 Hardness

We now show that the Constrained Connectivity-Sum and Constrained Connectivity-Degree problems are both hard to approximate to better than $2^{\log^{1-\epsilon} n}$ for any constant $\epsilon > 0$. We do this via a reduction from Min-Rep, a problem that is known to be impossible to approximate to better than $2^{\log^{1-\epsilon} n}$ unless NP $\subseteq$ DTIME($n^{\text{polylog}(n)}$) [13]. An instance of Min-Rep is a bipartite graph $G = (U, V, E)$ in which U is partitioned into groups $U_1, U_2, \ldots, U_m$ and V is partitioned into groups $V_1, V_2, \ldots, V_m$. There is a *super-edge* between U_i and V_j if there is an edge $\{u, v\} \in E$ such that $u \in U_i$ and $v \in V_j$. The goal is to find a minimum set S of vertices such that for all super-edges $\{U_i, V_j\}$ there is some edge $\{u, v\} \in E$ with $u \in U_i \cap S$ and $v \in V_j \cap S$. Vertices from a group that are in S are called the *representatives* of the group. It is easy to prove by a reduction from Label Cover that Min-Rep is hard to approximate to better than $2^{\log^{1-\epsilon} n}$, and in particular it is hard to distinguish the case when $2m$ vertices are enough (one from each group) from the case when $2m \times 2^{\log^{1-\epsilon} n}$ vertices are necessary [13].

Given an instance of Min-Rep, we want to convert it into an instance of Constrained Connectivity-Sum. We will create a graph with five types of vertices: vertices x_j^i for $j \in [m]$ and $i \in [d]$; vertices in U; vertices in V; vertices y_j^i for $j \in [m]$ and $i \in [d]$; and a special vertex z. Here the x nodes represent d copies of the groups of U and the y nodes represent d copies of the groups of V, where d is some parameter that we will define later. z is a dummy node that we will use to connect pairs that are not crucial to the analysis. Given this vertex set, there will be four types of edges: $\{x_j^i, u\}$ for all $j \in [m]$ and $i \in [d]$

and $u \in U_j$; $\{u, v\}$ for all edges $\{u, v\}$ in the original Min-Rep instance; $\{v, y_j^i\}$ for all $j \in [m]$ and $i \in [d]$ and $v \in V_j$; and $\{w, z\}$ for all vertices w.

Now that we have described the constrained connectivity graph, we need to define the safe sets. There are two types of safe sets: if in the original instance there is a super-edge between U_i and V_j then $S(x_i^k, y_j^k) = S(y_j^k, x_i^k) = \{x_i^k, y_j^k\} \cup U_i \cup V_j$ for all $k \in [d]$. All other safe sets consist of the two endpoints and z. Let e_{MR} denote the number of super-edges in the Min-Rep instance, let n_{MR} denote the number of vertices.

The following theorem shows that this reduction works. The intuition behind it is that a safe path between an x node and a y node corresponds to using the intermediate nodes in the path as the representatives of the groups corresponding to the x and y nodes, so minimizing the number of labels is like minimizing the number of edges incident on x and y nodes.

Theorem 5. *The original* Min-Rep *instance has a solution of size at most K if and only if there is a solution to the reduced Constrained Connectivity problem of size at most $Kd + e_{MR} + 2md + n_{MR}$.*

We can now set $d = n_{MR}^2$, which gives the following theorem:

Theorem 6. Constrained Connectivity-Sum *cannot be approximated better than $2^{\log^{1-\epsilon} n}$ for any $\epsilon > 0$ unless NP $\subseteq$ DTIME($n^{\text{polylog}(n)}$).*

To show that Constrained Connectivity-Degree has the same hardness we modify the above reduction so that along with having d copies of the x_j and y_j nodes there are also d^2 copies of the rest of the gadget. This lets us have a different copy of the Min-Rep instance for each pair of copies, forcing some vertex in some copy to have large degree.

Theorem 7. Constrained Connectivity-Degree *cannot be approximated better than $2^{\log^{1-\epsilon} n}$ for any constant $\epsilon > 0$ unless NP $\subseteq$ DTIME($n^{\text{polylog}(n)}$).*

5.2 Integrality Gap for Constrained Connectivity

We claim that the integrality gap of the flow LP relaxation is large for both Constrained Connectivity-Sum and Constrained Connectivity-Degree. The intuition is that we use a Min-Rep instance in which the edges between each group form a matching (i.e. an instance of *Unique Games*), allowing the LP to cheat by splitting the flow, but many representatives are needed for a valid integral solution. This instance is then changed into a Constrained Connectivity problem as in the hardness reduction. These results are in many ways similar to the $\Omega(n^{1/3-\epsilon})$ integrality gap for Min-Rep recently proved by Charikar et al. [6] and the $\Omega(n^{1/3-\epsilon})$ integrality gap for directed k-spanner by Dinitz and Krauthgamer [7], but the reduction to Constrained Connectivity adds some extra complications, especially when the objective is to minimize the maximum degree.

Theorem 8. *The flow LP for* Constrained Connectivity-Sum *has an integrality gap of $\Omega(n^{\frac{1}{3}-\epsilon})$ and the flow LP for* Constrained Connectivity-Degree *has an integrality gap of $\Omega(n^{\frac{1}{9}-\epsilon})$ for any constant $\epsilon > 0$.*

References

1. Basu, A., Ong, C.-H.L., Rasala, A., Shepherd, F.B., Wilfong, G.: Route oscillations in I-BGP with route reflection. In: Proc. ACM SIGCOMM (2002)
2. Bates, T., Chandra, R., Chen, E.: BGP route reflection - an alternative to full mesh IBGP. RFC 2796 (2000)
3. Berman, P., Bhattacharyya, A., Makarychev, K., Raskhodnikova, S., Yaroslavtsev, G.: Improved Approximation for the Directed Spanner Problem. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011. LNCS, vol. 6755, pp. 1–12. Springer, Heidelberg (2011)
4. Bhattacharyya, A., Grigorescu, E., Jung, K., Raskhodnikova, S., Woodruff, D.P.: Transitive-closure spanners. In: Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 932–941 (2009)
5. Buob, M.-O., Uhlig, S., Meulle, M.: Designing Optimal iBGP Route-Reflection Topologies. In: Das, A., Pung, H.K., Lee, F.B.S., Wong, L.W.C. (eds.) NETWORKING 2008. LNCS, vol. 4982, pp. 542–553. Springer, Heidelberg (2008)
6. Charikar, M., Hajiaghayi, M.T., Karloff, H.: Improved Approximation Algorithms for Label Cover Problems. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 23–34. Springer, Heidelberg (2009)
7. Dinitz, M., Krauthgamer, R.: Directed spanners via flow-based linear programs. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing, STOC 2011, pp. 323–332. ACM, New York (2011)
8. Dinitz, M., Wilfong, G.T.: iBGP and Constrained Connectivity. CoRR, abs/1107.2299 (2011), http://arxiv.org/abs/1107.2299
9. Fabrikant, A., Papadimitriou, C.: The complexity of game dynamics: BGP oscillations, sink equilibria, and beyond. In: SODA 2008: Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 844–853 (2008)
10. Feldman, M., Kortsarz, G., Nutov, Z.: Improved approximation algorithms for directed steiner forest. Journal of Computer and System Sciences 78(1), 279–292 (2012)
11. Griffin, T.G., Shepherd, F.B., Wilfong, G.: The stable paths problem and interdomain routing. IEEE/ACM Trans. Netw. 10(2), 232–243 (2002)
12. Griffin, T.G., Wilfong, G.: On the correctness of IBGP configuration. In: Proc. ACM SIGCOMM (September 2002)
13. Kortsarz, G.: On the hardness of approximating spanners. Algorithmica 30 (1999)
14. Levin, H., Schapira, M., Zohar, A.: Internet routing and games. In: STOC 2008: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, pp. 57–66. ACM, New York (2008)
15. Stewart, J.W.: BGP4: Inter-Domain Routing in the Internet. Addison-Wesley (1999)
16. Traina, P., McPherson, D., Scudder, J.: Autonomous system confederations for BGP. RFC 3065 (2001)
17. Vutukuru, M., Valiant, P., Kopparty, S., Balakrishnan, H.: How to Construct a Correct and Scalable iBGP Configuration. In: IEEE INFOCOM, Barcelona, Spain (April 2006)

Online Scheduling of Jobs with Fixed Start Times on Related Machines

Leah Epstein[1], Łukasz Jeż[2,3,⋆], Jiří Sgall[4,⋆⋆], and Rob van Stee[5]

[1] Department of Mathematics, University of Haifa, 31905 Haifa, Israel
lea@math.haifa.ac.il
[2] Institute of Computer Science, University of Wrocław, ul. Joliot-Curie 15, 50-383 Wrocław, Poland
lje@cs.uni.wroc.pl
[3] Institute of Mathematics, Academy of Sciences of the Czech Republic, Žitná 25, 115 67 Praha 1, Czech Republic.
[4] Computer Science Institute of Charles University, Faculty of Mathematics and Physics, Malostranské nám. 25, CZ-11800 Praha 1, Czech Republic
sgall@iuuk.mff.cuni.cz
[5] Max Planck Institute for Informatics, Saarbrücken, Germany
vanstee@mpi-inf.mpg.de

Abstract. We consider online preemptive scheduling of jobs with fixed starting times revealed at those times on m uniformly related machines, with the goal of maximizing the total weight of completed jobs. Every job has a size and a weight associated with it. A newly released job must be either assigned to start running immediately on a machine or otherwise it is dropped. It is also possible to drop an already scheduled job, but only completed jobs contribute their weights to the profit of the algorithm.

In the most general setting, no algorithm has bounded competitive ratio, and we consider a number of standard variants. We give a full classification of the variants into cases which admit constant competitive ratio (weighted and unweighted unit jobs, and C-benevolent instances, which is a wide class of instances containing proportional-weight jobs), and cases which admit only a linear competitive ratio (unweighted jobs and D-benevolent instances). In particular, we give a lower bound of m on the competitive ratio for scheduling unit weight jobs with varying sizes, which is tight. For unit size and weight we show that a natural greedy algorithm is 4/3-competitive and optimal on $m = 2$ machines, while for a large m, its competitive ratio is between 1.56 and 2. Furthermore, no algorithm is better than 1.5-competitive.

⋆ Partially supported by MNiSW grant N N206 368839, 2010–2013, UWr grant 1400/M/II/11, grant IAA100190902 of GA AV ČR, and a scholarship co-financed by an ESF project *Human Capital*.

⋆⋆ Partially supported by the Center of Excellence – Inst. for Theor. Comp. Sci., Prague (project P202/12/G061 of GA ČR).

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 134–145, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1 Introduction

Scheduling jobs with fixed start times to maximize (weighted) throughput is a well-studied problem with many applications, for instance work planning for personnel, call control and bandwidth allocation in communication channels [1,2]. In this paper, we consider it for uniformly related machines. In this problem, jobs with fixed starting times are released online to be scheduled on m machines. Each job needs to start immediately or else be rejected. The completion time of a job is determined by its length and the speed of a machine. As pointed out by Krumke et al. [3], who were the first to study them for uniformly related machines, problems like these occur when jobs or material should be processed immediately upon release, but there are different machines available for processing, for instance in a large factory where machines of different generations are used side by side. Because on identical machines the size of the job together with is fixed start time determine the time interval that one of the machines has to devote to the job in order to complete it, this problem is commonly known as *interval scheduling* [4,5,6,7,8]. In fact, Krumke et al. [3] used the name *interval scheduling on related machines* but we refrain from it as different speeds translate into different time intervals for different machines, albeit with a common start time.

We consider the preemptive version of this problem, where jobs can be preempted (and hence lost) at any time (for example, if more valuable jobs are released later). Without preemption, it is easy to see that no online algorithm can be competitive for most models. The only exception is the simplest version of this problem, where all jobs have unit size and weight. For this case, preemption is not needed.

1.1 Our Results

It is known (cf. Section 1.2) that if both the weight and the size of a job are arbitrary, then no (randomized) algorithm is competitive on identical machines, a special case of related machines. Therefore, we study several restricted models.

One of them is the case of jobs with unit sizes and unit weights, studied in Section 2. While a trivial greedy algorithm is 1-competitive in this case on identical machines (cf. Section 1.2), attaining this ratio on related machines is impossible. We give a lower bound of $(3 \cdot 2^{m-1} - 2)/(2^m - 1)$ on the competitive ratio for this case, which for large m tends to $3/2$ from below. The high level reason why this holds is that the optimal assignment of jobs to machines may depend on the timing of future arrivals. We also show that a simple greedy algorithm is 2-competitive and we use a more complicated lower bound construction to show that it is not better than 1.56-competitive for large m. For $m = 2$ machines, we show that it is $4/3$-competitive, matching the lower bound.

Next, in Section 3, we consider two extensions of this model: weighted unit-sized jobs and a model where the weight of a job is determined by a fixed function of its size. This last model includes the important case of proportional weights. A function $f : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ (where $\mathbb{R}_0^+$ denotes the non-negative reals) is C-benevolent if it is convex, $f(0) = 0$, and $f(p) > 0$ for all $p > 0$. This implies in

Table 1. An overview of old and new results for deterministic algorithms; upper bounds by randomized algorithms (UBr) are also given for a single machine. The upper bounds of m and $4m$ follow from Fact 1.1 below.

size, weight	1 machine			2 related machines		m related machines	
	LB	UB	UBr	LB	UB	LB	UB
1, 1	1	1 [10,4]	1 [10,4]	4/3	4/3	$\frac{3\cdot 2^{m-1}-2}{2^m-1}$	2
1, variable	4 [9]	4 [9]	2 [6]	2 [5]	4	1.693 [11,7]	4
variable, 1	1	1 [10,4]	1 [10,4]	2	2	m	m
variable, D-benevolent	3 [9][1]	4 [9]	2 [8]	2	8	m	$4m$
variable, C-benevolent	4 [9]	4 [9]	2 [8]	1.693 [11,7]	4	1.693 [11,7]	4
variable, proportional	4 [9]	4 [9]	2 [8]	1.693 [11,7]	4	1.693 [11,7]	4
variable, variable	∞ [9]	–	–	∞	–	∞	–

particular that f is continuous in $(0, \infty)$, and monotonically non-decreasing. We consider instances, called C-benevolent, where the weights of jobs are given by a fixed C-benevolent function f of their sizes. If $f(x) = ax$ for some $a > 0$, the weights are proportional. We give a 4-competitive algorithm, which can be used both for f-benevolent jobs and for weighted unit-sized jobs. This generalizes the results of Woeginger [9] for these models on a single machine; cf. Section 1.2.

Finally, in Section 4, we give a lower bound of m for unit-weight variable-sized jobs, which is tight due to a trivial 1-competitive algorithm for a single machine [10,4] and the following simple observation.

Fact 1.1. *If algorithm* Alg *is R-competitive on a single machine, then an algorithm that uses only the fastest machine by simulating* Alg *on it is $(R \cdot m)$-competitive on m related machines.*

Proof. Fix an instance and the optimum schedule for it on m related machines. To prove our claim, it suffices to show that a subset of jobs from that schedule with total weight no smaller than a $1/m$ fraction of the whole schedules weight can be scheduled on the fastest machine. Clearly, one of the machines is assigned a subset of sufficient weight in the optimum schedule, and this set can be scheduled on the fastest machine. □

Instances with unit-weight variable-sized jobs are a special case of D-benevolent instances: a function f is D-benevolent if it is decreasing on $(0, \infty)$, $f(0) = 0$, and $f(p) > 0$ for all $p > 0$. (Hence such functions have a discontinuity at 0.) Hence our lower bound of m applies to D-benevolent instances as well, and again we obtain an optimal (up to a constant factor) algorithm by combining a 4-competitive algorithm for a single machine [9] with Fact 1.1. Note that in contrast, C-benevolent functions are not a generalization of unit weights and variable sizes, because the constraint $f(0) = 0$ together with convexity implies that $f(cx) \geq c \cdot f(x)$ for all $c > 0, x > 0$, so the weight is at least a linear function of the size.

[1] This lower bound holds for all surjective functions.

We give an overview of our results and the known results in Table 1. In this table, a lower bound for a class of functions means that there *exists* at least one function in the class for which the lower bound holds.

1.2 Previous Work

As mentioned before, if both the weight and the size of a job are arbitrary, then no (randomized) algorithm is competitive, either on one machine [9,2] or identical machines [2]. For this general case on one machine, it is possible to give an $O(1)$-competitive algorithm, and even a 1-competitive algorithm, using constant resource augmentation on the speed; that is, the machine of the online algorithm is $O(1)$ times faster than the machine of the offline algorithm that it is compared to [12,13].

Faigle and Nawijn [4] and Carlisle and Lloyd [10] considered the version of jobs with unit weights on m identical machines. They gave a 1-competitive algorithm for this problem. Woeginger [9] gave optimal 4-competitive algorithms for unit sized jobs with weights, D-benevolent jobs, and C-benevolent jobs a single machine.

For unit sized jobs with weights, Fung et al. [5] gave a 3.59-competitive *randomized* algorithm for one and two (identical) machines, as well as a deterministic lower bound of 2 for two identical machines. The upper bound for one machine was improved to 2 by the same authors [6] and later generalized to the other nontrivial models [8]. See [11,14] for additional earlier randomized algorithms. A randomized lower bound of 1.693 for one machine was given by Epstein and Levin [11]; Fung et al. [7] pointed out that it holds for parallel machines as well, and gave an upper bound for that setting (not shown in the table): a 2-competitive algorithm for even m and a $(2 + 2/(2m - 1))$-competitive algorithm for odd $m \geq 3$.

1.3 Notation

There are m machines, $M_1, M_2 \ldots, M_m$, in order of non-increasing speed. Their speeds, all no larger than 1, are denoted $s_1, s_2 \ldots, s_m$ respectively. For an instance I and algorithm ALG, $\text{ALG}(I)$ and $\text{OPT}(I)$ denote the total weight of jobs completed by ALG and an optimal schedule, respectively. The algorithm is R-competitive if $\text{OPT}(I) \leq R \cdot \text{ALG}(I)$ for every instance I.

For a job j, we denote its size by $p(j)$, its release date by $r(j)$, and its weight by $w(j) > 0$; in Section 3 jobs are denoted by capital J's. Any job that an algorithm runs is executed in a half-open interval $[r, d)$, where $r = r(j)$ and d is the time at which the job completes or is preempted. We call such intervals *job intervals*. If a job (or a part of a job) of size p is run on machine M_i then $d = r + \frac{p}{s_i}$. A machine is called *idle* if it is not running any job, otherwise it is *busy*.

2 Unit Sizes and Weights

In this section we consider the case of equal jobs, i.e., all the weights are equal to 1 and also the size of each job is 1. We first note that it is easy to design a 2-competitive algorithm, and for 2 machines we find an upper bound of 4/3 for a natural greedy algorithm.

The main results of this section are the lower bounds. First we prove that no online algorithm on m machines can be better than $(3 \cdot 2^{m-1} - 2)/(2^m - 1)$-competitive. This matches the upper bound of 4/3 for $m = 2$ and tends to 1.5 from below for $m \to \infty$. For GREEDY on $m = 3n$ machines we show a larger lower bound of $(25 \cdot 2^{n-2} - 6)/(2^{n+2} - 3)$, which tends to $25/16 = 1.5625$ from below. Thus, somewhat surprisingly, GREEDY is not 1.5-competitive.

2.1 Greedy Algorithms and Upper Bounds

As noted in the introduction, in this case preemptions are not necessary. We may furthermore assume that whenever a job arrives and there is an idle machine, the job is assigned to some idle machine. We call such an algorithm *greedy-like*.

Fact 2.1. *Every greedy-like algorithm is 2-competitive.*

Proof. Let ALG be a greedy-like algorithm. Consider the following charging from the optimum schedule to ALG's schedule. Upon arrival of a job j that is in the optimum schedule, charge j to itself in ALG's schedule if ALG completes j; otherwise charge j to the job ALG is running on the machine where the optimum schedule assigns j. As every ALG's job receives at most one charge of either kind, ALG is 2-competitive. □

We also note that some of these algorithms are indeed no better than 2-competitive: If there is one machine with speed 1 and the remaining $m - 1$ have speeds smaller than $\frac{1}{m}$, an algorithm that assigns an incoming job to a slow machine whenever possible has competitive ratio no smaller than $2 - \frac{1}{m}$. To see this consider an instance in which $m - 1$ successive jobs are released, the i-th of them at time $i - 1$, followed by m jobs all released at time m. It is possible to complete them all by assigning the first $m - 1$ jobs to the fast machine, and then the remaining m jobs each to a unique machine. However, the algorithm in question will not complete any of the first $m - 1$ jobs before the remaining m are released, so it will complete exactly m jobs.

Algorithm GREEDY: Upon arrival of a new job: If some machine is idle, schedule the job on the fastest idle machine. Otherwise reject it.

While we cannot show that GREEDY is better than 2-competitive in general, we think it is a good candidate for such an algorithm. We support this by showing that it is optimal for $m = 2$.

Theorem 2.2. GREEDY *is* 4/3*-competitive algorithm for interval scheduling of unit size and weight jobs on 2 related machines.*

Proof. Consider a schedule of GREEDY and split it into independent intervals $[R_i, D_i)$ as follows. Let R_1 be the first release time. Given R_i, let D_i be the first time after R_i when both machines are available, i.e., each machine is either idle or just started a new job. Given D_i, let R_{i+1} be the first release time larger than or equal to D_i. Note that no job is released in the interval $[D_i, R_{i+1})$. Thus it is sufficient to show that during each $[R_i, D_i)$, the optimal schedule starts at most 4/3 times the number of jobs that GREEDY does.

At any time that a job j arrives in (R_i, D_i), both machines are busy in the schedule of GREEDY, as otherwise GREEDY could schedule it. An exception could be the case when GREEDY indeed scheduled j and one machine is idle, but then j's release time would be chosen as D_i. Thus any job that ADV starts in (R_i, D_i) can be assigned to the most recent job that GREEDY started on the same machine (possibly to itself). We get a one-to-one assignment between the jobs of ADV that arrive in (R_i, D_i), and the jobs of GREEDY that arrive in $[R_i, D_i)$. Hence the optimal schedule completes at most one additional job (the one started at time R_i on the idle machine). This proves the claim if GREEDY starts at least 3 jobs in $[R_i, D_i)$.

If GREEDY starts only one job in $[R_i, D_i)$, then so does the optimal schedule. If GREEDY starts two jobs in $[R_i, D_i)$, then the first job is started on M_1. No job is released in $[R_i, D_i)$ at or after the completion of the first job on the fast machine, as GREEDY would have scheduled it. It follows that the optimal schedule cannot schedule any two of the released jobs on the same machine and schedules also only two jobs. This completes the proof. □

2.2 Lower Bounds

We give two lower bounds, for any deterministic algorithm and for GREEDY that, with the number of machines tending to infinity, tend to 3/2 and 25/16 respectively from below. Due to space constraints, the rather tedious details of both constructions are left out.

For the first construction, we have m machines with geometrically decreasing speeds. The instance has two sets of jobs. The first part, I_m, is the set of jobs that both the algorithm and the adversary complete. The other part, E_m, consists of jobs that are completed only by the adversary.

Intuitively, the instance (I_m, E_m) can be described recursively. The set I_m contains one *leading* job j_m to be run on M_m plus two copies of I_{m-1}. One copy is aligned so that it finishes at the same time as j_m on M_m. The other is approximately aligned with the start of the job on M_m; we offset its release times slightly forward so that the first $m-1$ of its jobs are released before j_m. These jobs are actually the leading jobs $j_1, j_2, \ldots, j_{m-1}$ in the subinstances $I_1, I_2, \ldots, I_{m-1}$ along the first (leftmost) branch of the recursion tree. Because of the offsets of the release times, GREEDY runs the leading jobs $j_1, j_2, \ldots, j_m$ on $M_1, M_2, \ldots, M_m$, respectively. The adversary schedules these m jobs on different machines, cyclically shifted, so that one of them, namely the one running on M_1, finishes later than in GREEDY but the remaining $m-1$ finish earlier. Upon completion of each of these $m-1$ jobs, the adversary releases and schedules a job from E_m; the times are arranged so that at the time of release of any of these

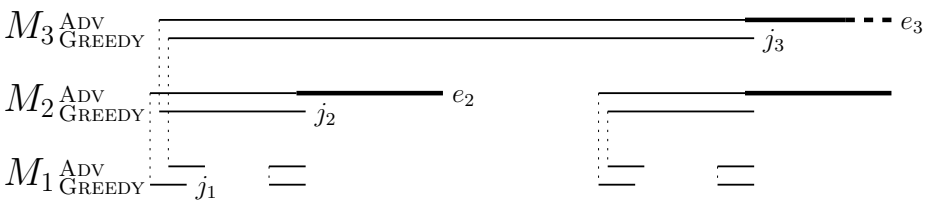

Fig. 1. The instance (I_3, E_3) for GREEDY. Common jobs in GREEDY and ADV schedule are joined using dotted lines. Jobs that only ADV completes are thicker. Note that machines M_2 and M_1 contain two instances of (I_2, E_2).

job from E_m, all the machines are busy in the schedule of GREEDY. In addition, E_m contains all the jobs from both sets E_{m-1} in the subinstances (I_{m-1}, E_{m-1}). This construction for $k = 3$ with $|I_3| = 7$ and $|E_3| = 3$ is illustrated in Figure 1; the constructions for $k = 1, 2$ appear as subinstances of a single job ($|I_1| = 1$, $E_1 = \emptyset$) and of four jobs ($|I_2| = 3$, $|E_2| = 1$) respectively.

The same idea works for a general algorithm in place of GREEDY, but we need to be more careful. Here the adversary dynamically determines the instance. The algorithm can use an arbitrary permutation to schedule the leading jobs. We let the adversary cyclically shift the jobs, so that on $m-1$ machines they start a little bit earlier than in the algorithm's assignment. However, this slightly disturbs the timing at the end of the subinstances, so that we cannot align them exactly. To overcome this, we need to change the offsets of the leading jobs, making them geometrically decreasing in the nested subinstances, and adjust the timing of the subinstances carefully depending on the actual schedule.

The proofs actually use a more convenient decomposition of the binary recursion tree: The instance (I_k, E_k) is decomposed into some jobs and subinstances (I_1, E_1), (I_2, E_2), ..., (I_{k-1}, E_{k-1}), each occurring exactly once. More precisely, I_k consists of all the k leading jobs and the sets I_1, I_2, ..., I_{k-1} that correspond to the subinstances whose completion is (approximately) aligned with the completion time of jobs $j_2, j_3, \ldots, j_k$. The actual exact timing of the subinstances I_1, I_2, ..., I_{k-1} depends on the algorithm's schedule of the leading jobs. Similarly, E_k contains $m - 1$ extra jobs corresponding to the leading jobs of I_k and the jobs from the sets E_1, E_2, ..., E_{k-1} that correspond to the same subinstances.

Theorem 2.3. *Let* ALG *be an online algorithm for interval scheduling of unit size and unit weight jobs on m related machines. Then the competitive ratio of* ALG *is at least* $(3 \cdot 2^{m-1} - 2)/(2^m - 1)$.

The second lower bound is higher, however it works only for GREEDY. We observe that cyclic shift of the leading jobs may not be the best permutation for the adversary. Instead, we create triplets of machines of the same speeds and shift the jobs cyclically among the triplets. I.e., the permutation of the leading jobs has three independent cycles of length $m/3$. Only for the three fastest machines we use different speeds and the previous construction as a subinstance.

Theorem 2.4. *The competitive ratio of the* Greedy *algorithm for interval scheduling of unit size and unit weight jobs on $m = 3n$ related machines is at least $(25 \cdot 2^{n-2} - 6)/(2^{n+2} - 3)$.*

3 Constant Competitive Algorithm for Two Input Classes

In this section we consider two types of instances. The first type are equal-sized jobs (of size 1, without loss of generality), whose weights can be arbitrary. We also consider input instances where the weights of jobs are given by a fixed C-benevolent function f of their sizes, that is, $w(J) = f((p(J))$. We call such an instance f-benevolent.

Algorithm Alg: On arrival of a new job J do the following.
1. Use an arbitrary idle machine if such a machine exists.
2. Otherwise, if no idle machines exist, preempt the job of minimum weight among the jobs running at time $r(J)$ having a weight less than $w(J)/2$ if such jobs exist.
3. If J was not scheduled in the previous steps, then reject it.

Note that we do not use the speeds in this algorithm in the sense that there is preference of slower or faster machines in any of the steps. But clearly, the eventual schedule does depend on the speeds.

Definition 3.1. *A* chain *is a maximal sequence of jobs $J_1, \ldots, J_n$ that* Alg *runs on one machine, such that J_j is preempted when J_{j+1} arrives ($j = 1, \ldots, n-1$).*

Observation 3.2. *For a chain $J_1, \ldots, J_n$ that* Alg *runs on machine i, J_1 starts running on an idle machine, and J_n is completed by* Alg. *Let $[r_j, d_j)$ be the time interval in which J_j is run ($j = 1, \ldots, n$). Then it holds that $r_j = r(J_j)$, $d_n - r_n = p(J_n)/s_i$, and finally $d_j - r_j < p(J_j)/s_i$, and $d_j = r_{j+1}$ for $j < n$.*

The following observation holds due to the preemption rule.

Observation 3.3. *For a chain $J_1, \ldots, J_n$, $2w(J_j) < w(J_{j+1})$ for $1 \leq j \leq n-1$.*

Consider a fixed optimal offline solution Opt, which runs all its selected jobs to completion. We say that a job J which is executed by Opt is *associated* with a chain $J_1, \ldots, J_n$ if Alg runs the chain on the machine where Opt runs J and J is released while this chain is running, i.e., $r(J) \in [r(J_1), d(J_n))$.

Claim. Every job J executed by Opt such that J is not the first job of any chain of Alg is associated with some chain.

Proof. Assume that J is not associated with any chain. The machine i which is used to execute J in Opt is therefore idle at the time $r(J)$ (before J is assigned). Thus, J is assigned in step 1 (to i or to another machine), and it is the first job of a chain. □

Thus, every job run by OPT but not by ALG is associated with a chain. We assume without loss of generality that every job in the instance either belongs to a chain or is run by OPT (or both), since other jobs have no effect on ALG and on OPT.

We assign every job that OPT runs to chains of ALG. The weight of a job J is split between J and the chain that J is associated with, where one of the two parts can be zero. In particular, if ALG does not run J then the first part must be zero, and if J is not associated with a chain then the second part must be zero. The assignment is defined as follows. Consider job J with release date r which OPT runs on machine i.

1. If J is not associated with any chain, assign a weight of $w(J)$ to J.
2. If J is associated with a chain of ALG (on machine i), let J' be the job such that $r(J) \in [r(J'), d(J'))$. Assign $\min\{w(J), 2 \cdot w(J')\}$ part of J to this chain, and assign the remainder $\max\{w(J) - 2 \cdot w(J'), 0\}$ part to J itself.

Note that for an f-benevolent instance, multiple jobs which are associated with a chain on a machine of speed s can be released while a given job J' of that chain is running, but only the last one can have weight above $w(J')$, since all other such jobs J satisfy $r(J) + \frac{p(J)}{s} \leq d(J')$ and $r(J) \geq r(J')$, so $p(J) \leq p(J')$ and by monotonicity $w(J) \leq w(J')$. The weight of all such jobs is assigned to the chain, while the last job associated with the chain may have some weight assigned to itself, if its weight is above $2w(J')$; this can happen only if ALG runs this job on another machine. This holds since if ALG does not run J, ALG does not preempt any of the jobs it is running, including the job J' on the machine that OPT runs J on, then $w(J) \leq 2w(J')$ (and J is fully assigned to the chain it is associated with). If ALG runs a job J on the same machine as OPT, then $J = J'$ must hold, and J is completely assigned to the chain (and not assigned to itself).

For any chain, we can compute the total weight assigned to the specific jobs of the chain (excluding the weight assignment to the entire chain).

Claim. For a chain $J_1, \ldots, J_n$ that ALG runs on machine i, the weight assigned to J_1 is at most $w(J_1)$. The weight assigned to J_k for $2 \leq k \leq n$ is at most $w(J_k) - 2w(J_{k-1})$. The total weight assigned to the jobs of the chain is at most $w(J_n) - \sum_{k=1}^{n-1} w(J_k)$.

Proof. The property for J_1 follows from the fact that the assigned weight never exceeds the weight of the job. Consider job J_k for $k > 1$. Then $w(J_k) > 2w(J_{k-1})$ by Observation 3.3. If there is a positive assignment to J_k, then the machine i' where OPT runs J_k is not i. At the time $r(J_k)$ all machines are busy (since the scheduling rule prefers idle machines, and J_k preempts J_{k-1}). Moreover, the job J' running on machine i' at time $r(J_k)$ satisfies $w(J') \geq w(J_{k-1})$. Thus J_k is assigned $w(J_k) - 2 \cdot w(J') \leq w(J_k) - 2w(J_{k-1})$. The total weight assigned to the jobs of the chain is at most $w(J_1) + \sum_{k=2}^{n} (w(J_k) - 2w(J_{k-1})) = w(J_1) + \sum_{k=2}^{n} w(J_k) - 2\sum_{k=1}^{n-1} w(J_k) = \sum_{k=1}^{n} w(J_k) - 2\sum_{k=1}^{n-1} w(J_k) = w(J_n) - \sum_{k=1}^{n-1} w(J_k)$. □

For a job J that has positive weight assignment to a chain of ALG it is associated with (such that the job J' of this chain was running at time $r(J)$), we define a pseudo-job $\pi(J)$. This job has the same release date time as J and its weight is the amount of J assigned to the chain, i.e., $\min\{w(J), 2 \cdot w(J')\}$. It is said to be assigned to the same chain of ALG that J is assigned to. If the input consists of unit jobs, then the size of $\pi(J)$ is 1. If the instance is an f-benevolent instance, then the size $p(\pi(J))$ of $\pi(J)$ is such that $f(p(\pi(J))) = 2w(J')$ (since f is continuous in $(0, \infty)$, and since there are values x_1,x_2 (the sizes of J, J') such that $f(x_1) = w(J') < 2w(J')$ and $f(x_2) = w(J) > 2w(J')$ then there must exist $x_1 < x_3 < x_2$ such that $f(x_3) = 2w(J')$), and $p(\pi(J)) \leq p(J)$.

Definition 3.4. *For a given chain $J_1, \ldots, J_n$ of* ALG *running on machine i, an* alt-chain *is a set of pseudo-jobs $J'_1, \ldots, J'_{n'}$ such that $r(J'_k) \geq r(J'_{k-1}) + \frac{p(J'_{k-1})}{s_i}$ for $2 \leq k \leq n'$, $r(J'_1) \geq r(J_1)$, $r(J'_{n'}) < d(J'_n)$, (that is, all jobs of the alt-chain are released during the time that the chain of* ALG *is running, and they can all be assigned to run on machine i in this order). Moreover, if $r(J'_k) \in [r_\ell, d_\ell)$, then $w(J'_k) \leq 2 \cdot w(J_\ell)$.*

Lemma 3.5. *For unit jobs, a chain $J_1, \ldots, J_n$ of* ALG *on machine i and any alt-chain $J'_1, \ldots, J'_{n'}$ satisfy*

$$\sum_{k=1}^{n'} w(J'_k) \leq \sum_{\ell=1}^{n} w(J_\ell) + 2w(J_n).$$

Proof. For every job J_ℓ, there can be at most one job of the alt-chain which is released in $[r_\ell, d_\ell)$, since the time to process a job on machine i is $\frac{1}{s_i}$ and thus difference between release times of jobs in the alt-chain is at least $\frac{1}{s_i}$, while $d_\ell \leq r_\ell + \frac{1}{s_i}$. However, every job of the alt-chain J'_k must have a job of the chain running at $r(J'_k)$. If job J'_k of the alt-chain has $r(J'_k) \in [r_\ell, d_\ell)$ then by definition $w(J'_k) \leq 2 \cdot w(J_\ell)$, which shows $\sum_{k=1}^{n'} w(J'_k) \leq 2 \sum_{\ell=1}^{n} w(J_\ell)$.

Using $w(J_k) > 2w(J_{k-1})$ for $2 \leq k \leq n$ we find $w(J_k) < \frac{w(J_n)}{2^{n-k}}$ for $1 \leq k \leq n$ and $\sum_{k=1}^{n-1} w(J_k) < w(J_n)$. Thus $\sum_{k=1}^{n'} w(J'_k) \leq \sum_{\ell=1}^{n} w(J_\ell) + 2w(J_n)$. □

Lemma 3.6. *For C-benevolent instances, a chain $J_1, \ldots, J_n$ of* ALG *on machine i and any alt-chain $J'_1, \ldots, J'_{n'}$ satisfy*

$$\sum_{k=1}^{n'} w(J'_k) \leq \sum_{\ell=1}^{n} w(J_\ell) + 2w(J_n).$$

We omit the proof due to space constraints, but note that it can be be deduced from a claim in the algorithm's original analysis for a single machine [9].

Observation 3.7. *For a chain $J_1, \ldots, J_n$ of* ALG*, the sorted list of pseudo-jobs (by release date) assigned to it is an alt-chain, and thus the total weight of pseudo-jobs assigned to it is at most $2 \sum_{\ell=1}^{n} w(J_\ell)$.*

Proof. By the assignment rule, every job which is assigned to the chain (partially or completely) is released during the execution of some job of the chain. Consider a pseudo-job J assigned to the chain, and let J' be the job of the chain executed at time $r(J)$.

The pseudo-job $\pi(J)$ has weight at most $\min\{w(J), 2 \cdot w(J')\}$. Since the set of pseudo-jobs assigned to the chain results from a set of jobs that OPT runs of machine i, by possibly decreasing the sizes of some jobs, the list of pseudo-jobs can still be executed on machine i. □

Theorem 3.8. *The competitive ratio of* ALG *is at most* 4 *for unit length jobs, and for C-benevolent instances.*

Proof. The weight allocation partitions the total weight of all jobs between the chains, thus it is sufficient to compare the total weight a chain was assigned (to the entire chain together with assignment to specific jobs) to the weight of the last job of the chain (the only one which ALG completes), which is $w(J_n)$.

Consider a chain $J_1, \ldots, J_n$ of ALG. The total weight assigned to it is at most $(w(J_n) - \sum_{k=1}^{n-1} w(J_k)) + (\sum_{\ell=1}^{n} w(J_\ell) + 2w(J_n)) = 4w(J_n)$. □

4 Lower Bound for Unit Weights and Variable Sizes

We give a matching lower bound to the upper bound of m shown in the introduction. Note that Krumke et al. [3] claimed an upper bound of 2 for this problem, which we show is incorrect.

Fix $0 < \varepsilon < \frac{1}{2}$ such that $\frac{1}{\varepsilon}$ is integer. Our goal is to show that no online algorithm can be better than $(1-\varepsilon)m$-competitive. We define $M = (\frac{1}{\varepsilon} - 1)m$ and $N = m^3 + Mm^2 + Mm$.

Input. One machine is fast and has speed 1. The other $m-1$ machines have speed $1/N$. The input sequence will consist of at most N jobs, which we identify with their numbers. Job j will have size $p(j) = 2^{N-j}$ and release time $r(j) \geq j$; we let $r(1) = 1$. The input consists of phases which in turn consist of subphases. Whenever a (sub)phase ends, no jobs are released for some time in order to allow the adversary to complete its most recent job(s). ALG will only be able to complete at most one job per full phase (before the next phase starts). The time during which no jobs are released is called a *break*.

Specifically, if ALG assigns job j to a slow machine or rejects it, the adversary assigns it to the fast machine instead, and we will have $r(j+1) = r(j) + p(j)$. We call this a *short break* (of length $p(j)$). A short break ends a subphase.

If ALG assigns job j to the fast machine, then in most cases, job j is *rejected* by the adversary and we set $r(j+1) = r(j)+1$. The only exception occurs when ALG assigns m consecutive jobs to the fast machine (since at most N jobs will arrive, and $p(j) = 2^{N-j}$, each of the first $m-1$ jobs is rejected by ALG when the next job arrives). In that case, the adversary assigns the first (i.e., largest) of these m jobs to the fast machine and the others to the slow machines (one job per machine). After the m-th job is released, no further jobs are released

until the adversary completes all these m jobs. The time during which no jobs are released is called a *long break*, and it ends a phase.

The input ends after there have been M long breaks, or if $m^2 + bm$ short breaks occur in total (in all phases together) before b long breaks have occurred. Thus the input always ends with a break. We claim (omitting the proof due to space constraints) that if there are $m^2 + bm$ short breaks in total before the b-th long break, ALG can complete at most $b-1+m$ jobs from the input (one per long break plus whatever jobs it is running when the input ends), whereas OPT earns $m^2 + bm$ during the short breaks alone. This implies a ratio of m and justifies ending the input in this case (after the $(m^2 + bm)$-th short break). If the M-th long break occurs, the input also stops. ALG has completed at most M jobs and can complete at most $m-1$ more. OPT completes at least Mm jobs in total (not counting any short breaks). The ratio is greater than $Mm/(M+m) = (1-\varepsilon)m$ for $M = (\frac{1}{\varepsilon} - 1)m$.

References

1. Awerbuch, B., Bartal, Y., Fiat, A., Rosén, A.: Competitive non-preemptive call control. In: Proc. of 5th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 1994), pp. 312–320 (1994)
2. Canetti, R., Irani, S.: Bounding the power of preemption in randomized scheduling. SIAM Journal on Computing 27(4), 993–1015 (1998)
3. Krumke, S.O., Thielen, C., Westphal, S.: Interval scheduling on related machines. Computers & Operations Research 38(12), 1836–1844 (2011)
4. Faigle, U., Nawijn, W.M.: Note on scheduling intervals on-line. Discrete Appliled Mathematics 58(1), 13–17 (1995)
5. Fung, S.P.Y., Poon, C.K., Zheng, F.: Online interval scheduling: randomized and multiprocessor cases. Journal of Combinatorial Optimization 16(3), 248–262 (2008)
6. Fung, S.P.Y., Poon, C.K., Zheng, F.: Improved Randomized Online Scheduling of Unit Length Intervals and Jobs. In: Bampis, E., Skutella, M. (eds.) WAOA 2008. LNCS, vol. 5426, pp. 53–66. Springer, Heidelberg (2009)
7. Fung, S.P.Y., Poon, C.K., Yung, D.K.W.: On-line scheduling of equal-length intervals on parallel machines. Information Processing Letters 112(10), 376–379 (2012)
8. Fung, S.P.Y., Poon, C.K., Zheng, F.: Improved randomized online scheduling of intervals and jobs. CoRR abs/1202.2933 (2012)
9. Woeginger, G.J.: On-line scheduling of jobs with fixed start and end times. Theoretical Computer Science 130(1), 5–16 (1994)
10. Carlisle, M.C., Lloyd, E.L.: On the k-coloring of intervals. Discrete Appliled Mathematics 59(3), 225–235 (1995)
11. Epstein, L., Levin, A.: Improved randomized results for the interval selection problem. Theoretical Computer Science 411(34-36), 3129–3135 (2010)
12. Kalyanasundaram, B., Pruhs, K.: Speed is as powerful as clairvoyance. Journal of the ACM 47(4), 617–643 (2000)
13. Koo, C., Lam, T.W., Ngan, T., Sadakane, K., To, K.: On-line scheduling with tight deadlines. Theoretical Computer Science 295, 251–261 (2003)
14. Seiden, S.S.: Randomized online interval scheduling. Operations Research Letters 22(4-5), 171–177 (1998)

A Systematic Approach to Bound Factor Revealing LPs and Its Application to the Metric and Squared Metric Facility Location Problems*

Cristina G. Fernandes[1], Luís A.A. Meira[2], Flávio K. Miyazawa, and Lehilton L.C. Pedrosa[3]

[1] Department of Computer Science, University of São Paulo, Brazil
cris@ime.usp.br
[2] Faculty of Technology, University of Campinas, Brazil
meira@ft.unicamp.br
[3] Institute of Computing, University of Campinas, Brazil
{fkm,lehilton}@ic.unicamp.br

Abstract. A systematic technique to bound factor-revealing linear programs is presented. We show how to derive a family of *upper bound* factor-revealing programs (UPFRP), and that each such program can be solved by a computer to bound the approximation factor. Obtaining an UPFRP is straightforward, and can be used as an alternative to analytical proofs, that are usually very long and tedious. We apply this technique to the Metric Facility Location Problem (MFLP) and to a generalization where the distance function is a squared metric. We call this generalization the Squared Metric Facility Location Problem (SMFLP) and prove that there is no approximation factor better than 2.04, assuming P $\neq$ NP. Then, we analyze the best known algorithms for the MFLP based on primal-dual and LP-rounding techniques when they are applied to the SMFLP. We prove very tight bounds for these algorithms, and show that the LP-rounding algorithm achieves a ratio of 2.04, and therefore has the best factor for the SMFLP. We use UPFRPs in the dual-fitting analysis of the primal-dual algorithms for both the SMFLP and the MFLP, improving some of the previous analysis for the MFLP.

1 Introduction

Let C and F be finite disjoint sets. Call *cities* the elements of C and *facilities* the elements of F. For each facility i and city j, let c_{ij} be a non-negative number representing the cost to connect i to j. Additionally, let f_i be a non-negative number representing the cost to open facility i. For each city j and subset F' of F, let $c(F', j) = \min_{i \in F'} c_{ij}$. The Facility Location Problem (FLP) consists of the following: given sets C and F, and c and f as above, find a subset F' of F such that $\sum_{i \in F'} f_i + \sum_{j \in C} c(F', j)$ is minimum. Hochbaum [7] presented an $O(\log n)$-approximation for the FLP.

* This research was partially supported by CNPq (grant numbers 306860/2010-4, 473867/2010-9, and 309657/2009-1) and FAPESP (grant number 2010/20710-4).

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 146–157, 2012.
© Springer-Verlag Berlin Heidelberg 2012

A well-studied particular case of the FLP is the METRIC FLP (MFLP), where the connection cost function is a *metric*, that is, c satisfies the triangle inequality $c_{ij} \leq c_{ij'} + c_{i'j'} + c_{i'j}$ for all $i, i' \in F$, and $j, j' \in C$. Several algorithms were proposed in the literature for the MFLP [2,6,8,9,10,12,13], and the best known algorithm is a 1.488-approximation proposed by Li [10]. Also, there is no approximation for the MFLP with a ratio smaller than 1.463, unless $\text{NP} \subseteq \text{DTIME}[n^{\text{O}(\log \log n)}]$ [6]. This result was later strengthened by Sviridenko, who showed that the lower bound holds unless $\text{P} = \text{NP}$ [14].

For the EUCLIDEAN FLP (EFLP), where facilities and cities are points in the Euclidean space, there is a PTAS by Arora, Raghavan, and Rao [1]. A variant of the EFLP is the SQUARED EUCLIDEAN FLP (E^2FLP), where the connection cost function is the square of the Euclidean distance. This cost function is known as ℓ_2^2 and was for instance considered by Jain and Vazirani [9, pp. 292–293]. Their approach implies a 9-approximation for the E^2FLP.

We consider a relaxed version of the triangle inequality: a connection cost function c is called a *squared metric*, if $\sqrt{c_{ij}} \leq \sqrt{c_{ij'}} + \sqrt{c_{i'j'}} + \sqrt{c_{i'j}}$ for all i, $i' \in F$, and j, $j' \in C$. The particular case of the FLP that only considers instances with a squared metric is called the SQUARED METRIC FLP (SMFLP). Notice that the SMFLP is a generalization of the E^2FLP and of the MFLP. The 9-approximation of Jain and Vazirani [9] applies also to the SMFLP and, to our knowledge, has the best known approximation factor. The choice of squared metrics discourages excessive distances in the solution. This effect is important in several applications, such as k-means and classification problems.

Although there are several algorithms for the MFLP in the literature, there are very few works on the SMFLP. Nevertheless, one may try to solve an instance of the SMFLP using good algorithms designed for the MFLP. Since these algorithms and their analysis are based on the assumption of the triangle inequality, it is reasonable to expect that they generate good solutions also for the SMFLP. However, there is no trivial way to derive an approximation factor from the MFLP to the SMFLP, so each algorithm must be individually reanalyzed. In this paper, we analyze three primal-dual algorithms (the 1.861 and the 1.61-approximation algorithms of Jain *et al.* [8], and the 1.52-approximation of Mahdian, Ye, and Zhang [12]) and an LP-rounding algorithm (Chudak and Shmoys's algorithm [4] used in the 1.5-approximation of Byrka and Aardal [2]) when applied to squared metric FLP instances. We show that these algorithms achieve ratios of 2.87, 2.43, 2.17, and 2.04 respectively. The last approximation factor is the best possible, as we show a 2.04-inapproximability limit for the SMFLP. This was obtained by extending the hardness results of Guha and Khuller [6] for the metric case. Although the primal-dual algorithms have larger factors, they are very fast [8], and so can be more interesting in practice.

The original analysis of the three primal-dual algorithms are based on the so called families of *factor-revealing linear programs* [8,12]. The lower bound on the approximation factor is given by a computer calculated solution of any program in such a family. The upper bound, however, is obtained analytically by bounding the value of every program in such a family, which requires long and

tedious non-straightforward proofs. In this paper, we propose a way to obtain a new family of upper bound factor-revealing programs for the SMFLP, as an alternative technique to achieve an upper bound. Now, the upper bound on the approximation factor is also obtained by a computer calculated solution of a single program. We note that, in our case, the factor-revealing programs are nonlinear, since the squared metric constraints contain square roots. We tackle this by replacing these constraints with an infinite set of linear constraints.

Recently, Mahdian and Yan [11] introduced the *strongly factor-revealing linear programs*. Our upper bound factor-revealing program is similar to a strongly factor-revealing program. The techniques involved in obtaining our program, however, are different. To obtain a strongly factor-revealing linear program, one projects a solution of an arbitrarily large linear program into a linear program with a constant number of variables, and guesses how to adjust the restrictions to obtain a feasible solution. In our approach, we define a candidate dual solution for a program with a fixed number of variables, and obtain an upper bound factor-revealing program directly in the form of a minimization program using only straightforward calculations. For the case of the SMFLP, we observed that calculating the dual upper bound program is easier than projecting the solutions on the primal. Also, we have considered the case of the MFLP, for which the obtained lower and upper bound factor-revealing programs converge.

Our contribution is two-folded. First, we make an important step towards generalizing the squared Euclidean distance, and successfully analyze this generalization in the context of FLP. Second, more importantly, we propose a new technique to systematically bound factor-revealing programs. This technique is used in the dual-fitting analysis of the primal-dual algorithms for both the SMFLP and the MFLP. We hope that this technique can also be used in the analysis of other dual-fitting algorithms analyzed through factor-revealing LPs.

A full version of this paper is available [5].

2 Preliminaries

Although the constraints over the cost function c from an SMFLP instance are defined by square roots, they are convex. Indeed, the next lemma shows that a squared metric can be expressed by an infinite set of linear inequalities. As a consequence, for any cost function not satisfying the squared metric inequality, there exists some linear inequality, as defined in Lemma 1, that is violated.

Lemma 1. *Let A, B, C, and D be non-negative numbers. Then $\sqrt{A} \leq \sqrt{B} + \sqrt{C} + \sqrt{D}$ if and only if $A \leq (1+\beta+\frac{1}{\gamma})B + (1+\gamma+\frac{1}{\delta})C + (1+\delta+\frac{1}{\beta})D$ for every positive numbers β, γ, and δ. In particular, if $\sqrt{A} \leq \sqrt{B} + \sqrt{C} + \sqrt{D}$, then $A \leq 3B + 3C + 3D$.*

We also use the concept of a bi-factor approximation algorithm, adopted in the context of the FLP for algorithms with distinct approximation factors for facility and connection costs. A bi-factor approximation for the FLP, as defined by Mahdian, Ye, and Zhang [12], is described in the following:

Definition 1 (Bi-factor approximation algorithm [12]). *An algorithm is called a* (γ_f, γ_c)*-approximation algorithm for the* FLP *if, for every instance* $\mathcal{I} = (C, F, c, f)$ *of the* FLP*, and for every solution* $S \subseteq F$ *for* $\mathcal{I}$ *with facility cost* $f(S) = \sum_{i \in S} f_i$ *and connection cost* $c(S) = \sum_{j \in C} c(S, j)$*, the cost of the solution produced by the algorithm is at most* $\gamma_f f(S) + \gamma_c c(S)$*.*

Jain *et al.* [8] showed that no algorithm is a (γ_f, γ_c)-approximation for the MFLP, with $\gamma_c < 1 + 2e^{-\gamma_f}$, unless NP $\subseteq$ DTIME$[n^{O(\log \log n)}]$. Following the lines of Sviridenko (see Vygen [14, Section 4.4]), the condition is changed to *unless* P = NP. We extend these results for the SMFLP as follows:

Theorem 1. *Let* γ_f *and* γ_c *be positive constants with* $\gamma_c < 1 + 8e^{-\gamma_f}$*. If there is a* (γ_f, γ_c)*-approximation for the* SMFLP*, then* P = NP*. In particular, let* $\alpha \approx 2.04011$ *be the solution of equation* $\gamma = 1 + 8e^{-\gamma}$*, then there is no* α'*-approximation with* $\alpha' < \alpha$ *for the* SMFLP *unless* P = NP*.*

3 A New Factor-Revealing Analysis

We analyze the algorithms of Jain *et al.* [8] using a new systematic factor-revealing technique. Their analysis uses a family of factor-reveling LPs parameterized by some k, so that, for a given k, the optimal value z_k of the LP is a *lower* bound on the approximation factor. To obtain the approximation factor, one has to analytically bound $\sup_{k \geq 1} z_k$. This is a nontrivial analysis, since it requires guessing a general suboptimal dual solution for the LP, usually inspired by numerically obtained dual LP solutions for small values of k. In this section, we show how to derive a new family of *upper bound factor-revealing programs* (UPFRP) parameterized by some t, so that, for any given t, the optimal value x_t of one such program is an upper bound on $\sup_{k \geq 1} z_k$. Obtaining an UPFRP and solving it using a computer is much simpler and more straightforward than using an analytical proof to obtain the approximation factor, since this does not include a guessing step and a manual verification of the feasibility of the solution. Additionally, as a property of the UPFRPs, we may tighten the obtained factor by solving the LP for larger values of t. In fact, in some cases (see Theorem 2 below), the lower and upper bound factor-revealing programs converge, that is, $\sup_{k \geq 1} z_k = \inf_{t \geq 1} x_t$.

We use an UPFRP to show that, when applied to SMFLP instances, the first algorithm of Jain *et al.* [8], denoted by $A1$, is a 2.87-approximation.

Algorithm $A1$ (C, F, c, f) *[8]*

1. Set $U := C$, meaning that every facility starts unopened, and every city unconnected. Each city j has some budget α_j, initially 0, and, at every moment, the budget that an unconnected city j offers to some unopened facility i equals to $\max(\alpha_j - c_{ij}, 0)$.
2. While $U \neq \emptyset$, the budget of each unconnected city is increased continuously until one of the following events occur:

(a) For some unconnected city j and some open facility i, $\alpha_j = c_{ij}$. In this case, connect city j to facility i and remove j from U.
(b) For some unopened facility i, $\sum_{j \in U} \max(\alpha_j - c_{ij}, 0) = f_i$. In this case, open facility i and, for every unconnected city j with $\alpha_j \geq c_{ij}$, connect j to i and remove it from U.

The analysis presented by Jain *et al.* [8] uses the dual fitting method. That is, their algorithms produce not only a solution for the MFLP, but also a vector $\alpha = (\alpha_1, \ldots, \alpha_{|C|})$ such that the value of the solution produced is equal to $\sum_j \alpha_j$. Moreover, for the first algorithm, following the dual fitting method, Jain *et al.* [8] proved that the vector $\alpha/1.861$ is a feasible solution for the dual linear program presented as (3) in [8], concluding that the algorithm is a 1.861-approximation for the MFLP. To present a similar analysis for the SMFLP, we use the same definitions and follow the steps of Jain *et al.* analysis. We start by adapting Lemma 3.2 from [8] for a squared metric.

Lemma 2. *For every $i \in F$, j, $j' \in C$, and α obtained by the first algorithm of Jain* et al. *[8] given an instance of the* SMFLP, $\sqrt{\alpha_j} \leq \sqrt{\alpha_{j'}} + \sqrt{c_{ij'}} + \sqrt{c_{ij}}$.

A facility i is said to be γ-*overtight* for some positive γ if, at the end of the algorithm, $\sum_j \max\left(\frac{\alpha_j}{\gamma} - c_{ij}, 0\right) \leq f_i$. Observe that, if every facility is γ-overtight, then the vector α/γ is a feasible solution for the dual linear program presented as (3) in [8]. Jain *et al.* proved that, for the MFLP, every facility is 1.861-overtight. We want to find a γ for the SMFLP, as close to 1 as possible, for which every facility is γ-overtight.

Fix a facility i. Let us assume without loss of generality that $\alpha_j \geq \gamma c_{ij}$ only for the first k cities. Following the lines of Jain *et al.* [8], we want to obtain the so called *factor-revealing* program. We define a set of variables f, d_j, and α_j, corresponding to facility cost f_i, distance c_{ij}, and city contribution α_j. Then, we capture the intrinsic properties of the algorithm using constraints over these variables. We assume without loss of generality that $\alpha_1 \leq \cdots \leq \alpha_k$. Also, we use Lemma 3.3 from [8], that states that the total contribution offered to a facility at any time is at most its cost, that is, $\sum_{l=j}^{k} \max(\alpha_j - d_l, 0) \leq f$. Besides these, we have the constraints from Lemma 2. Subject to all of these constraints, we want to find the minimum γ so that the facility is γ-overtight. In terms of the defined variables, we want the maximum ratio $\sum_{j=1}^{k} \alpha_j / (f + \sum_{j=1}^{k} d_j)$, resulting in

$$
\begin{array}{lll}
z_k = \max & \frac{\sum_{j=1}^{k} \alpha_j}{f + \sum_{j=1}^{k} d_j} & \\
\text{s.t.} & \alpha_j \leq \alpha_{j+1} & \forall\, 1 \leq j < k \\
& \sqrt{\alpha_j} \leq \sqrt{\alpha_l} + \sqrt{d_j} + \sqrt{d_l} & \forall\, 1 \leq j, l \leq k \\
& \sum_{l=j}^{k} \max(\alpha_j - d_l, 0) \leq f & \forall\, 1 \leq j \leq k \\
& \alpha_j, d_j, f \geq 0 & \forall\, 1 \leq j \leq k.
\end{array}
\tag{1}
$$

The next lemma has the same statement of Lemma 3.4 in [8], but it refers to program (1). Since the proof is the same, we omit it.

Lemma 3. *Let $\gamma = \sup_{k\geq 1} z_k$. Every facility is γ-overtight.*

Therefore $\sup_{k\geq 1} z_k$ is an upper bound on the approximation factor of the algorithm for the SMFLP. A slight modification of the example presented in Theorem 3.5 of [8] shows that this upper bound is tight.

3.1 A First Analysis Using Upper Bound Factor-Revealing Programs

Our first step is to relax (1) into a linear program. For that, we adjust the objective function as in [8], and we approximate the squared metric property by using inequalities given by Lemma 1. For simplicity, here we will use only the inequalities corresponding to $\beta = \gamma = \delta = 1$. With this, we will prove that $\sup_{k\geq 1} z_k$ is not greater than 3.236. We can improve this bound to 2.87 by using a whole set of inequalities from Lemma 1. See [5] for details. The relaxed factor-revealing linear program is:

$$\begin{array}{rll} w_k = \max & \sum_{j=1}^{k} \alpha_j & \\ \text{s.t.} & f + \sum_{j=1}^{k} d_j \leq 1 & \\ & \alpha_j \leq \alpha_{j+1} & \forall\, 1 \leq j < k \\ & \alpha_j \leq 3\alpha_l + 3d_j + 3d_l & \forall\, 1 \leq j, l \leq k \\ & x_{jl} \geq \alpha_j - d_l & \forall\, 1 \leq j \leq l \leq k \\ & \sum_{l=j}^{k} x_{jl} \leq f & \forall\, 1 \leq j \leq k \\ & \alpha_j, d_j, f, x_{jl} \geq 0 & \forall\, 1 \leq j, l \leq k. \end{array} \tag{2}$$

As (2) is a relaxation of (1), we have that $z_k \leq w_k$ and thus an upper bound on $\sup_{k\geq 1} w_k$ is also an upper bound on $\sup_{k\geq 1} z_k$. Solving linear program (2) using CPLEX for $k = 540$, we get that $\sup_{k\geq 1} w_k$ is at least 3.220. The next lemma uses a linear program to give a very tight bound on $\sup_{k\geq 1} w_k$.

Lemma 4. *For every k, $w_k \leq 3.236$.*

Proof. In what follows, we deduce an upper bound on w_k by deriving a linear minimization program whose feasible solutions are upper bounds on w_k. Then we present a feasible solution of value less than 3.236 for this program.

The idea is to determine a conical combination of the inequalities of (2) so that a given facility is γ-overtight for γ as small as possible. The linear minimization program will help us to choose the coefficients of such conical combination.

First rewrite the third inequality of program (2), so that the right-hand side is zero. For each j and l, we multiply the corresponding inequality by φ_{jl}. Denote by A the sum of all these inequalities, that is,

$$\sum_{j=1}^{k} \sum_{l=1}^{k} \varphi_{jl}(\alpha_j - 3\alpha_l - 3d_l - 3d_j) \leq 0.$$

The fourth and fifth inequalities of program (2) can be relaxed to the set of inequalities $\sum_{i=j}^{l_j}(\alpha_j - d_i) \leq f$, one for each l_j such that $j \leq l_j \leq k$. For each j and l_j, we multiply the corresponding inequality by θ_{jl_j} and denote by B the inequality resulting of summing them up, that is,

$$\sum_{j=1}^{k} \sum_{l=j}^{k} \theta_{jl} \sum_{i=j}^{l}(\alpha_j - d_i) \leq \left(\sum_{j=1}^{k} \sum_{l=j}^{k} \theta_{jl}\right) f.$$

The coefficients of α_j in A and B are $\text{coeff}_A[\alpha_j] = \sum_{l=1}^{k}(\varphi_{jl} - 3\varphi_{lj})$ and $\text{coeff}_B[\alpha_j] = \sum_{l=j}^{k}(l-j+1)\theta_{jl}$, respectively. The coefficients of $-d_j$ in A and B are $\text{coeff}_A[-d_j] = \sum_{l=1}^{k} 3(\varphi_{jl} + \varphi_{lj})$ and $\text{coeff}_B[-d_j] = \sum_{i=1}^{j}\sum_{l=j}^{k}\theta_{il}$.

Now, we sum inequalities A and B and obtain a new inequality C:

$$\sum_{j=1}^{k} \text{coeff}_C[\alpha_j]\ \alpha_j - \sum_{j=1}^{k} \text{coeff}_C[-d_j]\ d_j \leq \text{coeff}_C[f]\ f. \tag{3}$$

We want to find values for γ, θ_{jl}, and φ_{jl} so that the coefficients of C in inequality (3) imply, for sufficiently large k, that

$$\sum_{j=1}^{k} \alpha_j - \gamma \sum_{j=1}^{k} d_j \leq \gamma f. \tag{4}$$

Moreover, we want γ as small as possible. For (3) to imply (4), it is enough that, for each j, coefficient $\text{coeff}_C[\alpha_j] \geq 1$, $\text{coeff}_C[-d_j] \leq \gamma$, and $\text{coeff}_C[f] \leq \gamma$. Hence, this can be expressed by the following linear program.

$$\begin{array}{lll} y_k = \min & \gamma & \\ \text{s.t.} & \text{coeff}_C[\alpha_j] \geq 1 & \forall\ 1 \leq j \leq k \\ & \text{coeff}_C[-d_j] \leq \gamma & \forall\ 1 \leq j \leq k \\ & \text{coeff}_C[f] \leq \gamma & \\ & \varphi_{jl} \geq 0 & \forall\ 1 \leq j, l \leq k \\ & \theta_{jl} \geq 0 & \forall\ 1 \leq j \leq l \leq k. \end{array} \tag{5}$$

The interested reader may note that (5) is the dual of a relaxed version of the factor-revealing linear program (2). Thus, its optimal value is an upper bound on the optimal value of (2).

As w_k does not decrease for multiples of k, we may assume that k has the form $k = pt$ with p and t positive integers. We will use a scaling argument to create a linear minimization program with a small number of variables, and obtain a feasible solution for program (5) from a solution of the former program. Then, we will show that the value of the generated solution is bounded by the value of the small solution.

Consider variables $\gamma' \in \mathbb{R}_+$, $\varphi'_{jl} \in \mathbb{R}_+$ for $1 \leq j,\ l \leq t$, and $\theta'_{jl} \in \mathbb{R}_+$ for $1 \leq j \leq l \leq t$. For an arbitrary n, let $\hat{n} = \lceil \frac{n}{p} \rceil$. We obtain a candidate solution for program (5) by taking $\varphi_{jl} = \varphi'_{\hat{j}\hat{l}}/p$, $\theta_{jl} = \theta'_{\hat{j}\hat{l}}/p^2$, and $\gamma = \gamma'$. Let us calculate each coefficient of C for this solution.

$$\begin{aligned} \text{coeff}_C[\alpha_j] &= \textstyle\sum_{l=1}^{k}(\varphi_{jl} - 3\varphi_{lj}) + \sum_{l=j}^{k}(l-j+1)\theta_{jl} \\ &= \tfrac{1}{p}\textstyle\sum_{l=1}^{k}(\varphi'_{\hat{j}\hat{l}} - 3\varphi'_{\hat{l}\hat{j}}) + \frac{1}{p^2}\sum_{l=j}^{k}(l-j+1)\theta'_{\hat{j}\hat{l}} \\ &\geq \tfrac{1}{p}\textstyle\sum_{l=1}^{pt}(\varphi'_{\hat{j}\hat{l}} - 3\varphi'_{\hat{l}\hat{j}}) + \frac{1}{p^2}\sum_{l=p\,\hat{j}+1}^{pt}(l - p\,\hat{j})\theta'_{\hat{j}\hat{l}} \\ &= \tfrac{1}{p}\textstyle\sum_{l'=1}^{t} p(\varphi'_{\hat{j}l'} - 3\varphi'_{l'\hat{j}}) + \frac{1}{p^2}\sum_{l'=\hat{j}+1}^{t}\theta'_{\hat{j}l'}\sum_{i=0}^{p-1}(p\,l' - i - p\,\hat{j}) \\ &= \textstyle\sum_{l'=1}^{t}(\varphi'_{\hat{j}l'} - 3\varphi'_{l'\hat{j}}) + \frac{1}{p^2}\sum_{l'=\hat{j}+1}^{t}\theta'_{\hat{j}l'}(p^2\,l' - \frac{p(p-1)}{2} - p^2\,\hat{j}) \\ &\geq \textstyle\sum_{l'=1}^{t}(\varphi'_{\hat{j}l'} - 3\varphi'_{l'\hat{j}}) + \sum_{l'=\hat{j}+1}^{t}(l' - \hat{j} - \frac{1}{2})\theta'_{\hat{j}l'}. \end{aligned}$$

Straightforward calculations (see the full version [5]) show that

$$\text{coeff}_C[-d_j] \leq \sum_{l'=1}^{t} 3(\varphi'_{\hat{j}l'} + \varphi'_{l'\hat{j}}) + \sum_{i'=1}^{\hat{j}} \sum_{l'=\hat{j}}^{t} \theta'_{i'l'}, \text{ and } \text{coeff}_C[f] \leq \sum_{j'=1}^{t} \sum_{l'=\hat{j}}^{t} \theta'_{j'l'}.$$

Now, we want to find the minimum value of γ' and values for φ'_{jl} and θ'_{jl} such that the candidate solution for program (5) is feasible. We may define the following linear program, named the *upper bound factor-revealing program*.

$$\begin{array}{lll} x_t = \min & \gamma' & \\ \text{s.t.} & \sum_{l=1}^{t}(\varphi'_{jl} - 3\varphi'_{lj}) + \sum_{l=j+1}^{t}(l-j-\frac{1}{2})\theta_{jl} \geq 1 & \forall\, 1 \leq j \leq t \\ & \sum_{l=1}^{t} 3(\varphi'_{jl} + \varphi'_{lj}) + \sum_{i=1}^{j}\sum_{l=j}^{t}\theta'_{il} \leq \gamma' & \forall\, 1 \leq j \leq t \\ & \sum_{j=1}^{t}\sum_{l=j}^{t}\theta'_{jl} \leq \gamma' & \\ & \varphi'_{jl} \geq 0 & \forall\, 1 \leq j,l \leq t \\ & \theta'_{jl} \geq 0 & \forall\, 1 \leq j \leq l \leq t. \end{array} \tag{6}$$

Consider an optimal solution for program (5). Replacing it in (3), that is, in inequality C, we obtain $\sum_{j=1}^{k}\alpha_j - \gamma\sum_{j=1}^{k} d_j \leq \gamma f$. Thus, $w_k \leq \gamma = y_k$. Now, consider an optimal solution for program (6) and the corresponding generated solution for (5). We obtain $y_k \leq \gamma = \gamma' = x_t$, and conclude that $w_k \leq x_t$, for all t. Using CPLEX to solve (6), we get that $x_{800} \approx 3.23586 < 3.236$.

3.2 Improved Factor-Revealing Analysis Using UPFRPs

In Lemma 4, we obtained the minimization program (6) from a conical combination of constraints from program (2) that bounds the approximation factor. This process is similar to obtaining the dual and using a scaling argument. Indeed, we propose a systematic way to obtain an upper bound factor-revealing program.

Consider the dual program of a traditional maximization factor-revealing linear program for some k. Take k in the form $k = pt$, for a fixed t. We want to create a minimization program that mimics the dual, but depends only on t and bounds its optimal value for every k. The idea is to constrain the variables of the small program to obtain a feasible solution for the dual program. To obtain a linear program independent of k, we scale the variables by p. The strategy to obtain an upper bound factor-revealing program may be summarized as follows:

1. obtain the dual $P(k)$ of the lower bound factor-revealing linear program;
2. consider a block variable x'_i for variables $x_{(i-1)p+1}, \ldots, x_{(i-1)p+p}$ of $P(k)$;
3. identify each variable x_i with the block variable $x'_{\lceil i/p \rceil}$ scaled by p;
4. replace identified variables in $P(k)$, canceling factors p.

Denote the resulting program by $P'(t)$. If $P'(t)$ depends only on t, both in number of variables and constraints, then any feasible solution of $P'(t)$ is an upper bound on the solution of $P(pt)$ for every p. Also, if it is the case that the value of $P(k)$ is not greater than the value of $P(kt)$, for every t, then a solution of $P'(t)$ for any t is also a bound on the approximation factor. Therefore, we call $P'(t)$ an *upper bound factor-revealing program*.

Notice that we could also have used a relaxed version of the factor-revealing linear program, as in Lemma 4. Observe that the factor-revealing program is not required to be linear. If one constraint is nonlinear, but convex, then one can approximate this by a set of linear inequalities, and calculate the dual program normally. Using this strategy on program (1), one may show that the approximation factor of $A1$ is, in fact, 2.87.

If we apply this analysis to the metric case, we simplify and strengthen the original 1.861 analysis. In fact, we show that the values of the upper and lower bound factor-revealing programs converge. See the full version [5] for details.

Theorem 2. *Let z_k be the optimal value of program (5) in [8] and let x_k be the optimal value for the corresponding upper bound factor-revealing program. Then* $\sup_{k\geq 1} z_k = \inf_{k\geq 1} x_k$. *Moreover, for the* MFLP*, the approximation factor of $A1$ [8] is between* 1.814 *and* 1.816.

We also analyzed the second algorithm of Jain *et al.* [8] for the SMFLP. The algorithm, denoted by $A2$, is similar to $A1$, but each connected city j keeps contributing to unopened facilities. For the metric case, the approximation factor is 1.61. With a completely analogous reasoning, we derive the corresponding lower and upper bound factor-revealing programs for $A2$, and obtain the following.

Theorem 3. *The approximation factor of $A2$ when applied to* SMFLP *instances is between* 2.415 *and* 2.425.

4 Scaling and Greedy Augmentation

Algorithm $A2$ can also be analyzed as a bi-factor approximation algorithm. The analysis uses a factor-revealing linear program, and is similar to the previous analysis. Mahdian, Ye, and Zhang [12] observed that, due to the asymmetry between the approximation guarantee for the opened facilities cost and the connections cost, Algorithm $A2$ may be used to open facilities that are very economical. This gives rise to a two-phase algorithm, denoted here by $A3(\delta)$, based on scaling facility costs by a constant $\delta \geq 1$, and on the greedy augmentation technique [6]. First, we scale the facility costs by δ and run Algorithm $A2$ on the modified instance. Then, while there is a facility i that may be opened to decrease the total cost by g_i, we open the facility that maximizes the ratio g_i/f_i.

In [12], a factor-revealing linear program is used to analyze Algorithm $A3(\delta)$ using a somewhat different, but equivalent, greedy augmentation procedure. Such analysis may be used to balance a bi-factor of Algorithm $A2$ for the MFLP. As noticed by Byrka and Aardal [2], this analysis is not restricted to Algorithm $A2$, and applies to any bi-factor approximation for the FLP. Therefore, since it does not depend on the cost function being a metric, we can use it to balance a bi-factor approximation for the squared metric case. This result is precisely stated as follows.

Lemma 5 ([12]). *Consider a (γ_f, γ_c)-approximation for the* FLP*. Then, for every $\delta \geq 1$, Algorithm $A3(\delta)$ is a $(\gamma_f + \ln\delta + \varepsilon, 1 + \frac{\gamma_c - 1}{\delta})$-approximation for the* FLP.

For the metric case, Algorithm $A2$ is a $(1.11, 1.78)$-approximation. This and Lemma 5 give a $(1.52+\varepsilon)$-approximation for the MFLP. We present an analysis for the SMFLP using an upper bound factor-revealing program. Using straightforward calculations, we have the following:

Lemma 6. *Let $\gamma_f \geq 1$ be a fixed value and let $\gamma_c = x_k$, where*

$$\begin{array}{rll} x_k = \max & \frac{\sum_{j=1}^{k} \alpha_j - \gamma_f f}{\sum_{j=1}^{k} d_j} & \\ \text{s.t.} & \alpha_l \leq \alpha_{l+1} & \forall\, 1 \leq l < k \\ & r_{jl} \geq r_{j,l+1} & \forall\, 1 \leq j < l < k \\ & \sqrt{\alpha_l} \leq \sqrt{r_{jl}} + \sqrt{d_l} + \sqrt{d_j} & \forall\, 1 \leq j < l \leq k \\ & \sum_{j=1}^{l-1} \max(r_{jl} - d_j, 0) + \sum_{j=l+1}^{k} \max(\alpha_l - d_j, 0) \leq f & \forall\, 1 \leq l \leq k \\ & \alpha_j, d_j, f, r_{jl} \geq 0 & \forall\, 1 \leq j \leq l \leq k. \end{array} \tag{7}$$

Then, if $\gamma_c < \infty$, Algorithm $A2$ is a (γ_f, γ_c)-approximation for the SMFLP.

Theorem 4. *Algorithm $A3(2.0543)$ is a 2.17-approximation for the* SMFLP.

5 An Optimal Approximation Algorithm

Byrka and Aardal [2] (see also [3]) gave a 1.5-approximation for the MFLP combining a $(1.11, 1.78)$-approximation of Jain *et al.* [8] and a new analysis of the LP-rounding algorithm $CS(\gamma)$ of Chudak and Shmoys [4], that leads to a $(1.6774, 1.3737)$-approximation. Byrka and Aardal showed that $CS(\gamma)$ has the optimal bi-factor approximation $(\gamma, 1 + 2e^{-\gamma})$ for $\gamma \geq \gamma_0 \approx 1.6774$.

We show that $CS(\gamma)$ is a 2.04-approximation for the SMFLP, and thus has the best possible factor. We start with the natural linear program relaxation.

$$\begin{array}{rll} \min & \sum_{i \in F} y_i f_i + \sum_{j \in C} \sum_{i \in F} x_{ij} c_{ij} & \\ \text{s.t.} & \sum_{i \in F} x_{ij} = 1 & \forall j \in C \\ & x_{ij} \leq y_i & \forall i \in F, j \in C \\ & x_{ij}, y_i \geq 0 & \forall i \in F, j \in C. \end{array} \tag{8}$$

The corresponding integer variables y_i indicate whether facility i is open, and the corresponding integer variables x_{ij} indicate whether facility i serves city j in the solution. Algorithm $CS(\gamma)$ may be summarized as follows. First, a solution (x^*, y^*) of program (8) is obtained. Then, the fractional opening variables y_i^* are scaled by a factor $\gamma \geq 1$, $\overline{y}_i = \gamma y_i^*$, and variables $\overline{x}_{ij}$ are defined so that city j is served entirely by its closest facilities, obtaining a new solution $(\overline{x}, \overline{y})$. We may assume that this solution is *complete*, *i.e.*, for every city j and facility i, if $\overline{x}_{ij} > 0$, then $\overline{x}_{ij} = \overline{y}_i$, and that for every i, $\overline{y}_i \leq 1$, since, in either case, we can split facility i, and obtain an equivalent instance with these properties. Finally, a clustering of some of the facilities is obtained according to a given criterion, and a probabilistic rounding procedure is used to obtain the final solution. For a detailed description of the algorithm, see [2] (also [3]).

A facility i with $\overline{x}_{ij} > 0$ is called a *close facility* of city j, and the set of such facilities is denoted by C_j. Similarly, a facility i with $\overline{x}_{ij} = 0$ but $x^*_{ij} > 0$ is called a *distant facility* of j, and the set of such facilities is denoted by D_j. Let $F_j = C_j \cup D_j$. The analysis of $CS(\gamma)$ uses the notion of *average distance* between a city $j \in C$ and a subset of facilities $F' \subseteq F$ such that $\sum_{i \in F'} \overline{y}_i > 0$, defined as $d(j, F') = \frac{\sum_{i \in F'} c_{ij} \cdot \overline{y}_i}{\sum_{i \in F'} \overline{y}_i}$. For a city j, we also use some definitions from [3]: the average connection cost, $d_j = d(j, F_j)$; the average distance from close facilities, $d_j^{(c)} = d(j, C_j)$; the average distance from distant facilities, $d_j^{(d)} = d(j, D_j)$; and the maximum distance from close facilities, $d_j^{(\max)} = \max_{i \in C_j} c_{ij}$.

With these definitions, we may describe the clustering of the facilities. In each iteration, greedily select a city j, called the *cluster center*, such that the sum $d_j^{(c)} + d_j^{(\max)}$ is minimum, and build a cluster formed by j and its close facilities C_j. Remove j and every other city j' such that $C_j \cap C_{j'}$ is not empty, and repeat this process until every city is removed. The set of facilities opened by $CS(\gamma)$ is given by the following rounding procedure: for each cluster center j, open one facility i from C_j with probability $\overline{x}_{ij} = \overline{y}_i$, and, for each unclustered facility i, open it independently with probability $\overline{y}_i$. Each city is connected to its closest opened facility.

The following lemma of Byrka and Aardal [2] is used to bound the expected connection cost between a city and the closest facility from a set of facilities.

Lemma 7 ([2]). *Consider a random vector $y \in \{0,1\}^{|\mathcal{F}|}$ produced by Algorithm $CS(\gamma)$, a subset $A \subseteq \mathcal{F}$ of facilities such that $\sum_{i \in A} \bar{y}_i > 0$, and a city $j \in C$. Then, the following holds: $\mathbb{E}\left[\min_{i \in A, y_i = 1} c_{ij} \mid \sum_{i \in A} y_i \geq 1\right] \leq d(j, A)$.*

For a given city j, if one facility in C_j or D_j is opened, then Lemma 7 states that the expected connection cost is bounded by $d_j^{(c)}$ and $d_j^{(d)}$, respectively. If no facility in $C_j \cup D_j = F_j$ is opened, then city j can always be connected to one of the close facilities $C_{j'}$ of the associated cluster center j', with expected connection cost $d(j, C_{j'} \setminus F_j)$. Byrka and Aardal [2] showed that, for the MFLP, when $\gamma < 2$, this cost is at most $d_j^{(d)} + d_{j'}^{(\max)} + d_{j'}^{(c)}$. Since for the SMFLP we need $\gamma > 2$, we will use an improved version of this lemma by Li [10]. The adapted lemma for the squared metric is given in the following. The proof is the same, but we use the squared metric property, instead of the triangle inequality.

Lemma 8. *Let j be a city and j' be the associated cluster center such that $C_j \cap C_{j'} \neq \emptyset$. Then, $d(j, C_{j'} \setminus F_j) \leq 3 \cdot \left((2-\gamma) d_j^{(\max)} + (\gamma - 1) d_j^{(d)} + d_{j'}^{(\max)} + d_{j'}^{(c)}\right)$.*

Now, we can bound the expected facility and connection cost of a solution generated by $CS(\gamma)$. The next theorem extends Theorem 2.5 from [3].

Theorem 5. *For $\gamma \geq 1$, Algorithm $CS(\gamma)$ produces a solution (x, y) for the integer program corresponding to (8) with expected facility and connection costs $\mathbb{E}[y_i f_i] = \gamma \cdot F_i^*$, and $\mathbb{E}\left[\min_{i \in F, y_i = 1} c_{ij}\right] \leq \max\left\{1 + 8e^{-\gamma}, \frac{5e^{-\gamma} + e^{-1}}{1 - \frac{1}{\gamma}}\right\} \cdot C_j^*$, where $F_i^* = y_i^* f_i$ and $C_j^* = \sum_{i \in F} x_{ij}^* c_{ij}$.*

Let γ_0 be the solution of equation $(5e^{-\gamma} + e^{-1})/(1 - \frac{1}{\gamma}) = (1 + 8e^{-\gamma})$. For $\gamma \geq \gamma_0 \approx 2.00492$, the maximum connection cost factor is $1 + 8e^{-\gamma}$, so $CS(\gamma)$ touches the curve $(\gamma, 1 + 8e^{-\gamma})$, that is, its approximation factor is the best possible for the SMFLP, unless P = NP. The next theorem follows immediately.

Theorem 6. *Let $\alpha \approx 2.04011$ be the solution of equation $\gamma = 1 + 8e^{-\gamma}$. Then $CS(\alpha)$ is an α-approximation for the* SMFLP *and the approximation factor is the best possible unless* P = NP*.*

References

1. Arora, S., Raghavan, P., Rao, S.: Approximation schemes for Euclidean k-medians and related problems. In: Proc. 30th Annual ACM Symp. on Theory of Computing, pp. 106–113. ACM, New York (1998)
2. Byrka, J., Aardal, K.: An Optimal Bifactor Approximation Algorithm for the Metric Uncapacitated Facility Location Problem. SIAM J. on Comp. 39(6), 2212–2231 (2010)
3. Byrka, J., Ghodsi, M., Srinivasan, A.: LP-rounding algorithms for facility-location problems (2010), http://arxiv.org/abs/1007.3611
4. Chudak, F.A., Shmoys, D.B.: Improved Approximation Algorithms for the Uncapacitated Facility Location Problem. SIAM J. on Comp. 33(1), 1–25 (2003)
5. Fernandes, C.G., Meira, L.A.A., Miyazawa, F.K., Pedrosa, L.L.C.: Squared Metric Facility Location Problem (2012), http://arxiv.org/abs/1111.1672
6. Guha, S., Khuller, S.: Greedy Strikes Back: Improved Facility Location Algorithms. Journal of Algorithms 31(1), 228–248 (1999)
7. Hochbaum, D.S.: Heuristics for the fixed cost median problem. Mathematical Programming 22, 148–162 (1982)
8. Jain, K., Mahdian, M., Markakis, E., Saberi, A., Vazirani, V.V.: Greedy facility location algorithms analyzed using dual fitting with factor-revealing LP. Journal of the ACM 50(6), 795–824 (2003)
9. Jain, K., Vazirani, V.V.: Approximation algorithms for metric facility location and k-Median problems using the primal-dual schema and Lagrangian relaxation. Journal of the ACM 48(2), 274–296 (2001)
10. Li, S.: A 1.488 Approximation Algorithm for the Uncapacitated Facility Location Problem. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 77–88. Springer, Heidelberg (2011)
11. Mahdian, M., Yan, Q.: Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In: Proc. 43rd Annual ACM Symp. on Theory of Computing, pp. 597–606. ACM, New York (2011)
12. Mahdian, M., Ye, Y., Zhang, J.: Approximation Algorithms for Metric Facility Location Problems. SIAM J. on Comp. 36(2), 411–432 (2006)
13. Shmoys, D.B., Tardos, E., Aardal, K.: Approximation algorithms for facility location problems. In: Proc. 29th Annual ACM Symp. on Theory of Computing, pp. 265–274. ACM, New York (1997)
14. Vygen, J.: Approximation algorithms for facility location problems (Lecture Notes). Tech. Rep. 05950-OR, Research Institute for Discrete Mathematics, University of Bonn (2005)

Approximating Bounded Occurrence Ordering CSPs*

Venkatesan Guruswami[1] and Yuan Zhou[2]

[1] Computer Science Department
Carnegie Mellon University
guruswami@cmu.edu
[2] Computer Science Department
Carnegie Mellon University
yuanzhou@cs.cmu.edu

Abstract. A theorem of Håstad shows that for every constraint satisfaction problem (CSP) over a fixed size domain, instances where each variable appears in at most $O(1)$ constraints admit a non-trivial approximation algorithm, in the sense that one can beat (by an additive constant) the approximation ratio achieved by the naive algorithm that simply picks a random assignment. We consider the analogous question for ordering CSPs, where the goal is to find a linear ordering of the variables to maximize the number of satisfied constraints, each of which stipulates some restriction on the local order of the involved variables. It was shown recently that without the bounded occurrence restriction, for *every* ordering CSP it is Unique Games-hard to beat the naive random ordering algorithm.

In this work, we prove that the CSP with monotone ordering constraints $x_{i_1} < x_{i_2} < \cdots < x_{i_k}$ of arbitrary arity k can be approximated beyond the random ordering threshold $1/k!$ on bounded occurrence instances. We prove a similar result for all ordering CSPs, with arbitrary payoff functions, whose constraints have arity at most 3. Our method is based on working with a carefully defined Boolean CSP that serves as a proxy for the ordering CSP. One of the main technical ingredients is to establish that certain Fourier coefficients of this proxy constraint have substantial mass. These are then used to guarantee a good ordering via an algorithm that finds a good Boolean assignment to the variables of a low-degree bounded occurrence multilinear polynomial. Our algorithm for the latter task is similar to Håstad's earlier method but is based on a greedy choice that achieves a better performance guarantee.

1 Introduction

Constraint satisfaction. Constraint satisfaction problems (CSPs) are an important class of optimization problems. A CSP is specified by a finite set Π of relations, each of arity k, over a domain $\{0, 1, \ldots, D-1\}$, where k, D are some fixed constants. An instance of such a CSP consists of a set of variables V and a collection of constraints (possibly with weights) each of which is a relation from Π applied to some k-tuple of

* This research was supported in part by NSF CCF 1115525 and US-Israel BSF grant 2008293. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. A full version of this paper can be found hat http://eccc.hpi-web.de/report/2012/074/

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 158–169, 2012.
© Springer-Verlag Berlin Heidelberg 2012

variables from V. The goal is to find an assignment $\sigma : V \to D$ that maximizes the total weight of satisfied constraints. For example in the Max Cut problem, $k = D = 2$ and Π consists of the single relation $\mathrm{CUT}(a, b) = \mathbf{1}(a \neq b)$. More generally, one can also allow real-valued payoff functions $f : \{0, 1, \ldots, D-1\}^k \to \mathbb{R}^+$ in Π (instead of just $\{0, 1\}$-valued functions), with the goal being to find an assignment maximizing the total payoff.

Most Max CSP problems are NP-hard, and there is by now a rich body of work on approximation algorithms and hardness of approximation results for CSPs. Algorithmically, semidefinite programming (SDP) has been the principal tool to obtain good approximation ratios. In fact, SDP is universal for CSPs in the sense that under the Unique Games conjecture a certain canonical SDP achieves the optimal approximation ratio [Rag08]. However, many CSPs, including Max 3SAT, Max 3LIN, Max NAE-4-SAT, etc., are *approximation resistant*, meaning that for any $\epsilon > 0$, even when given a $(1 - \epsilon)$-satisfiable instance, it is hard to find an assignment that satisfies more than a fraction $r + \epsilon$ of the constraints, where r, the *random assignment threshold*, is the expected fraction of constraints satisfied by a random assignment [Hås01, AM09]. In other words, it is hard to improve upon the naive algorithm that simply picks a random assignment without even looking at the structure of the instance.

Let us call a CSP that is not approximation resistant as *non-trivially approximable*. Inspite of a rich body of powerful algorithmic and hardness results, we are quite far from a complete classification of all CSPs into approximation resistant or non-trivially approximable. Several partial results are known; for example, the classification is known for Boolean predicates of arity 3. It is known that every binary CSP (i.e., whose constraints have arity 2), regardless of domains size (as long as it is fixed), is non-trivially approximable via a SDP-based algorithm [GW95, EG04, Hås08]. In a different vein, Håstad [Hås00] showed that for *every* Boolean CSP, when restricted to sparse instances where each variable participates in a bounded number B of constraints, one can beat the random assignment threshold (by an amount that is at least $\Omega(1/B)$). Trevisan showed that for Max 3SAT beating the random assignment threshold by more than $O(1/\sqrt{B})$ is NP-hard, so some degradation of the performance ratio with the bound B is necessary [Tre01].

Ordering CSPs. With this context, we now turn to ordering CSPs, which are the focus of this paper. The simplest ordering CSP is the well-known Maximum Acyclic Subgraph (MAS) problem, where we are given a directed graph and the goal is to order the vertices $V = \{x_1, \ldots, x_n\}$ of the graph so that a maximum number of edges go forward in the ordering. This can be viewed as a "CSP" with variables V and constraints $x_i < x_j$ for each directed edge (x_i, x_j) in the graph; the difference from usual CSPs is that the variables are to be ordered, i.e., assigned values from $\{1, 2, \ldots, n\}$, instead of being assigned values from a fixed domain (of size independent of n).

An ordering CSP of arity k is specified of a constraint $\Pi : S_k \to \{0, 1\}$ where S_k is the set of permutations of $\{1, 2, \ldots, k\}$. An instance of such a CSP consists of a set of variables V and a collection of constraints which are (ordered) k-tuples. The constraint tuple $e = (x_{i_1}, x_{i_2}, \cdots, x_{i_k})$ is satisfied by an ordering of V if the local ordering of the variables $x_{i_1}, x_{i_2}, \cdots, x_{i_k}$, viewed as an element of S_k, belongs to the subset Π. The goal is to find an ordering that maximizes the number of satisfied constraint tuples. An example of an arity 3 ordering CSP is the *Betweenness* problem with constraints of the

form x_{i_2} occurs between x_{i_1} and x_{i_3} (this corresponds to the subset $\Pi = \{123, 321\}$ of S_3). More generally, one can allow more than one kind of ordering constraint, or even a payoff function $\omega_e : S_k \to \mathbb{R}^+$ for each constraint tuple e. The goal in this case is to find an ordering $\mathcal{O}$ that maximizes $\sum_e \omega_e(\mathcal{O}_{|e})$ where $\mathcal{O}_{|e}$ is the relative ordering of vertices in e induced by $\mathcal{O}$.

For the problem to decide whether all the constraints of an ordering CSP can be satisfied or not (i.e. the decision version), [GM06] showed a dichotomy theorem for 3-ary ordering CSPs. [BK10] proved a more generalised dichotomy theorem for a broader class of problems (the temporal CSPs) of all arities.

Now we turn to the optimization problem – to maximize the number of satisfied ordering constraints. Despite much algorithmic progress on CSPs, even for MAS there was no efficient algorithm known to beat the factor $1/2$ achieved by picking a random ordering. This was explained by the recent work [GMR08] which showed that such an algorithm does not exist under the Unique Games conjecture, or in other words, MAS is approximation resistant. This hardness result was generalized to all ordering CSPs of arity 3 [CGM09], and later to higher arities, showing that every ordering CSP is approximation resistant (under the UGC) [GHM+11].[1]

In light of this pervasive hardness of approximating ordering CSPs, in this work we ask the natural question raised by Håstad's algorithm for bounded occurrence CSPs [Hås00], namely whether bounded occurrence instances of ordering CSPs admit a non-trivial approximation. For the case of MAS, Berger and Shor [BS97] gave an efficient algorithm that given any directed graph of total degree D, finds an ordering in which at least a fraction $(1/2 + \Omega(1/\sqrt{D}))$ of the edges go forward. This shows that bounded occurrence MAS is non-trivially approximable. The algorithm is quite simple, though its analysis is subtle. The approach is to order the vertices randomly, and process vertices in this order. When a vertex is processed, if it has more incoming edges than outgoing edges (in the graph at that stage), all outgoing edges are removed, and otherwise all its incoming edges are removed. The graph remaining after all the vertices are processed is returned as the acyclic subgraph.

Evidently, this algorithm is tailored to the MAS problem, and heavily exploits its underlying graph-theoretic structure. It therefore does not seem amenable for extensions to give non-trivial approximations to other ordering CSPs.

Our Results. In this work, we prove that important special cases of ordering CSPs do admit non-trivial approximation on bounded occurrence instances. In particular, we prove this for the following classes of ordering CSPs:

1. The monotone ordering k-CSP for arbitrary k with constraints of the form $x_{i_1} < x_{i_2} < \cdots < x_{i_k}$ (i.e., the CSP defined by the constraint subset $\{123\ldots k\} \subseteq S_k$ consisting of the identity permutation). This can be viewed as the arity k generalization of the MAS problem. (Note that we allow multiple constraint tuples on the same *set* of k variables, just as one would allow 2-cycles in a MAS instance given by a directed graph.)

[1] This does not rule out non-trivial approximations for *satisfiable* instances. Of course for satisfiable instances of MAS, which correspond to DAGs, topological sorting satisfies all the constraints. For Betweenness, a factor $1/2$ approximation for satisfiable instances is known [CS98, Mak09].

2. All ordering CSPs of arity 3, even allowing for arbitrary payoff functions as constraints.

Our proofs show that these ordering CSPs admit an ordering into "4 slots" that beats the random ordering threshold. We remark that CSP instances which are satisfiable for orderings into n slots but do not admit good "c slot" solutions for any fixed constant c are the basis of the Unique Games hardness results for ordering CSPs [GMR08, GHM+11]. Our results show that for arity 3 CSPs and monotone ordering k-ary CSPs such gap instances cannot be bounded occurrence.

Our Methods. As mentioned above, the combinatorial approach of the Berger-Shor algorithm for MAS on degree-bounded graphs seems to have no obvious analog for more complicated ordering constraints. We prove our results by applying (after some adaptations) Håstad's algorithm [Hås00] to certain "proxy" Boolean CSPs that correspond to solutions to the ordering CSP that map the variables into a domain of size 4. The idea is to only consider orderings where the range is a small constant (like [4]) instead of [n]. This idea was also used in recent hardness results on ordering [GMR08, CGM09, GHM+11]. But the fact that one can afford to restrict the range even for algorithm design (in the case of some CSPs) is a surprise.

For the case of monotone ordering constraints (of arbitrary arity k), we prove that for this proxy payoff function on the Boolean hypercube, a specific portion of the Fourier spectrum carries non-negligible mass. This is the technical core of our argument. Once we establish this, the task becomes finding a Boolean assignment to the variables of a bounded-occurrence low-degree multilinear polynomial (namely the sum of the Fourier representations of all the constraints) that evaluates to a real number that is non-negligibly larger than the constant term (which is the random assignment threshold). We present a greedy algorithm for this latter task which is similar to Håstad's algorithm [Hås00], but yields somewhat better quantitative bounds.

Our result on general ordering 3-CSPs faces an additional complication since it can happen that the concerned part of Fourier spectrum is in fact zero for certain kinds of constraints. We identify all the cases when this troublesome phenomenon occurs, proving that in such cases the pay-off function can be expressed as a linear combination of arity 2 pay-off functions (accordingly, we call these cases as "binary representable" pay-off functions). If the binary representable portion of the pay-offs is bounded away from 1, then the remaining pay-offs (which we call "truly 3-ary") contribute a substantial amount to the Fourier spectrum. Fortunately, the binary representable portion of pay-offs can be handled by our argument for monotone ordering constraints (specialized to arity two). So in the case when they comprise most of the constraints, we prove that their contribution to the Fourier spectrum is significant and cannot be canceled by the contribution from the truly 3-ary pay-offs.

1.1 Outline for the Rest of the Paper

In Section 2, we formally define the ordering CSPs with bounded occurrence, and the proxy problems (the t-ordering version). We also introduce the notation and analytic tools we will need in the remainder of the paper. In Section 3, we present an algorithm which is a variant of Håstad's algorithm in [Hås00], and is used to solve the proxy

problems. In Section 4 and Section 5, we prove the two main theorems (Theorems 1 and 2) of the paper.

2 Preliminaries

2.1 Ordering CSPs, Bounded Occurrence Ordering CSPs

An *ordering* over vertex set V is an *injective* mapping $\mathcal{O} : V \to \mathbb{Z}^+$. An instance of k-ary monotone ordering problem $G = (V, E, \omega)$ consists of vertex set V, set E of k-tuples of distinct vertices, and weight function $\omega : E \to \mathbb{R}^+$. The weight satisfied by ordering $\mathcal{O}$ is

$$\mathsf{Val}^{\mathcal{O}}(G) \stackrel{\text{def}}{=} \sum_{e=(v_{i_1}, v_{i_2}, \cdots, v_{i_k}) \in E} \omega(e) \cdot \mathbf{1}_{\mathcal{O}(v_{i_1}) < \mathcal{O}(v_{i_2}) < \cdots < \mathcal{O}(v_{i_k})}.$$

We also denote the value of the optimal solution by

$$\mathsf{Val}(G) \stackrel{\text{def}}{=} \max_{\text{injective } \mathcal{O}: V \to \mathbb{Z}} \{\mathsf{Val}^{\mathcal{O}}(G)\}.$$

We can extend the definition of the monotone ordering problem to ordering CSPs $\mathcal{I} = (V, E, \Omega)$ with general pay-off functions, where V and E are similarly defined. For each k-tuple $e = (v_1, v_2, \cdots, v_k) \in E$, a general pay-off function $\omega_e \in \Omega$, mapping from all $k!$ possible orderings among $\mathcal{O}(v_1), \mathcal{O}(v_2), \cdots, \mathcal{O}(v_k)$ to $\mathbb{R}^{\geqslant 0}$, is introduced. That is, for an ordering $\mathcal{O}$, its pay-off $\omega_e(\mathcal{O})$ for constraint tuple e only depends on $\mathcal{O}_{|e}$, the relative ordering on vertices of e induced by $\mathcal{O}$. The overall pay-off achieved by an ordering $\mathcal{O}$ is defined as $\mathsf{Val}^{\mathcal{O}}(\mathcal{I}) \stackrel{\text{def}}{=} \sum_{e \in E} \omega_e(\mathcal{O})$. The optimal pay-off for the instance is then given by

$$\mathsf{Val}(\mathcal{I}) \stackrel{\text{def}}{=} \max_{\text{injective } \mathcal{O}: V \to \mathbb{Z}} \{\mathsf{Val}^{\mathcal{O}}(\mathcal{I})\}\ .$$

An ordering CSP problem $\mathcal{I} = (V, E, \Omega)$ (or a monotone ordering problem $G = (V, E, \omega)$) is called *B-occurrence bounded* if each vertex $v \in V$ occurs in at most B tuples in E.

2.2 The t-Ordering Version of Ordering CSPs

We start this section with several definitions. Two orderings $\mathcal{O}$ and $\mathcal{O}'$ are *essentially the same* if $\forall u, v \in V, \mathcal{O}(u) < \mathcal{O}(v) \Leftrightarrow \mathcal{O}'(u) < \mathcal{O}'(v)$, otherwise we call them *essentially different*. For a positive integer m, denote $[m] = \{1, 2, \ldots, m\}$. For integer $t > 0$, a *t-ordering* on V is a mapping $\mathcal{O}_t : V \to [t]$, not necessarily injective. An ordering $\mathcal{O}$ is *consistent* with a t-ordering $\mathcal{O}_t$, denoted by $\mathcal{O} \sim \mathcal{O}_t$, when $\forall u, v \in V, \mathcal{O}_t(u) < \mathcal{O}_t(v) \Rightarrow \mathcal{O}(u) < \mathcal{O}(v)$.

The monotone ordering problem G can be naturally extended to its t-ordering version, which is a regular CSP problem over domain $[t]$ defined as follows. For each constraint $e = (v_1, v_2, \cdots, v_k) \in E$, we introduce a pay-off function

$$\pi_e(\mathcal{O}_t) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{\mathcal{O} \sim \mathcal{O}_t} [\mathbf{1}_{\mathcal{O}(v_1) < \mathcal{O}(v_2) < \cdots < \mathcal{O}(v_k)}],$$

where the expectation is uniformly taken over all the essentially different orderings $\mathcal{O}$ that are consistent with $\mathcal{O}_t$. (In this paper, when $\mathcal{O}$ becomes a random variable for total ordering without further explanation, it is always uniformly taken over all essentially different orderings (that satisfy certain criteria).) Note that although π_e receives an n-dimensional vector as parameter in the equation above, its value depends only on the k values to $v_1, v_2, \cdots, v_k$. Then the k-ary CSP problem (t-ordering version of G) is to find the t-ordering $\mathcal{O}_t$ to maximize the objective function

$$\mathsf{Val}_t^{\mathcal{O}_t}(G) \stackrel{\text{def}}{=} \sum_{e \in E} \omega(e) \cdot \pi_e(\mathcal{O}_t).$$

We denote the value of the optimal solution by

$$\mathsf{Val}_t(G) \stackrel{\text{def}}{=} \max_{\mathcal{O}_t \in [t]^n} \{\mathsf{Val}_t^{\mathcal{O}_t}(G)\}.$$

We can also extend the ordering CSP problem $\mathcal{I}$ with general pay-off functions to its t-ordering version. For each constraint $e \in E$, the pay-off function in the t-ordering version is defined as $\pi_e(\mathcal{O}_t) \stackrel{\text{def}}{=} \mathbf{E}_{\mathcal{O} \sim \mathcal{O}_t}[\omega_e(\mathcal{O})]$. The pay-off achieved by a particular t-ordering $\mathcal{O}_t$ is given by $\mathsf{Val}_t^{\mathcal{O}_t}(\mathcal{I}) \stackrel{\text{def}}{=} \sum_{e \in E} \pi_e(\mathcal{O}_t)$, and the value of the optimal t-ordering solution is $\mathsf{Val}_t(\mathcal{I}) \stackrel{\text{def}}{=} \max_{\mathcal{O}_t \in [t]^n} \{\mathsf{Val}_t^{\mathcal{O}_t}(\mathcal{I})\}$.

Our approach to getting a good solution for (occurrence bounded) ordering CSPs is based on the following fact.

Fact 1. *For all positive integers t, $\mathsf{Val}(\mathcal{I}) \geqslant \mathsf{Val}_t(\mathcal{I})$.*

Note that for $t = 1$, $\mathsf{Val}_1(\mathcal{I})$ equals the expected pay-off of a random ordering. Since the monotone ordering problem is a special case of ordering CSP with general pay-off functions, Fact 1 is also true for the monotone ordering problem. By fact 1, it is enough to find a good solution for t-ordering version of $\mathcal{I}$ (or G) to show that $\mathsf{Val}(\mathcal{I})$ (or $\mathsf{Val}(G)$) is large.

2.3 Fourier Transform of Boolean Functions

For every $f : \{-1, 1\}^d \to \mathbb{R}$, we write the Fourier expansion of f as

$$f(x) = \sum_{S \subseteq [d]} \hat{f}(S) \chi_S(x),$$

where $\hat{f}(S)$ is the Fourier coefficient of f on S, and $\chi_S(x) = \prod_{i \in S} x_i$.

The Fourier coefficients can be computed by the inverse Fourier transform, i.e., for every $S \subseteq [d]$,

$$\hat{f}(S) = \mathop{\mathbf{E}}_{x \in \{-1,1\}^d}[f(x)\chi_S(x)].$$

3 Finding Good Assignments for Bounded Occurrence Polynomials

Let f be a polynomial in n variables $x_1, x_2, \ldots, x_n$ containing only multilinear terms of degree at most k with coefficients $\hat{f}(S)$. In other words, let $f(x) = \sum_{S\subseteq[n],|S|\leqslant k} \hat{f}(S)\chi_S(x)$. We say that f is D-occurrence bounded if for each coordinate $i \in [n]$, we have $|\{S \ni i : \hat{f}(S) \neq 0\}| \leqslant D$. We also define

$$|f| \stackrel{\text{def}}{=} \sum_{\emptyset\neq S\subseteq[n]} |\hat{f}(S)|.$$

Then, the following proposition shows us how to find a good assignment for f.

Proposition 1. *Given a D-occurrence bounded polynomial f of degree at most k, it is possible, in* $\mathrm{poly}(n, 2^k)$ *time, to find $x \in \{-1,1\}^n$ such that $f(x) \geqslant \hat{f}(\emptyset)+|f|/(2kD)$.*

Proof. We use the following algorithm to construct x.

Algorithm. As long as $|f| > 0$, the algorithm finds a non-empty set T that maximizes $|\hat{f}(T)|$, and let $\gamma = \hat{f}(T)$. We want to make sure we get $|\gamma|$ for credit while not losing too much other terms in $|f|$.

Note that for all $\emptyset \neq U \subsetneq T$, we have $\mathbf{E}_{z\in\{-1,1\}^T:\chi_T(z)=\mathrm{sgn}(\hat{f}(T))}[\chi_U(z)] = 0$, and therefore

$$\mathop{\mathbf{E}}_{z\in\{-1,1\}^T:\chi_T(z)=\mathrm{sgn}(\hat{f}(T))}\Big[\sum_{U\subseteq T}\hat{f}(U)\chi_U(z)\Big] = \hat{f}(\emptyset) + |\hat{f}(T)| = \hat{f}(\emptyset) + |\gamma|.$$

We can enumerate all the $z \in \{-1,1\}^T$ such that $\chi_T(z_{|T}) = \mathrm{sgn}(\hat{f}(T))$ to find a particular z^*, with $\sum_{U\subseteq T}\hat{f}(U)\chi_U(z^*) \geqslant \hat{f}(\emptyset) + |\gamma|$. We fix $x_{|T} = z^*$. For the rest of the coordinates, let $g : \{-1,1\}^{[n]\setminus T} \to \mathbb{R}$ be defined as,

$$g(y) \stackrel{\text{def}}{=} f(y, z^*), \forall y \in \{-1,1\}^{[n]\setminus T}.$$

We note that g is also a D-occurrence bounded polynomial f of degree at most k, and by fixing all the variables in T, we have

$$\hat{g}(\emptyset) = \sum_{U\subseteq T}\hat{f}(U)\chi_U(z^*) \geqslant \hat{f}(\emptyset) + |\gamma|.$$

On the other hand, observing that $|T| \leqslant k$ and $|\gamma|$ is an upper bound of all $|\hat{f}(S)|$ with $S \neq \emptyset$, we have

$$\begin{aligned}
|g| &= \sum_{\emptyset\neq S\subseteq[n]\setminus T}|\hat{g}(S)| = \sum_{\emptyset\neq S\subseteq[n]\setminus T}\Big|\sum_{U\subseteq T}\hat{f}(S\cup U)\chi_U(z^*)\Big| \\
&\geqslant \sum_{\emptyset\neq S\subseteq[n]\setminus T}|\hat{f}(S)| - \sum_{\emptyset\neq S\subseteq[n]\setminus T}\sum_{\emptyset\neq U\subseteq T}|\hat{f}(S\cup U)| \\
&\geqslant |f| - 2\sum_{S:S\cap T\neq\emptyset}|\hat{f}(S)| \\
&\geqslant |f| - 2\sum_{i\in T}\sum_{S\ni i}|\hat{f}(S)| \geqslant |f| - 2|T|D|\gamma| \geqslant |f| - 2kD|\gamma|.
\end{aligned}$$

Then we can use the two inequalities above to establish $\hat{g}(\emptyset) + |g|/(2kD) \geqslant \hat{f}(\emptyset) + |f|/(2kD)$.

By recursively applying this algorithm on g, we can eventually fix all the coordinates in x, and get a constant function whose value is at least $\hat{f}(\emptyset) + |f|/(2kD)$.

Remark 1. The algorithm is similar to Håstad's algorithm in [Hås00] but we make a greedy choice of the term $\chi_T(x)$ to satisfy (the one with the largest coefficient $|\hat{f}(T)|$) at each stage. Our analysis of the loss in $|g|$ is more direct and leads to a better quantitative bound, avoiding the loss of a "scale" factor (which divides all non-zero coefficients of the polynomial) in the advantage over $\hat{f}(\emptyset)$.

Remark 2. In the performance guarantee of the algorithm, $\hat{f}(\emptyset)$ corresponds to the value of random assignments in the later sections, while $|f|/(2kD)$ corresponds the advantage we get over random assignments. Because of the $1/D$ factor, our algorithm gives weaker gaurantee than Berger-Shor gives, but our algorithm extends to permutation CSPs of larger arities.

4 Bounded Occurrence Monotone Ordering Problem

Our main result in this section is the following.

Theorem 1. *For any constant $k > 1$, given a B-occurrence bounded k-ary monotone ordering problem $G = (V, E, \omega)$, there is a poly-time randomized algorithm to find a solution satisfying at least $\mathsf{Val}(G)(1/k! + \Omega_k(1/B))$ weight (in expectation).*

To prove the above theorem, we will show the following lemma.

Lemma 1. *For any constant $k > 1$, given a B-occurrence bounded k-ary monotone ordering problem $G = (V, E, \omega)$ with total weight W. Then it is possible, in polynomial time, to find a 4-ordering solution $\mathcal{O}_4$ with $\mathsf{Val}(G)(1/k! + \Omega_k(1/B))$ weight.*

Note that given Lemma 1, the randomized algorithm that samples ordering $\mathcal{O} \sim \mathcal{O}_4$ fulfills the task promised in the theorem.

Lemma 1 also implies the following.

Corollary 1. *For any B-occurrence bounded k-ary monotone ordering problem G, we have $\mathsf{Val}_4(G) \geqslant \mathsf{Val}(G)(1/k! + \Omega_k(1/B))$.*

Proof (Proof of Lemma 1)

We begin the proof with the analysis of the pay-off function $\pi_e : [4]^{\{v_1, v_2, \cdots, v_k\}} \to \mathbb{R}$ for some $e = (v_1, v_2, \cdots, v_k) \in E$. We can also view π_e as a real-valued function defined on Boolean cube $\{-1, 1\}^{2k}$, so that

$$\pi_e(x_1, x_2, \cdots, x_{2k}) = \pi_e\Big((1 - x_1) + \frac{(1 - x_2)}{2} + 1, \cdots, (1 - x_{2k-1}) + \frac{(1 - x_{2k})}{2} + 1\Big).$$

If we let $\Gamma(e)$ be the set of all $k!$ permutations of e, then

$$\begin{aligned}
&\sum_{e' \in \Gamma(e)} \mathop{\mathbf{E}}_{\mathcal{O}_4 \in [4]^k} [\pi_{e'}(\mathcal{O}_4)] \\
&= \sum_{e'=(v_{i_1}, v_{i_2}, \cdots v_{i_k}) \in \Gamma(e)} \mathop{\mathbf{E}}_{\mathcal{O}_4 \in [4]^k} \left[\mathop{\mathbf{E}}_{\mathcal{O} \sim \mathcal{O}_4} [\mathbf{1}_{\mathcal{O}(v_{i_1}) < \mathcal{O}(v_{i_2}) < \cdots < \mathcal{O}(v_{i_k})})] \right] \\
&= \mathop{\mathbf{E}}_{\mathcal{O}_4 \in [4]^k} \left[\mathop{\mathbf{E}}_{\mathcal{O} \sim \mathcal{O}_4} \left[\sum_{e'=(v_{i_1}, v_{i_2}, \cdots v_{i_k}) \in \Gamma(e)} \mathbf{1}_{\mathcal{O}(v_{i_1}) < \mathcal{O}(v_{i_2}) < \cdots < \mathcal{O}(v_{i_k})} \right] \right] = 1.
\end{aligned}$$

Since $\mathbf{E}_{\mathcal{O}_4 \in [4]^k}[\pi_{e'}(\mathcal{O}_4)]$ is the same for all $e' \in \Gamma(e)$, we know that $\mathbf{E}_{\mathcal{O}_4 \in [4]^k}[\pi_e(\mathcal{O}_4)] = 1/k!$. Hence we have the following fact.

Fact 2. $\widehat{\pi_e}(\emptyset) = \mathop{\mathbf{E}}_{x \in \{-1,1\}^{2k}}[\pi_e(x)] = \mathop{\mathbf{E}}_{\mathcal{O}_4 \in [4]^k}[\pi_e(\mathcal{O}_4)] = \dfrac{1}{k!}$.

By Fact 2, if we apply the algorithm in Proposition 1, to the objective function $f(x) = \sum_{e \in E} \omega(e)\pi_e(x)$ of the 4-ordering version, we are guaranteed to have a solution that is no worse than the random threshold $(1/k!)$. Then, we only need to identify some non-negligible weights on the rest of the Fourier spectrum of f.

Let $S_{\text{odd}} = \{2i-1 | i \in [k]\}$, and $S^+_{\text{odd}} = S_{\text{odd}} \cup \{2k\}$. We make the following claim whose proof is included in the full version of the paper.

Claim 1. $\widehat{\pi_e}(S^+_{\text{odd}}) = \dfrac{-2^{1-k} + 2^{2-2k}}{k!}$.

The above claim makes sure there is indeed non-negligible mass on non-empty-set Fourier coefficients for each constraint. Then we prove that, when summing up these constraints, either of the following two cases happens.

- Some weights shown in Claim 1 are not canceled by others, and finally appears in the non-empty-set Fourier coefficients for the final objective function f.
- Some weights are canceled by others, but in this case, the guarantee by $\hat{f}(\emptyset)$ itself beats $1/k!$ in terms of approximation ratio.

We define

$$\hat{\|}\pi_e\hat{\|} = \sum_{S \subseteq [2k] : \forall i \in [k], S \cap \{2i-1, 2i\} \neq \emptyset} |\widehat{\pi_e}(S)| .$$

Now Claim 1 implies $\hat{\|}\pi_e\hat{\|} = \Omega_k(1)$ for all $k \geqslant 2$. Let $\Gamma \subseteq E$ be a set of constraints sharing the same $\Gamma(e)$, and let us define, by abusing notation slightly

$$\omega(\Gamma) = \sum_{e \in \Gamma} \omega(e), \quad \omega_{\max}(\Gamma) = \max_{e \in \Gamma} \{\omega(e)\}, \quad \text{and} \quad \pi_\Gamma(x) = \sum_{e \in \Gamma} \omega(e)\pi_e(x) .$$

We treat $\pi_\Gamma : \{-1,1\}^{2k} \to \mathbb{R}$ as a real-valued function defined on a Boolean cube.

The idea of defining $\hat{\|}\pi_e\hat{\|}$ and Γ is as follows. The Fourier mass identified in Claim 1 could be canceled within Γ, but once the mass goes into $\hat{\|}\pi_\Gamma\hat{\|}$, it cannot be canceled by $\hat{\|}\pi_{\Gamma'}\hat{\|}$ for a different Γ', and will finally go into $|f|$. Then the following lemma shows that either $\hat{\|}\pi_\Gamma\hat{\|}$, or $\widehat{\pi_\Gamma}(\emptyset)$ alone, beats $\omega_{\max}(\Gamma)/k!$, where $\omega_{\max}(\Gamma)$ is an upperbound on the optimal solution's performance on the constraints in Γ.

Lemma 2. *For all* $0 < \alpha < 1$, *we have* $\widehat{\pi_\Gamma}(\emptyset) + \alpha\hat{\|}\pi_\Gamma\hat{\|} \geqslant \omega_{\max}(\Gamma)(\frac{1}{k!} + \alpha \cdot \Omega_k(1))$.

Proof. First, by Fact 2, we know that $\widehat{\pi_\Gamma}(\emptyset) = \sum_{e\in\Gamma} \widehat{\pi_e}(\emptyset) = \omega(\Gamma) \cdot \frac{1}{k!}$. Let $e^* \in \Gamma$ be the constraint with the most weight. If $\omega(e^*) = \omega_{\max}(\Gamma) \geqslant 2/3 \cdot \omega(\Gamma)$, we have

$$\begin{aligned}
\widehat{\pi_\Gamma}(\emptyset) + \alpha\hat{\|}\pi_\Gamma\hat{\|} &= \widehat{\pi_\Gamma}(\emptyset) + \alpha\hat{\|}\sum_{e\in\Gamma}\omega(e)\pi_e\hat{\|} \\
&\geqslant \widehat{\pi_\Gamma}(\emptyset) + \alpha\Big(\hat{\|}\omega(e^*)\pi_{e^*}\hat{\|} - \sum_{e\in\Gamma\setminus\{e^*\}}\hat{\|}\omega(e)\pi_e\hat{\|}\Big) \\
&= \omega(\Gamma)\cdot\frac{1}{k!} + \alpha\Big(\omega(e^*) - \sum_{e\in\Gamma\setminus\{e^*\}}\omega(e)\Big)\hat{\|}\pi_{e^*}\hat{\|} \\
&\geqslant \omega_{\max}(\Gamma)\Big(\frac{1}{k!} + \frac{\alpha}{2}\hat{\|}\pi_{e^*}\hat{\|}\Big) = \omega_{\max}(\Gamma)\Big(\frac{1}{k!} + \alpha\cdot\Omega_k(1)\Big).
\end{aligned}$$

where the last step follows from Claim 1. On the other hand, when $\omega(e^*) = \omega_{\max}(\Gamma) < 2/3 \cdot \omega(\Gamma)$,

$$\begin{aligned}
\widehat{\pi_\Gamma}(\emptyset) + \alpha\hat{\|}\pi_\Gamma\hat{\|} &\geqslant \widehat{\pi_\Gamma}(\emptyset) = \omega(\Gamma)\cdot\frac{1}{k!} \\
&> \omega_{\max}(\Gamma)\Big(\frac{1}{k!} + \frac{1}{2}\cdot\frac{1}{k!}\Big) = \omega_{\max}(\Gamma)\Big(\frac{1}{k!} + \Omega_k(1)\Big).
\end{aligned}$$

□

Given a k-ary monotone ordering problem $G = (V, E, \omega)$, we partition $E = \Gamma_1 \cup \Gamma_2 \cup \cdots \cup \Gamma_m$ into m disjoint groups, so that constraints e_j in each group Γ_i share a distinct $\Gamma(e)$ value. Then we write the objective function of its 4-ordering version as

$$f(x) = \sum_{e\in E}\omega(e)\pi_e(x) = \sum_{i=1}^{m}\sum_{e\in\Gamma_i}\pi_e(x) = \sum_{i=1}^{m}\pi_{\Gamma_i}(x),$$

where $f : \{-1,1\}^{2n} \to \mathbb{R}$ is defined on Boolean cube. For each $1 \leqslant i \leqslant m$, let $\{v_{i_1}, v_{i_2}, \cdots, v_{i_k}\}$ be the k vertices participating in Γ_i, then we note that for each $S \in \{2i_t - 1, 2i_t : t \in [k]\}$ that intersects with $\{2i_t - 1, 2i_t\}$ for each $t \in k$, we have $\hat{f}(S) = \widehat{\pi_{\Gamma_i}}(S)$, since all other constraints will have 0 as its Fourier coefficient over S. Then, for $\alpha \in (0, 1)$, we have

$$\hat{f}(\emptyset) + \alpha\cdot|f| \geqslant \sum_{i=1}^{m}\Big(\widehat{\pi_{\Gamma_i}}(\emptyset) + \alpha\cdot\hat{\|}\pi_{\Gamma_i}\hat{\|}\Big) \geqslant \sum_{i=1}^{m}\omega_{\max}(\Gamma_i)\Big(\frac{1}{k!} + \alpha\cdot\Omega_k(1)\Big), \quad (1)$$

where the last inequality is because of Lemma 2.

For each $\Gamma_i (1 \leqslant i \leqslant m)$, a total ordering $\mathcal{O}$ will satisfy at most $\omega_{\max}(\Gamma_i)$ weight of constraints. This give an upper bound of the optimal solution

$$\mathsf{Val}(G) \leqslant \sum_{i=1}^{m}\omega_{\max}(\Gamma_i). \quad (2)$$

Fix a coordinate $i \in [2k]$, each constraint π_e contributes at most 2^{k-1} non-zero Fourier coefficients containing i. Since $G = (V, E, \omega)$ is B-occurrence bounded, there are at most $B2^{k-1}$ non-zero Fourier coefficients of f containing i, therefore f is $B2^{k-1}$-occurrence bounded.

Applying Proposition 1 to f, the polynomial time algorithm gets a vector $x \in \{-1, 1\}^{2n}$, which corresponds to a 4-ordering $\mathcal{O}_4$ (recall that every two consecutive bits in x encode a value in $[4]$), such that

$$\begin{aligned}\mathsf{Val}_4^{\mathcal{O}_4}(G) = f(x) &\geqslant \hat{f}(\emptyset) + \frac{1}{k2^k B}|f| \\ &\geqslant \sum_{i=1}^{m} \omega_{\max}(\Gamma_i)\Big(\frac{1}{k!} + \Omega_k(1/B)\Big) \geqslant \mathsf{Val}(G)\Big(\frac{1}{k!} + \Omega_k(1/B)\Big),\end{aligned}$$

where the last two inequalities use (1) and (2) separately.

5 Bounded Occurrence 3-Ary Ordering CSP with General Pay-Off Functions

For a ordering CSP problem $\mathcal{I} = (V, E, \Omega)$ with general pay-off functions, we define

$$\mathsf{Rand}(\mathcal{I}) \stackrel{\text{def}}{=} \mathop{\mathbf{E}}_{\text{injective } \mathcal{O}:V\to\mathbb{Z}}[\mathsf{Val}^{\mathcal{O}}(\mathcal{I})],$$

as the performance of random ordering. Then we have our main result for 3-ary ordering CSPs:

Theorem 2. *Given a B-occurrence bounded 3-ary ordering CSP problem $\mathcal{I} = (V, E, \Omega)$ with general pay-off functions, there is a poly-time randomized algorithm, to find a solution satisfying at least* $\mathsf{Rand}(\mathcal{I}) + (\mathsf{Val}(\mathcal{I}) - \mathsf{Rand}(\mathcal{I})) \cdot \Omega(1/B)$ *weight (in expectation).*

The proof of Theorem 2 is included in the full version of this paper.

6 Concluding Remarks

In this paper, we investigate the problem whether there are non-trivial approximation algorithms for bounded occurrence ordering CSPs. By reducing the problem to a CSP over a fixed size domain, and applying a variant of Håstad's algorithm [Hås00], we give a positive answer for monotone ordering problems, and 3-ary ordering CSPs. The obvious open question left by our work is whether we can extend the technique presented in this paper to ordering CSPs with arbitrary arity. Given our approach, the following natural question arises in this vein: given maximum occurrence B, and arity k, does there exist a constant $t = t(B, k)$ so that it is enough to solve the t-ordering version to get a non-trivial approximate solution for the original ordering CSP? At first glance, one might believe that the answer to this question is "no", as t being independent of n seems too strong a restriction. But surprisingly, as Lemma 1 showed, even under a stronger restriction that $t = 4$ (which is independent of k as well), the answer is still "yes" for monotone ordering problems. In view of this special case, we believe that there is a generalization of our proof techniques to general bounded occurrence ordering CSPs, and leave the resolution of this as an open question.

References

[AM09] Austrin, P., Mossel, E.: Approximation resistant predicates from pairwise independence. Computational Complexity 18(2), 249–271 (2009); Preliminary version in CCC 2008

[BK10] Bodirsky, M., Kára, J.: The complexity of temporal constraint satisfaction problems. J. ACM 57, 9:1–9:41 (2010)

[BS97] Berger, B., Shor, P.W.: Tight bounds for the Maximum Acyclic Subgraph problem. J. Algorithms 25(1), 1–18 (1997)

[CGM09] Charikar, M., Guruswami, V., Manokaran, R.: Every permutation CSP of arity 3 is approximation resistant. In: Proceedings of the 24th IEEE Conference on Computational Complexity, pp. 62–73 (July 2009)

[CS98] Chor, B., Sudan, M.: A geometric approach to betweenness. SIAM J. Discrete Math. 11(4), 511–523 (1998)

[EG04] Engebretsen, L., Guruswami, V.: Is constraint satisfaction over two variables always easy? Random Structures and Algorithms 25(2), 150–178 (2004)

[GHM+11] Guruswami, V., Håstad, J., Manokaran, R., Raghavendra, P., Charikar, M.: Beating the random ordering is hard: Every ordering CSP is approximation resistant. SIAM J. Comput. 40(3), 878–914 (2011)

[GM06] Guttmann, W., Maucher, M.: Variations on an Ordering Theme with Constraints. In: Navarro, G., Bertossi, L.E., Kohayakawa, Y. (eds.) IFIP TCS. IFIP, vol. 209, pp. 77–90. Springer, Boston (2006)

[GMR08] Guruswami, V., Manokaran, R., Raghavendra, P.: Beating the random ordering hard: Inapproximability of maximum acyclic subgraph. In: Proceedings of the 49th IEEE Symposium on Foundations of Computer Science, pp. 573–582 (2008)

[GW95] Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM 42(6), 1115–1145 (1995)

[Hås00] Håstad, J.: On bounded occurrence constraint satisfaction. Inf. Process. Lett. 74(1-2), 1–6 (2000)

[Hås01] Håstad, J.: Some optimal inapproximability results. Journal of the ACM 48(4), 798–859 (2001)

[Hås08] Håstad, J.: Every 2-CSP allows nontrivial approximation. Computational Complexity 17(4), 549–566 (2008)

[Mak09] Makarychev, Y.: Simple linear time approximation algorithm for betweenness. Microsoft Research Technical Report MSR-TR-2009-74 (2009)

[Rag08] Raghavendra, P.: Optimal algorithms and inapproximability results for every CSP? In: Proceedings of the 40th ACM Symposium on Theory of Computing, pp. 245–254 (2008)

[Tre01] Trevisan, L.: Non-approximability results for optimization problems on bounded degree instances. In: Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, pp. 453–461 (2001)

On the NP-Hardness of Max-Not-2

Johan Håstad*

KTH Royal Institute of Technology

Abstract. We prove that, for any $\epsilon > 0$, it is NP-hard to, given a satisfiable instance of Max-NTW (Not-2), find an assignment that satisfies a fraction $\frac{5}{8} + \epsilon$ of the constraints. This, up to the existence of ϵ, matches the approximation ratio obtained by the trivial algorithm that just picks an assignment at random and thus the result is tight. Said equivalently the result proves that Max-NTW is approximation resistant on satisfiable instances and this makes our understanding of arity three Max-CSPs with regards to approximation resistance complete.

1 Introduction

In this paper we study the approximability of Maximum Constraint Satisfaction Problems, written more shortly as Max-CSPs. In the generic problem we are given a large number of constraints each affecting only a constant number of variables and the goal is to find an assignment to the variables that satisfies the maximal number of constraints. The most common domain of these variables is given by Boolean values and this is also the focus of this paper. Constraints can be of many forms, but the most studied case, and this is also the situation here, is when each constraint is of the form of a fixed predicate, P, applied to a sequence of literals corresponding to different variables. Almost all Max-CSPs are NP-hard and we turn to efficient approximation algorithms.

For such a maximization problem we say that an algorithm, A, is a C-approximation algorithm if it, for each instance I, outputs a number $A(I)$ such that $Opt(I) \geq A(I) \geq COpt(I)$ where $Opt(I)$ is the optimal value on instance I. Most algorithms will, at least with the help of randomness, in fact find an assignment that satisfies this $A(I)$ fraction of the constraints but we do not put this as a formal requirement.

The simple algorithm of picking an assignment uniformly at random gives a lower bound for approximability. In the basic situation this is just the probability, E_P, that a random assignment satisfies the defining predicate P. It is somewhat surprising, but mounting evidence [2] shows that for most predicates this is the best constant of approximability that can be guaranteed by any algorithm running in polynomial time. The predicates, for which this is indeed the best constant of approximability, are called *approximation resistant*.

An equivalent way to formulate approximation resistance is to say that, for any $\epsilon > 0$, it is NP-hard to distinguish instances for which the best assignment

* Funded by ERC Advanced Investigator Grant 226203.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 170–181, 2012.
© Springer-Verlag Berlin Heidelberg 2012

satisfies a fraction $1 - \epsilon$ of the constraints from those where this fraction is $E_P + \epsilon$. A slightly stronger property is that it is NP-hard to distinguish completely satisfiable instances from those where the best assignment satisfies a fraction $E_P + \epsilon$ of the constraints. We call such predicates *approximation resistant on satisfiable instances*.

If NP=P then all Max-CSPs are possible to solve optimally in polynomial time and thus the strongest results we can hope for is to prove a predicate approximation resistant based on the standard assumption NP≠P. Another frequently used assumption is the Unique Games Conjecture (UGC) proposed by Khot [9] in 2002. If we are willing to assume this conjecture, then as briefly mention above, if follows from [2] that most predicates are indeed approximation resistant. Let us note in passing that assuming the UGC the same authors [3] completely classify the property of being "useless" which is a generalization of approximation resistant.

When it comes to establishing approximation resistance on satisfiable instances much less is known. A key problem here is that the UGC does not have perfect completeness and thus cannot be used as a starting point for reductions. The question whether the difference between satisfiable instances and almost satisfiable instances is real or just a technicality is, in our eyes, not fully answered. The main example separating the two cases is parity and while there are some differences also for higher degree equations [7], also these examples have a strong smell of linear equations.

In this paper we focus on predicates of small arity and in particular on arity at most three. Using the seminal paper by Goemans and Williamson [4] it follows that all predicates of arity two are non-trivially approximable. When it comes to arity three, by combining the results of Zwick [16] and the results of Håstad [6], a predicate is approximation resistant iff it is implied by parity or its negation.

When turning to satisfiable instances, all problems of arity two can be solved perfectly while for arity three the situation is more interesting. For parity (of any size) it is the case that if all constraints can be satisfied simultaneously then such an assignment can be found efficiently by Gaussian elimination. At the other end, the predicates that are implied by parity (or its negation) and accept at least six of the eight inputs were proved to be approximation resistant on satisfiable instances already in [6]. In view of these results, the only approximation resistance problem of arity three that has remained open is the status of predicates accepting 5 inputs and being implied by parity (or its negation) when considering satisfiable instances. Such a predicate accepts the four strings accepted by parity (or its negation) and one more string. If we negate a suitable set of inputs to make this extra string the all zero string the predicate turns into "Not Two", which is true unless exactly two of the three Boolean inputs take the value true. As negating some inputs does not change the approximation resistance we may as well study this predicate and we call the resulting problem *Max-NTW*.

To address this problem O'Donnell and Wu [13] proved that, assuming the d-to-1 conjecture of Khot [9], that Max-NTW is approximation resistant on

satisfiable instances. The purpose of the current paper is to establish the same result based only on NP≠P and thus making our knowledge with respect to approximation resistance of predicates of arity at most three complete. Let us briefly discuss the methods used.

We, as do previous papers establishing similar results, obtain our result by producing a Probabilistically Checkable Proof (PCP), where the acceptance criteria is given by the target predicate (in our case "Not Two"). We follow the approach of [13] to a great extent and our starting point is a projection label cover instance. Such an instance is given by two sets of variables $u \in U$ and $v \in V$ all of which should given labels from the sets[1] $[M]$ and $[L]$, respectively, where for some pairs (u, v) we are given constraints in the form of projection operators π_{uv}. A labeling satisfies a given constraint iff $\pi_{uv}(l_v) = l_u$.

It turns out that, for any $\epsilon > 0$, it is NP-hard to distinguish the situation when all constraints can be satisfied from those where only a fraction ϵ of the constraints can be simultaneously satisfied. An interesting parameter here is the degeneracy[2] of the projections π_{uv} used in the instances constructed to prove hardness. In the known proofs of NP-hardness the degeneracy grows polynomially in ϵ^{-1}. The d-to-1 conjecture says that it is possible to obtain the same result with this parameter bounded by d, and just letting the sizes of the label sets go to infinity.

To prove our result we reuse several results from [13]. Their PCP for a projection label cover instance has a parameter δ and we use their protocol for a random choice of δ. We need one additional modification and that is to use instances of *smooth label cover* as introduced by Khot [11]. If we think of the label cover instance as a two prover game these instances are construted by sending a large set of identical questions to both provers.

Usually the key property of such instances is that unequal answers by the prover with the longer answers project to unequal answers of the other prover and this is the definition of "smooth". We use slightly more and in particular we need that we have copies of the original game within the extended game.

As is well known, Max-CSPs are in fact in one-to-one correspondence with non-adaptive PCPs. Thus our results establishes $\frac{5}{8}$ as the tight infimum of the soundness for any non-adaptive PCP that reads three bits with perfect completeness. This lower bounds was proved by Zwick [16] while, the upper bound by O'Donnell and Wu [13] was conditioned on the d-to-1 conjecture. The previously best upper bound based only on NP-completeness was $\frac{20}{27}$ by Khot and Saket [10]. For a longer discussion of these issues we refer to [13].

Finally let us note that we can make a Max-NTW into a two-prover games by sending a uniformly chosen constraint to one prover and a random variable from that constraint to the other prover. It is not difficult to see that this protocol has soundness $\frac{7}{8} + \epsilon$ and is three-to-one and this is, as far as we know, the best soundness for such a protocol.

[1] Any sets of given cardinalities works equally well.

[2] This is defined to be the maximal number of elements from the big set that project onto the same element.

2 Preliminaries

We use mostly standard notation. We use $\{-1, 1\}$-notation for Boolean variables with -1 corresponding to "true". We have real-valued functions of Boolean variables and the Fourier expansion is given by

$$f(x) = \sum_{\alpha} \hat{f}_\alpha \chi_\alpha(x),$$

where χ_α is a character function defined to equal $\prod_{i \in \alpha} x_i$.

We let $[M]$ denote the set of integers $0, 1 \ldots M-1$ and we are interested in projection operators π mapping $[L]$ to $[M]$. Any such operator creates a partitioning of $[L]$, defining the blocks to be the elements that map onto the same element. For a set $\beta \subseteq [L]$ we define $\pi(\beta)$ to the set of projected elements, i.e.,

$$\pi(\beta) = \{i \mid \exists j \in \beta,\ \pi(j) = i\}.$$

Given $g : \{-1, 1\}^L \mapsto \mathbb{R}$ we use the decomposition

$$g(y) = \sum_{\alpha} g^\alpha(y), \tag{1}$$

where

$$g^\alpha(y) = \sum_{\beta \mid \pi(\beta) = \alpha} \hat{g}_\beta \chi_\beta(y). \tag{2}$$

This decomposition in fact equals the Efron-Stein decomposition with regards to blocks in the partitioning defined above. For a longer discussion of the Efron-Stein decomposition and its properties we refer to [12].

Functions of special interest to us are the dictator functions $f(x) = x_i$ which are also known as the "long code of i". For tables in a PCP we use the standard techniques of folding to make sure that $f(-x) = -f(x)$.

We use correlated spaces as introduced by Mossel [12]. For a measure μ we use the L^2-norm by

$$\|f\|_2 = E_\mu[f(x)^2]^{1/2}.$$

Definition 1. *Suppose μ is a probability measure on $(X \times Y)$ such that the marginals of μ have full support. Define the* correlation *of X and Y under μ by*

$$\rho(X, Y, \mu) = \max E_\mu[f(x)g(y)],$$

where the maximum is taking over functions f and g such that $\|f\|_2 = \|g\|_2 = 1$ and $E[f] = E[g] = 0$.

It is important for us how correlated spaces behave under products and how they interact with the Efron-Stein decomposition. Given a sequence of correlated spaces $(X_i, Y_i, \mu_i)_{i=1}^n$ we can define the product space $((X_i)_{i=1}^n, (Y_i)_{i=1}^n, \prod_{i=1}^n \mu_i)$ and Proposition 2.13 of [12] establishes that the correlation of this product space is bounded by the maximum correlation of any underlying space.

Lemma 1. *[12] Let $(X_i \times Y_i)$ and μ_i be correlated spaces, then $\rho((X_i)_{i=1}^n, (Y_i)_{i=1}^n, \prod_{i=1}^n \mu_i) \leq \max_i \rho(X_i, Y_i, \mu_i)$.*

If g is a function on $(Y_i)_{i=1}^n$ and $g = \sum_S g_S$ is its Efron-Stein decomposition then by Proposition 2.12 of [12] we have the following lemma.

Lemma 2. *[12] Let $(X_i \times Y_i)$ and μ_i be correlated spaces with $\rho(X_i, Y_i, \mu_i) = \rho_i$, and $f : (X_i)_{i=1}^n \mapsto \mathbb{R}$ and $g : (Y_i)_{i=1}^n \mapsto \mathbb{R}$ then*

$$E[f(x)g_S(y)] \leq \|f\|_2 \|g_S\|_2 \prod_{i \in S} \rho_i.$$

3 From Label-Cover to a PCP

We start with a standard projection label cover instance and think of it as a two-prover game. In this game the verifier generates tuples (q_1, q_2, π) and sends question q_i to prover P_i. The prover P_2 gives an answer $a_2 \in [L]$ while P_1 answers $a_1 \in [M]$ and the verifier accepts iff $\pi(a_2) = a_1$. We here assume that π is at most d-to-1, in other words for any a_1 there are at most d different a_2 such that $\pi(a_2) = a_1$. The following lemma follows from the PCP-theorem [1] and Raz parallel repetition theorem [14].

Lemma 3. *For any $\epsilon > 0$ there exists a two-prover game with parameters M, L and d and where the verifiers uses $O(\log n)$ random bits, such that it is NP-hard to distinguish the cases when all constraints can be simultaneously satisfied from those where the optimal strategy of the provers makes the verifier accept with probability at most ϵ. The sizes of M, L, and d are polynomial in ϵ^{-1}.*

We make the two-prover-protocol more robust by generating T extra independent copies of the question q_2. These questions are sent to both players. Thus the prover P_2 thus get $T+1$ independent instances of its standard type of questions while P_1 gets T questions of the type initially sent to P_2 and one of its original type of questions. Let us denote these new type of questions by Q_2 and Q_1, respectively.

Both provers are supposed to answer all questions and the extended verifier accepts if it gets the same answer from the two provers on the questions sent to both provers and if the original verifier would have accepted the answers given to the standard questions. We call this protocol the T-extended protocol.

This protocol has similar properties to that of the the original, not extended, protocol. The parameters d and ϵ do not change while M and L do increase in the extended game and we reserve M and L to be used for these new values.

3.1 The PCP

For each question to one of the provers in the extended two-prover game we introduce a table which, in a correct proof of a correct statement, should be the long code of the answer to this question. We now have the below basic test, called NTW_δ, with a parameter δ. It is identical to the test used by O'Donnell and Wu [13].

Test NTW_δ

Written Proof. For each question Q_1 to P_1 we have a table $f_{Q_1} : \{-1,1\}^M \mapsto \{-1,1\}$ and similarly tables $g_{Q_2} : \{-1,1\}^L \mapsto \{-1,1\}$ for questions to P_2. These tables are folded over true.

Desired Property. To check that the tables form a long coding of a strategy in the T-extended game that makes the verifier of that game always accept.

Verifier

1. Choose a question (Q_1, Q_2, π) in the two prover game.
2. Choose $x \in \{-1,1\}^M$ and $y \in \{-1,1\}^L$ independently with the uniform probability and set $z_j = -y_j x_{\pi(j)}$ for all $j \in L$.
3. For each $i \in [M]$ with probability δ chose a random j such that $\pi(j) = i$ and set $z_j = y_j = x_i$.
4. Accept iff not two of the bits $f_{Q_1}(x)$, $g_{Q_2}(y)$ and $g_{Q_2}(z)$ are -1.

For each δ we define two parameters $s_\delta = c' \log(1/\delta)/\log d$, for a constant c' and $S_\delta = c'' \log(1/\delta) d^3 2^{2d} \delta^{-2}$ for a constant c''. We later find suitable values for these constants. We are now in a position to define our final test, which operates on the same type of written test and checks the same property.

Test $NTW^k_{\delta'}$

1. Set $\delta_0 = \delta'$ and for $i = 1, \ldots k-1$ choose δ_i such that $s_{\delta_i} = S_{\delta_{i-1}}$.
2. Pick a random $i \in [k]$ uniformly at random and run NTW_{δ_i}.

First note that, as s_δ tends to infinity when δ tends to 0, we do get a well defined sequence δ_i. We can also observe that $\log(1/\delta_k)$ is a constant that only depends on d, k and δ' and that it is bounded by a tower of exponentials of height around k.

4 Completeness and Soundness of Main PCP

Let us start by the easy completeness.

Lemma 4. *If label-cover instance is satisfiable, then there is a proof such that the verifier in $NTW^k_{\delta'}$ always accepts.*

Proof. Consider a written proof that is given by correct long codes of a strategy that always convinces the verifier in the extended two-prover game. In this situation, the three bits read are of the form $x_{\pi(j)}, y_j, z_j$ and these either have product -1 or are all 1. In either case the verifier accepts.

Let us turn to the more interesting problem of analyzing soundness. The key soundness lemma is the following.

Lemma 5. *For any $\epsilon' > 0$, and any basic two-prover games with parameters L, M, and d there are constants $\delta' > 0$, k and T such that if the verifier accepts in $NTW^k_{\delta'}$ with probability at least $\frac{5}{8} + \epsilon'$, then there is a strategy for the provers in the basic two-prover game that makes that verifier accept with probability ϵ'^2.*

We can conclude that for any $\epsilon' > 0$, for appropriate values of δ', k and T, the soundness of $NTW^k_{\delta'}$ is at most $\frac{5}{8} + \epsilon'$. This follows by choosing $\epsilon < \epsilon'^2$ obtaining parameters L, M, and d such that the soundness of the basic two-prover game is at most ϵ and then using values of δ', k and T produced by Lemma 5.

As all involved numbers are constants we get, by the standard translation from a PCP with a given acceptance criteria to the Max-CSP with the same predicate, our main theorem.

Theorem 1. *For any $\epsilon > 0$ it is NP-hard to approximate Max-NTW within $\frac{5}{8} + \epsilon$ on satisfiable instances.*

All that remains is to prove Lemma 5.

Proof. We analyze NTW_δ for a fixed value of δ. For readability let us drop the subscripts on the functions f and g, as well as the parameters s and S.

Arithmetizing the predicate "Not Two" we see that

$$\frac{5 + f(x) + g(y) + g(z) + f(x)g(y) + f(x)g(z) + g(y)g(z) - 3f(x)g(y)g(z)}{8} \tag{3}$$

is one if the verifier accepts and zero otherwise. We need to analyze the expected value of this quantity and we start by noting that each of x, y and z is uniformly random and that y and z are symmetric.

From the uniformity of x, y and z, it follows, by folding, that the first three non-trivial terms in (3) have expectation 0. For the next three terms we use the analysis of [13] (details omitted from this abstract) which proves that each of them is bounded, in absolute value, by δ.

We turn to analyzing $E[f(x)g(y)g(z)]$ which is the most challenging term. Let us look at the Fourier expansion

$$g(y) = \sum_\alpha \hat{g}_\alpha \chi_\alpha(y)$$

and divide the terms into four parts forming functions g_i, $1 \leq i \leq 4$. This division is guided by our two parameters s and S and g_1 contains the terms of size at least S and g_2 the terms of size smaller than S but larger than s.

For the small sets β we define a set β to be *shattered* if for any $j_1, j_2 \in \beta$ with $j_1 \neq j_2$, we have that j_1 and j_2 give different answer to the T questions sent to both provers. Note that this in particular implies that $\pi(j_1) \neq \pi(j_2)$. We let g_3 be the small terms that are not shattered and g_4 the small terms that are shattered. Obviously

$$E[f(x)g(y)g(z)] = \sum_{i=1}^{4} E[f(x)g(y)g_i(z)] \tag{4}$$

and we estimate these terms separately.

Let us consider the first term in (4). The function g_1 consists only of terms given by Fourier coefficients of size at least S. Using the definition of the Efron-Stein decomposition given by (1) we see that it contains only terms of size at least S/d. We want to use Lemma 2 with the subdivision $X \times Y$ and Z and we have the following correlation bound, which appears as Lemma 5.3 of [13].

Lemma 6. *[13]* $\rho(X \times Y, Z) \leq (1 - \frac{\delta^2}{d^2 2^{2d+1}})$ *and the same bound applies to* $\rho(X \times Z, Y)$.

From Lemma 2 we conclude that

$$|E[f(x)g(y)g_1(z)]| \leq (1-\frac{\delta^2}{d^2 2^{2d+1}})^{S/d} \|fg\|_2 \|g_1\|_2 \leq (1-\frac{\delta^2}{d^2 2^{2d+1}})^{S/d} \leq e^{-\frac{\delta^2 S}{d^3 2^{2d+1}}}.$$

For the second term and third terms in (4) we have

$$|E[f(x)g(y)g_i(z)]| \leq \|fg\|_2 \|g_i\|_2 \leq \|g_i\|_2,$$

where we will bound these L^2-norms later. For the last term we use

$$E[f(x)g(y)g_4(z)] = \sum_{i=1}^{4} E[f(x)g_i(y)g_4(z)]. \tag{5}$$

Since y and z are symmetric and the only property we used in the previous steps is that $f(x)g(y)$ has L^2-norm bounded by 1, and the same is true for $f(x)g_4(z)$, we can repeat the above argument and we are left to analyze

$$E[f(x)g_4(y)g_4(z)]. \tag{6}$$

We expand the three functions by the Fourier transform and we need to analyze

$$E\left[\sum_{\alpha,\beta,\gamma} \hat{f}_\alpha \hat{g}_\beta \hat{g}_\gamma \chi_\alpha(x)\chi_\beta(y)\chi_\gamma(z)\right].$$

Remember that all β and γ occurring in the sum are of size at most s and are shattered. In fact any sum over β from now on contains only such terms. Moving the expectation inside the sum we first note that for $\beta = \gamma$ and $\pi(\beta) = \alpha$ we have

$$E[\chi_\alpha(x)\chi_\beta(y)\chi_\gamma(z)] = (1 - \delta/d)^{|\beta|}.$$

Most other terms are zero by the following lemma.

Lemma 7. *Unless* $\alpha \subseteq \pi(\beta) \cup \pi(\gamma)$ *and for any element in* i *contained in* $\pi(\beta) \cup \pi(\gamma)$ *but not in* α *we have an element in* $\beta \cap \gamma$ *with* $\pi(j) = i$ *then*

$$E[\chi_\alpha(x)\chi_\beta(y)\chi_\gamma(z)] = 0.$$

Proof. In the first case for $i \in \alpha$ but $i \notin \pi(\beta) \cup \pi(\gamma)$ we have x_i uniform and independent of all other terms. The other claim follows by inspection.

For the terms not covered above we have

Lemma 8. *For any term not covered by Lemma 7 and which does not satisfy $\beta = \gamma$ and $\pi(\beta) = \alpha$ we have*

$$|E[\chi_\alpha(x)\chi_\beta(y)\chi_\gamma(z)]| \leq \delta.$$

Proof. To see this, take and j which is in the symmetric difference of β and γ (or which belong to both β and γ but $\pi(j) \notin \alpha$ if $\beta = \gamma$) fix all values except y_j and z_j (and $x_{\pi(j)}$ in the latter case and remembering that β is shattered). The absolute value of the expectation over these remaining variables is at most δ.

Next we bound

$$\sum_{\alpha,\beta,\gamma} |\hat{f}_\alpha \hat{g}_\beta \hat{g}_\gamma| \tag{7}$$

where we sum over all α, β, γ that give a non-zero value of $E[\chi_\alpha(x)\chi_\beta(y)\chi_\gamma(z)]$.

To help the readers intuition let us point out that any bound, b, on the sum (7) in terms of s and d that allows s to go to infinity while making δb tend to 0 is good enough for us.

We can apply Cauchy-Schwarz to (7) to get the bound

$$\left(\sum_\alpha f_\alpha^2\right)^{1/2} \left(\sum_\alpha \left(\sum_{\beta,\gamma} |\hat{g}_\beta \hat{g}_\gamma|\right)^2\right)^{1/2} \tag{8}$$

where the inner sum over pairs β and γ that could appear together with a given α. In particular for any $i \in \alpha$, at least one of β and γ contains an element that projects onto i and for $i \notin \alpha$ if β contains an element j such that $\pi(j) = i$ then j belongs also to γ .

The first factor of (8) is bounded by one and let us look at the second. Expanding the square in the second factor we get a sum of the form

$$\sum_{\beta,\gamma,\beta'\gamma'} |\hat{g}_\beta \hat{g}_\gamma \hat{g}_{\beta'} \hat{g}_{\gamma'}|, \tag{9}$$

and we claim (and leave to the reader to verify) that each term that appears, is a *projective double cover*. This is defined to mean for any i that appears in $\pi(\beta \cup \gamma \cup \beta' \cup \gamma')$ there are at least two elements in $\beta \cup \gamma \cup \beta' \cup \gamma'$ that project onto this element. Note also that any term appears for at most 2^s different α. The following lemma is essentially from [5] but is proved in the full version of this paper.

Lemma 9. *Suppose $\sum_\beta \hat{g}_\beta^2 = 1$ and each set β occurring is of size at most s. Then the sum (9) taken over distinct projective double covers is at most $(729d^2/2)^s$.*

Summing up we get that $|E[f(x)g(y)g(z)]|$ is bounded by (remember that each term in the sum (9) can appear at most 2^s times),

$$\sum_{\beta} |\hat{f}_{\pi(\beta)}\hat{g}_{\beta}^2|(1-\delta/d)^{|\beta|} + \delta(729d^2)^{s/2} + 2e^{-\frac{\delta^2 S}{d^3 2^{2d+1}}} + 2\|g_2\|_2 + 2\|g_3\|_2. \quad (10)$$

We first take care of the last term.

Lemma 10. *$E[\|g_3\|_2] \leq (s^2T^{-1})^{1/2}$. The expectation is taken over a random question Q_1 that can be asked jointly with Q_2.*

Proof. We have

$$\|g_3\|_2^2 = \sum \hat{g}_{\beta}^2,$$

where the sum is taken over β of that size as most s and which are not shattered by π. For any $j_1, j_2 \in \beta$ such that $j_1 \neq j_2$ we have that the probability that they give the same values to the T questions sent to both provers is at most $\frac{1}{T}$ and as we have less than s^2 such pairs the probability that any individual β is not shattered is bounded by s^2T^{-1}. Since

$$\sum \hat{g}_{\beta}^2 \leq 1$$

the lemma follows.

We proceed to bound $\|g_2\|_2$.

Lemma 11. *$E[\|g_2\|_2] \leq k^{-1/2}$. The expectation is taken over a random value of i in $NTW_{\delta'}^k$.*

Proof. If i is chosen in the protocol then

$$\|g_2\|_2^2 = \sum_{s_{\delta_i} \leq |\beta| \leq S_{\delta_i}} \hat{g}_{\beta}^2.$$

These summation intervals are disjoint and the sum over all β is bounded by one. The lemma follows.

Now by setting the constant c' sufficiently small, the constant c'' sufficiently large, δ' sufficiently small, k sufficiently large and, finally, T sufficiently large, we can conclude that if the verifier in $NTW_{\delta'}^k$ accepts with probability at least $\frac{5}{8} + \epsilon'$ then

$$E\left[\sum_{\beta} |\hat{f}_{\pi(\beta)}\hat{g}_{\beta}^2|(1-\delta_k/d)^{|\beta|}\right] \geq \epsilon',$$

where the sum is over β which are of size at most s_{δ_k} and shattered by π. By an application Cauchy-Schwarz it follows that

$$E\left[\sum_{\beta} \hat{f}_{\pi(\beta)}^2\hat{g}_{\beta}^2(1-\delta_k/d)^{2|\beta|}\right] \geq \epsilon'^2. \quad (11)$$

Now consider the following probabilistic strategy for P_1 and P_2 in the basic two-prover game.

Add the same independent random T copies of q_2 to each of the two questions and look at the corresponding tables f and g in the PCP. Pick sets α and β with probabilities $\hat{f}_\alpha^2$ and $\hat{g}_\beta^2$ respectively. Look at the elements of these sets and what answers they give to the added question. Take the element that defines the lexicographically first value on these added questions and return the answer of this element to the real question. We claim that if $\pi(\beta) = \alpha$ and β is shattered then this strategy succeeds. This follows as each element of β and α give different values to the answers to the added questions and thus choosing the element that gives the lexicographically first value (or any either uniquely defined value) as these values, gives coordinated strategies such that the answers also respect π in the essential coordinate. It follows that the success probability is at least

$$E\left[\sum_\beta \hat{f}^2_{\pi(\beta)}\hat{g}^2_\beta\right], \tag{12}$$

where the sum is over shattered β. Comparing this to the expression (11) we see that this success probability is at least ϵ'^2. This completes the proof of Lemma 5.

5 Conclusions

As the ideas contained in [6] did not seem sufficient to prove our main theorem, it is instructive to see what additional ideas were used. Note that idea of choosing a random bias δ was used in [6] to prove approximation resistance of Max-3Sat on satisfiable instances. Khot later realized (in a, as far as we know, unpublished note) that this complication was not needed by starting the reduction with a smooth instance of label cover.

An important part of the current paper is to combine these two ingredients. While the interaction of these two ideas is not technically difficult the resulting constants are very poor. We are not aware of any other approximation resistance result where the size blow-up as a function of the parameter ϵ is equally dramatic.

Another important part of the current paper is to use the correlated spaces of Mossel and the seemingly simple but very powerful results on what happens to these spaces under products.

We also note that a another key ingredient is the final step where we manage to coordinate the strategies of P_1 and P_2 in the basic two-prover game. In previous similar arguments it has been sufficient to choose a random element in the picked sets α and β. This is not sufficient in the current situation as the resulting acceptance probability would be much smaller than the soundness in the basic two-prover game. Here the smoothness is essential. This idea could be used in many previous arguments in other papers but it is not clear to us that it would result in a significant strengthening of any previous result.

It is an interesting open question to what extent the current methods can be used to eliminate the need of the d-to-1 conjecture in other situations. Huang [8] extended the results of O'Donnell and Wu (also assuming the d-to-1 conjecture), to prove that for any arity $k \geq 4$ the predicate which accepts all odd strings and

one even string is approximation resistant on satisfiable instances. It is likely that our proof could be adopted to this situation but on the other hand Wenner [15] has a proof for this result using other methods (which, however, cannot handle our case, $k = 3$) giving much better constants.

We finally note that we get that Not-Two is useless on satisfiable instances in the sense of [3].

Acknowledgement. I thank Sangxia Huang and Cenny Wenner for discussions relating to this paper and I am also grateful to Oded Goldreich for some comments on the presentation, and to John Wright for pointing out the consequence for the three-to-one conjecture.

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. Austrin, P., Håstad, J.: Randomly supported independence and resistance. SIAM Journal on Computing 40, 1–27 (2011)
3. Austrin, P., Håstad, J.: On the usefulness of predicates. To appear at the Conference for Computational Complexity, 2012 (2012)
4. Goemans, M., Williamson, D.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. Journal of the ACM 42, 1115–1145 (1995)
5. Håstad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. Acta Mathematica 182, 105–142 (1999)
6. Håstad, J.: Some optimal inapproximability results. JACM 48, 798–859 (2001)
7. Håstad, J.: Satisfying Degree-d Equations over $GF[2]^n$. In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) APPROX/RANDOM 2011. LNCS, vol. 6845, pp. 242–253. Springer, Heidelberg (2011)
8. Huang, S.: Approximation resistance on satisfiable instances for predicates strictly dominating parity. ECCC Report 12-040 (2012)
9. Khot, S.: On the power of unique 2-prover 1-round games. In: Proceedings of 34th ACM Symposium on Theory of Computating, pp. 767–775 (2002)
10. Khot, S., Saket, R.: A 3-query non-adaptive pcp with perfect completeness. In: Proc. of 21st Annual Conference on Computational, pp. 159–169. IEEE Computer Society (2006)
11. Khot, S.: Hardness results for coloring 3-colorable 3-uniform hypergraphs. In: Proceedings of 43rd Annual IEEE Symposium of Foundations of Computer Science, pp. 23–32 (2002)
12. Mossel, E.: Gaussian bounds for noise correlation of functions. GAFA 19, 1713–1756 (2010)
13. O'Donnell, R., Wu, Y.: Conditional hardness for satisfiable 3-CSPs. In: Proceedings of 41st ACM Symposium on Theory of Computating, pp. 493–502 (2009)
14. Raz, R.: A parallel repetition theorem. SIAM J. on Computing 27, 763–803 (1998)
15. Wenner, C.: Circumventing D-to-1 for Approximation Resistance of Satisfiable Predicates Strictly Containing Parity of Width Four. In: Gupta, A., et al. (eds.) APPROX/RANDOM 2012. LNCS, vol. 7408, pp. 325–337. Springer, Heidelberg (2012)
16. Zwick, U.: Approximation algorithms for constraint satisfaction problems involving at most three variables per constraint. In: Proceedings 9th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 201–210. ACM (1998)

The Remote Set Problem on Lattices

Ishay Haviv

School of Computer Science
The Academic College of Tel Aviv-Yaffo, Israel

Abstract. We initiate studying the *Remote Set Problem* (RSP) on lattices, which given a lattice asks to find a set of points containing a point which is far from the lattice. We show a polynomial-time deterministic algorithm that on rank n lattice $\mathcal{L}$ outputs a set of points at least one of which is $\sqrt{\log n/n} \cdot \rho(\mathcal{L})$-far from $\mathcal{L}$, where $\rho(\mathcal{L})$ stands for the covering radius of $\mathcal{L}$ (i.e., the maximum possible distance of a point in space from $\mathcal{L}$). As an application, we show that the Covering Radius Problem with approximation factor $\sqrt{n/\log n}$ lies in the complexity class NP, improving a result of Guruswami, Micciancio and Regev by a factor of $\sqrt{\log n}$ (Computational Complexity, 2005).

Our results apply to any ℓ_p norm for $2 \leq p \leq \infty$ with the same approximation factors (except a loss of $\sqrt{\log \log n}$ for $p = \infty$). In addition, we show that the output of our algorithm for RSP contains a point whose ℓ_2 distance from $\mathcal{L}$ is at least $(\log n/n)^{1/p} \cdot \rho^{(p)}(\mathcal{L})$, where $\rho^{(p)}(\mathcal{L})$ is the covering radius of $\mathcal{L}$ measured with respect to the ℓ_p norm. The proof technique involves a theorem on balancing vectors due to Banaszczyk (Random Struct. Alg., 1998) and the 'six standard deviations' theorem of Spencer (Trans. AMS, 1985).

Keywords: Lattices, Covering radius, Remote Set Problem.

1 Introduction

An m-dimensional lattice of rank n is the set of all integer combinations of n linearly independent vectors in $\mathbb{R}^m$ called a basis. Lattices were investigated since the late 18th century by mathematicians, and during the last decades they have also attracted lots of attention from a computational point of view. On one hand, a long line of research shows that many fundamental lattice problems are hard and indicates that it is impossible to solve them in polynomial running time. On the other hand, lattices were shown to be useful as an algorithmic tool as well as applicable in cryptography (see, e.g., [27]). Interestingly, the use of lattices in constructions of cryptographic primitives enjoys strong security relied on the worst-case hardness of certain lattice problems, as was first shown by Ajtai [2]. Therefore, research on algorithms for lattice problems and on their hardness is highly motivated.

There are many important computational problems associated with lattices. The two most fundamental ones are the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP). In the former, for a lattice given by an *arbitrary*

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 182–193, 2012.
© Springer-Verlag Berlin Heidelberg 2012

basis we are supposed to find (the length of) a shortest nonzero vector in the lattice. The problem CVP is an inhomogeneous variant of SVP, in which given a lattice and some target point one has to find (the distance from) the closest lattice point. Another lattice problem of interest is the Covering Radius Problem (CRP) in which given a lattice the goal is to find (a point in space which attains) the maximum possible distance from the lattice. This distance is referred to as the *covering radius* of the lattice. In all problems, the distance is measured relative to some fixed norm on $\mathbb{R}^m$. Usually it is the Euclidean norm ℓ_2 (to which we refer unless otherwise specified) but other ℓ_p norms for $1 \leq p \leq \infty$ are of interest as well (see, e.g., [31]). We note that all the mentioned problems have analogous intensively studied problems in the context of linear codes.

The first polynomial-time approximation algorithm for SVP was presented by Lenstra, Lenstra and Lovász (LLL) in 1982 and achieved an approximation factor of $2^{O(n)}$, where n is the rank of the lattice [22]. Using their algorithm, Babai came up with the nearest plane algorithm achieving the same approximation factor for CVP [6]. A few years later, Schnorr obtained a slightly sub-exponential approximation factor for SVP, namely $2^{O(n(\log\log n)^2/\log n)}$ [32], and this has since been improved by a randomized algorithm of [3]. Kannan presented deterministic algorithms solving SVP and CVP *exactly* requiring running time $n^{O(n)}$ [20], and this was improved to $2^{O(n)}$ more than two decades later by Micciancio and Voulgaris [28]. The algorithm of [28] was recently extended to any ℓ_p norm (and other norms) by Dadush, Peikert and Vempala [12].

On the hardness side, it is known that CVP is NP-hard to approximate to within $n^{c/\log\log n}$ [14] for some constant $c > 0$ and that (under randomized reductions) it is NP-hard to approximate SVP to within any constant [21]. Hardness of approximating SVP to within some $n^{c/\log\log n}$ factor is known to date only assuming some stronger (yet plausible) complexity assumptions [19,25] (see [13] for stronger results for the ℓ_∞ norm). In contrast to the hardness results, there is a line of research showing limits on the hardness of lattice problems. For example, suitably defined gap versions of both SVP and CVP are known to lie in coNP for approximation factor of $\sqrt{n}$ [1] and in coAM for approximation factor of $\sqrt{n/\log n}$ [15]. Therefore, they are unlikely to be NP-hard to approximate to within $\sqrt{n/\log n}$, as this would imply the collapse of the polynomial-time hierarchy [11]. The results of [1] were extended by Peikert to SVP and CVP in the ℓ_p norm for $2 \leq p \leq \infty$ with essentially the same approximation factors [30].

The study of the Covering Radius Problem on lattices (CRP) from a computational point of view was initiated by Guruswami, Micciancio and Regev in [16]. Previously this problem was used by Micciancio to get tighter connections between the average-case and worst-case complexity of lattice problems [23]. It was shown in [16] that approximating CRP to within $\gamma(n)$ can be done in exponential time $2^{O(n)}$ for any constant $\gamma(n) > 1$ and in polynomial time for some $\gamma(n) = 2^{O(n\log\log n/\log n)}$.[1] In addition, they showed that CRP is in AM

[1] To be precise, the algorithms of [16] were randomized since they used randomized algorithms of [4]. However, the deterministic algorithm of [28] implies that the approximation obtained in [16] can be achieved deterministically.

for $\gamma(n) = 2$, in coAM for $\gamma(n) = \sqrt{n/\log n}$, and in NP ∩ coNP for $\gamma(n) = \sqrt{n}$. Peikert showed in [30] that CRP in the ℓ_p norm for $2 \leq p \leq \infty$ lies in coNP for the same $\sqrt{n}$ approximation factor (except a loss of $\sqrt{\log n}$ for $p = \infty$). However, such an extension to ℓ_p norms is not known for NP and this was left as an open question in [16]. On the hardness side, very little is known. The decisional gap version of CRP (of deciding whether the covering radius is at most some given r) naturally lies in the complexity class Π_2 and is conjectured to be Π_2-hard [23]. However, Π_2-hardness is only known for CRP in the ℓ_p norm for any sufficiently large value of p [18].

Among the results mentioned above regarding CRP, the one saying that CRP is in AM for $\gamma(n) = 2$ is unique for this lattice problem. The proof of this fact is relatively simple, and follows from the following AM protocol. Given a lattice $\mathcal{L}$ and a number r, the verifier sends to the prover a uniformly chosen random point in space and the prover has to provide a lattice point whose distance from the random point is at most r. Clearly, if the covering radius is at most r then the prover can act in a way that the verifier accepts with probability 1. On the other hand, the soundness is crucially based on an observation of [16] that random points in space are far from the lattice with high probability. More precisely, a uniformly chosen random point is with constant probability at least $\frac{1}{2} \cdot \rho(\mathcal{L})$-far from a lattice $\mathcal{L}$, where $\rho(\mathcal{L})$ stands for the covering radius of $\mathcal{L}$.

A natural question to ask is whether CRP with $\gamma(n) = 2$ (or with some other factor smaller than $\sqrt{n}$) can be shown to be in NP. Observe that if the verifier could *deterministically* pick a point in space which is quite far from the input lattice, then the protocol above could yield an NP verifier for CRP. Moreover, it can be seen that a deterministic algorithm which outputs polynomially many points at least one of which is quite far from the lattice could suffice for this purpose as well. This challenge is the driving force of the current work, in which we study deterministic polynomial-time algorithms which given a lattice find a set of points containing a point which is far from the lattice.

1.1 Our Contribution

In this paper we initiate studying the Remote Set Problem (RSP) on lattices. This problem can be viewed as a *generalized search* variant of the Covering Radius Problem studied in [23,16,18,17]. In RSP the input is a rank n lattice given by a basis generating it. The goal is to find a set S of points in the span of B containing a point which is far from the lattice. This problem is analogous to a problem suggested for study by Alon, Panigrahy and Yekhanin in the context of linear codes [5].

Recall that the maximum possible distance of a point in space from a lattice $\mathcal{L}$ is called the covering radius of $\mathcal{L}$ and is denoted by $\rho(\mathcal{L})$. The quality of an algorithm for RSP depends on two parameters (to be minimized):

1. the *size* d of the set S constructed by the algorithm, and
2. the *remoteness parameter* which is defined as the minimum $\gamma \geq 1$ for which S contains a point whose distance from $\mathcal{L}$ is at least $\frac{1}{\gamma} \cdot \rho(\mathcal{L})$ for every input lattice $\mathcal{L}$.

As was mentioned before, for every lattice $\mathcal{L}$ a uniformly chosen random point in space has distance at least $\frac{1}{2} \cdot \rho(\mathcal{L})$ from $\mathcal{L}$ with a constant probability [16]. This implies that the efficient algorithm which uniformly and randomly picks a point in space (without even looking at the specific input) solves RSP with $d = 1$ and $\gamma = 2$ with a constant probability of success. Moreover, an algorithm that independently and randomly picks d points and outputs all of them solves RSP with parameters d and $\gamma = 2$ with failure probability which tends to 0 exponentially in d. However, the problem seems much more challenging if we require the algorithm to be *deterministic* (this is also the case for linear codes; see [5] for details).

To obtain a deterministic algorithm for RSP one can use an observation made in [7,16] saying that for every lattice $\mathcal{L}$ there exists a point in $\frac{1}{2} \cdot \mathcal{L}$ whose distance from $\mathcal{L}$ is at least $\frac{1}{2} \cdot \rho(\mathcal{L})$. This implies that the algorithm, which outputs all the linear combinations of the basis vectors with all coefficients in $\{0, \frac{1}{2}\}$, deterministically solves RSP with $\gamma = 2$. However, the number of points that this algorithm outputs is $d = 2^n$, where n is the rank of the input lattice, and, in particular, its running time is exponential in n.

In this paper we consider the task of finding an algorithm for RSP which is simultaneously deterministic and of polynomial running time. First, we observe that the LLL algorithm [22] can be used to deterministically and efficiently calculate a point whose distance from the lattice approximates the covering radius with an exponential factor.

Theorem 1. *There exists a deterministic polynomial-time algorithm for* RSP *with* $d = 1$ *and* $\gamma(n) = 2^{O(n)}$.

Our main result significantly improves the remoteness parameter γ achieved in Theorem 1 at the price of having d polynomial in the input size, as stated below.

Theorem 2. *There exists a deterministic polynomial-time algorithm for* RSP *with* $\gamma(n) = \sqrt{n/\log n}$.

Notice that the number d of points that the algorithm of Theorem 2 outputs is polynomial in the input size, as d clearly cannot be higher than the running time.

As alluded to before, besides being a natural lattice problem, studying RSP is motivated by research on the Covering Radius Problem (CRP). In the promise version of CRP with parameter $\gamma \geq 1$ the input consists of a lattice $\mathcal{L}$ and a number r, and the goal is to decide whether the covering radius $\rho(\mathcal{L})$ of $\mathcal{L}$ is at most r or larger than $\gamma \cdot r$. This problem lies in the complexity class Π_2 (for any γ), since $\rho(\mathcal{L}(B)) \leq r$ if and only if *for all* x in the span of $\mathcal{L}$ there *exists* $y \in \mathcal{L}$ such that the distance between x and y is at most r. For small values of γ the problem is conjectured to be Π_2-hard [23], however it is known that for $\gamma(n) = \sqrt{n}$ it lies in NP [16] (see also [26, Section 7]). In order to prove that CRP with certain $\gamma = \gamma(n)$ is in NP one should come up with an efficiently verifiable witness for instances with $\rho(\mathcal{L}(B)) \leq r$ which does not exist if $\rho(\mathcal{L}(B)) > \gamma \cdot r$. We claim that a deterministic and efficient algorithm for RSP can be useful for

this purpose. Indeed, such an algorithm outputs a set S of points at least one of which is quite far from the lattice, hence in order to verify that the covering radius is small it suffices to verify that the points in S are close to the lattice. This can be easily done taking the witness which consists of the lattice points closest to the points in S. We combine this idea with Theorem 2 and obtain the following theorem which improves upon the $\sqrt{n}$ factor obtained in [16].

Theorem 3. CRP *with approximation factor* $\sqrt{n/\log n}$ *is in* NP.

Another motivation to study RSP comes from the connections between CRP and the Closest Vector Problem (CVP). Known connections between these problems were found useful in several results of [16] regarding the complexity of CRP, namely, the exponential-time approximation algorithm for any $\gamma > 1$ and the proof systems implying that CRP with approximation factors $\sqrt{n}$ and $\sqrt{n/\log n}$ are in coNP and coAM respectively. It turns out that algorithms for RSP imply reductions from CRP to CVP. Specifically, we show that our algorithm for RSP from Theorem 2 implies a deterministic rank-preserving polynomial-time Cook reduction from CRP to CVP with $\sqrt{n/\log n}$ loss in the approximation factor. The only similar result we are aware of is implied by a paper of Micciancio [24] and gives a $\sqrt{n}$ loss in the approximation factor.[2] We also show that Karp reductions from CRP to CVP can be derived from algorithms for RSP. For details see the full version of the paper.

In the above discussion RSP and CRP were considered with respect to the Euclidean norm, but it is natural to consider them with respect to any other ℓ_p norm for $1 \leq p \leq \infty$. It is easy to prove that our results can be adapted to arbitrary ℓ_p norm, since in $\mathbb{R}^m$ all ℓ_p norms are within $\sqrt{m}$ from the ℓ_2 one. However, this introduces a $\sqrt{m}$ loss in the approximation factors (where m is the dimension of the lattice). We actually show that this loss is not necessary answering a question asked in [16]. We prove that Theorem 2 holds for any ℓ_p norm for $2 \leq p < \infty$. Namely, for every $2 \leq p < \infty$, there exists a deterministic polynomial-time algorithm that given a lattice whose covering radius with respect to the ℓ_p norm is r outputs a set of points guaranteed to contain a point whose ℓ_p distance from the lattice is at least $\sqrt{\log n/n} \cdot r$. Interestingly, we show that our algorithm can also be generalized to the ℓ_p norm in the following manner: given a lattice whose covering radius with respect to the ℓ_p norm is r it outputs a set of points guaranteed to contain a point whose ℓ_2 distance from the lattice is at least $(\log n/n)^{1/p} \cdot r$. Our results are similarly extended to the ℓ_∞ norm and imply a generalization of Theorem 3 to every ℓ_p norm for $2 \leq p \leq \infty$.

Open Questions. Our work raises several open questions. It will be interesting to understand for which parameters the Remote Set Problem can be deterministically solved in polynomial time. We have shown that there exists a deterministic

[2] Strictly speaking, Micciancio shows in [24] a gap-preserving Cook reduction from the Shortest Independent Vectors Problem (SIVP) to CVP which, combined with known relations between SIVP and CRP, implies a Cook reduction from CRP to CVP with $\sqrt{n}$ loss in the approximation factor.

polynomial-time algorithm that given a lattice outputs a set of points one of which has distance at least $\sqrt{\log n/n}$ times the covering radius. Can the guarantee on the distance be improved? Can it be improved to the factor $1/2$ for which this can be achieved by a randomized algorithm [16]? Can one achieve the $\sqrt{\log n/n}$ factor (or even a $1/\sqrt{n}$ factor) by an algorithm which outputs only one point instead of polynomially many points? Can it be achieved for the ℓ_p norm for $1 \leq p < 2$?

Outline. The paper is organized as follows. In Section 2 we gather all the definitions on lattices and computational lattice problems that we need in the paper. In Section 3 we present our algorithm for RSP proving Theorem 2 and some extensions of it. In Section 4 we present an application of RSP to CRP implying Theorem 3. The proof of Theorem 1 and additional applications of RSP can be found in the full version of the paper.

2 Preliminaries

Notations. For $1 \leq p < \infty$ the ℓ_p norm of a vector $x \in \mathbb{R}^m$ is defined as $\|x\|_p = (\sum_{i=1}^m |x_i|^p)^{1/p}$ and for $p = \infty$ it is defined as $\|x\|_\infty = \max_{1 \leq i \leq m} |x_i|$. The ℓ_p distance between two vectors $x, y \in \mathbb{R}^m$ is defined as $\mathrm{dist}_p(x, y) = \|x - y\|_p$. The ℓ_p distance of a vector $x \in \mathbb{R}^m$ from a set $S \subseteq \mathbb{R}^m$ is defined as $\mathrm{dist}_p(x, S) = \min_{y \in S} \mathrm{dist}_p(x, y)$. We say that x is *r-far* from S if $\mathrm{dist}_p(x, S) \geq r$. When we omit the subscript p (or a superscript (p)) we refer to the the Euclidean norm ℓ_2.

Lattices. A *lattice* is a discrete additive subgroup of $\mathbb{R}^m$. Equivalently, it is the set of all integer combinations $\mathcal{L}(b_1, \ldots, b_n) = \{\sum_{i=1}^n x_i b_i : x_i \in \mathbb{Z} \text{ for all } 1 \leq i \leq n\}$ of n linearly independent vectors $b_1, \ldots, b_n$ in $\mathbb{R}^m$ $(n \leq m)$. If the lattice *rank* n equals its *dimension* m we say that the lattice is *full-rank*. The set $(b_1, \ldots, b_n)$ is called a *basis* of the lattice. Note that a lattice has many possible bases. We often represent a basis by an m by n matrix B having the basis vectors as columns, and we say that the basis B *generates* the lattice $\mathcal{L}$. In such case we write $\mathcal{L} = \mathcal{L}(B)$. The linear space spanned by B is denoted $\mathrm{span}(B) = \{\sum_{i=1}^n x_i b_i : x_i \in \mathbb{R} \text{ for all } 1 \leq i \leq n\}$. A *sublattice* of $\mathcal{L}$ is a lattice $\mathcal{L}(S) \subseteq \mathcal{L}$ generated by some linearly independent lattice vectors $S \subseteq \mathcal{L}$.

A basic parameter associated with lattices is the covering radius. For a lattice basis $B = (b_1, \ldots, b_n)$ the *covering radius* of $\mathcal{L}(B)$ with respect to the ℓ_p norm is defined as $\rho^{(p)}(\mathcal{L}(B)) = \max_{x \in \mathrm{span}(B)} \mathrm{dist}_p(x, \mathcal{L}(B))$. Hence, $\rho^{(p)}(\mathcal{L}(B)) \leq r$ means that for any $x \in \mathrm{span}(B)$ there exists a lattice point $y \in \mathcal{L}(B)$ such that $\mathrm{dist}_p(x, y) \leq r$. Conversely, $\rho^{(p)}(\mathcal{L}(B)) > r$ means that there exists some $x \in \mathrm{span}(B)$ such that any lattice point $y \in \mathcal{L}(B)$ satisfies $\mathrm{dist}_p(x, y) > r$. A *deep hole* of $\mathcal{L}(B)$ is a point $x \in \mathrm{span}(B)$ at distance $\mathrm{dist}_p(x, \mathcal{L}(B)) = \rho^{(p)}(\mathcal{L}(B))$ from the lattice.

The following lemma shows that in order to find a point quite far from a lattice $\mathcal{L}(B)$ it suffices to consider linear combinations of vectors in B with coefficients

in $\{0, \frac{1}{2}\}$. This lemma (in more general forms) was proved in [7,16], and we repeat its proof here for completeness.

Lemma 1. *For every $1 \le p \le \infty$ and any lattice basis $B = (b_1, \ldots, b_n)$ there exists a vector*

$$v = a_1 \cdot b_1 + \ldots + a_n \cdot b_n$$

with $a_j \in \{0, \frac{1}{2}\}$ for all $1 \le j \le n$ such that $\mathrm{dist}_p(v, \mathcal{L}(B)) \ge \frac{1}{2} \cdot \rho^{(p)}(\mathcal{L}(B))$.

Proof. Let w be a deep hole of the lattice $\mathcal{L}(B)$ with respect to the ℓ_p norm. Consider the point $2w$ and observe that, like any point in $\mathrm{span}(B)$, its ℓ_p distance from $\mathcal{L}(B)$ is at most $\rho^{(p)}(\mathcal{L}(B))$. This means that there exists a lattice point $u \in \mathcal{L}(B)$ such that $\mathrm{dist}_p(u, 2w) \le \rho^{(p)}(\mathcal{L}(B))$ and hence $\mathrm{dist}_p(\frac{1}{2} \cdot u, w) \le \frac{1}{2} \cdot \rho^{(p)}(\mathcal{L}(B))$. Now, by triangle inequality,

$$\mathrm{dist}_p(\frac{1}{2} \cdot u, \mathcal{L}(B)) \ge \mathrm{dist}_p(w, \mathcal{L}(B)) - \mathrm{dist}_p(\frac{1}{2} \cdot u, w) \ge \frac{1}{2} \cdot \rho^{(p)}(\mathcal{L}(B)).$$

Finally, observe that $\frac{1}{2} \cdot u \in \frac{1}{2} \cdot \mathcal{L}(B)$, so by reducing modulo 1 its coefficients as a linear combination of B, we obtain a vector of the required form with the same ℓ_p distance from $\mathcal{L}(B)$.

Computational Lattice Problems. For any $1 \le p \le \infty$ and any approximation factor $\gamma \ge 1$ (which is usually considered as a function of the lattice rank n) we define the following computational problems.

Definition 1 (Covering Radius Problem). *An instance of $\mathsf{GapCRP}_{\gamma}^{(p)}$ is a pair (B, r) where $B \in \mathbb{Q}^{m \times n}$ is a rank n lattice basis and $r \in \mathbb{Q}$ is a rational number. In YES instances $\rho^{(p)}(\mathcal{L}(B)) \le r$ and in NO instances $\rho^{(p)}(\mathcal{L}(B)) > \gamma \cdot r$.*

Definition 2 (Remote Set Problem). *An instance of $\mathsf{RSP}_{d,\gamma}^{(p)}$ is a rank n lattice basis $B \in \mathbb{Q}^{m \times n}$. The goal is to find a set $S \subseteq \mathrm{span}(B)$ of size $|S| \le d$ containing a point v such that $\mathrm{dist}_p(v, \mathcal{L}(B)) \ge \frac{1}{\gamma} \cdot \rho^{(p)}(\mathcal{L}(B))$.*

Balancing Vectors. The analysis of the main algorithm presented in this work relies on upper bounds on the length of linear combinations with ± 1 coefficients of a given set of vectors. In the following we provide the needed background.

In Banach spaces theory, a normed space X is said to have *type* 2 if there exists a constant $T < \infty$ such that for every n and $x_1, \ldots, x_n \in X$,

$$\Big(\mathbb{E} \, \| \sum_{i=1}^{n} \varepsilon_i \cdot x_i \|_X^2\Big)^{1/2} \le T \cdot \big(\sum_{i=1}^{n} \|x_i\|_X^2\big)^{1/2}, \tag{1}$$

where the expectation is over a uniform choice of signs $\varepsilon_1, \ldots, \varepsilon_n \in \{-1, +1\}$. For example, it is easy to see that the Euclidean space ℓ_2 has type 2, since for ℓ_2 equality holds in (1) with $T = 1$ as follows from the parallelogram law. It is well-known that for every $2 \le p < \infty$ the ℓ_p normed space has type 2 with $T = c \cdot \sqrt{p}$ for some absolute constant $c > 0$ (see, e.g., [29]). In particular, for every n vectors $x_1, \ldots, x_n$ there exists some choice of signs for which the corresponding linear combination has ℓ_p norm at most $O(\sqrt{n})$ times the maximum ℓ_p norm of the x_i's. This is stated in the following lemma.

Lemma 2. *For every $2 \leq p < \infty$ there exists a constant $c_p > 0$ for which the following holds. For every n vectors $x_1, \ldots, x_n \in \mathbb{R}^m$ there exist $\varepsilon_1, \ldots, \varepsilon_n \in \{-1, +1\}$ such that $\| \sum_{i=1}^n \varepsilon_i \cdot x_i \|_p \leq c_p \cdot \sqrt{n} \cdot \max_{1 \leq i \leq n} \|x_i\|_p$.*

A similar statement, motivated by questions on set systems in combinatorial discrepancy, is known for the ℓ_∞ norm. By a simple probabilistic argument it can be seen that every set of n vectors in $\mathbb{R}^m$ has a linear combination with ± 1 coefficients whose ℓ_∞ norm is at most $O(\sqrt{n \log m})$ times the maximum ℓ_∞ norm of the vectors. Interestingly, Spencer showed in 1985 that this can be improved to $O(\sqrt{n \log (2m/n)})$ [33]. For the special case of $m = n$ he showed a bound of $6\sqrt{n}$, commonly referred to as the 'six standard deviations' theorem. In a recent breakthrough, Bansal [9] gave algorithmic results related to Spencer's bound.

Theorem 4 ([33]). *There exists a constant $c_\infty > 0$ such that for every n vectors $x_1, \ldots, x_n \in \mathbb{R}^m$ $(m \geq n)$ there exist $\varepsilon_1, \ldots, \varepsilon_n \in \{-1, +1\}$ such that $\| \sum_{i=1}^n \varepsilon_i \cdot x_i \|_\infty \leq c_\infty \cdot \sqrt{n \cdot \log (2m/n)} \cdot \max_{1 \leq i \leq n} \|x_i\|_\infty$.*

3 Algorithms for the Remote Set Problem

In this section we present our deterministic polynomial-time algorithm for RSP, namely we prove Theorem 2 and some extensions. The proof of Theorem 1 can be found in the full version of the paper.

3.1 Proof of Theorem 2

We start with the following statement from which we derive Theorem 2.

Theorem 5. *For every $2 \leq p < \infty$ and every $k = k(n) \geq 1$ there exists a deterministic $2^k \cdot s^{O(1)}$ time algorithm for $\mathsf{RSP}_{d,\gamma}^{(p)}$ with $d(n) = O(\frac{n}{k} \cdot 2^k)$ and $\gamma(n) = O(\sqrt{\frac{n}{k}})$, where n denotes the lattice rank and s denotes the input size. The same holds for $p = \infty$ with $\gamma(n, m) = O(\sqrt{\frac{n}{k} \cdot \log (2mk/n)})$, where m denotes the lattice dimension.*

Proof. Assume for simplicity that $k = k(n)$ divides n. We consider the algorithm that given a lattice basis $B = (b_1, \ldots, b_n)$ first partitions its vectors into $\frac{n}{k}$ sets of size k each. Then the algorithm outputs all vectors in space which form a linear combination with all coefficients in $\{0, \frac{1}{2}\}$ of vectors in one of these sets. More precisely, for every $1 \leq i \leq \frac{n}{k}$ let S_i be the set of all vectors of the form

$$a_1 \cdot b_{(i-1)k+1} + \ldots + a_k \cdot b_{ik}$$

where $a_j \in \{0, \frac{1}{2}\}$ for all j. Our algorithm outputs the union $S = \cup_{i=1}^{n/k} S_i$ (see Figure 1). Observe that $|S| \leq \frac{n}{k} \cdot 2^k$ and that S can be constructed in time $2^k \cdot s^{O(1)}$ where s is the input size.

Fix some $2 \leq p < \infty$. We claim that there exists a vector in S whose ℓ_p distance from $\mathcal{L}(B)$ is at least $\frac{1}{2 \cdot c_p} \cdot \sqrt{\frac{k}{n}} \cdot \rho^{(p)}(\mathcal{L}(B))$, where $c_p > 0$ is the constant

Remote Set Problem(B)
Input: A lattice basis $B = (b_1, \ldots, b_n) \in \mathbb{Q}^{m \times n}$.
Output: A set S of $\frac{n}{k} \cdot 2^k$ vectors in span(B) at least one of which is far from $\mathcal{L}(B)$.

- For every $1 \leq i \leq \frac{n}{k}$,
  1. Define $B_i = (b_{(i-1)k+1}, \ldots, b_{ik})$.
  2. Construct the set S_i of all vectors that form a linear combination with all coefficients in $\{0, \frac{1}{2}\}$ of the vectors in B_i.
- Output $S = \cup_{i=1}^{n/k} S_i$.

Fig. 1. An Algorithm for the Remote Set Problem

from Lemma 2 which depends solely on p. Assume for contradiction that this is not the case. By Lemma 1, there exists a vector $v = a_1 \cdot b_1 + \ldots + a_n \cdot b_n$ with $a_j \in \{0, \frac{1}{2}\}$ for all $1 \leq j \leq n$ such that $\mathrm{dist}_p(v, \mathcal{L}(B)) \geq \frac{1}{2} \cdot \rho^{(p)}(\mathcal{L}(B))$. Write $v = \frac{1}{2}(v_1 + \ldots + v_{n/k})$ where for every $1 \leq i \leq \frac{n}{k}$, $v_i = 2 \cdot (a_{(i-1)k+1} \cdot b_{(i-1)k+1} + \ldots + a_{ik} \cdot b_{ik})$. Since $\frac{1}{2} \cdot v_i \in S$ our assumption implies that there exists a lattice vector $u_i \in \mathcal{L}(B)$ such that

$$\|\frac{1}{2} \cdot v_i - u_i\|_p < \frac{1}{2 \cdot c_p} \cdot \sqrt{\frac{k}{n}} \cdot \rho^{(p)}(\mathcal{L}(B)). \tag{2}$$

For every $1 \leq i \leq \frac{n}{k}$, denote $s_i = \frac{1}{2} \cdot v_i - u_i$, and apply Lemma 2 to obtain $\varepsilon_1, \ldots, \varepsilon_{n/k} \in \{-1, +1\}$ such that $\| \sum_{i=1}^{n/k} \varepsilon_i \cdot s_i \|_p \leq c_p \cdot \sqrt{\frac{n}{k}} \cdot \max_{1 \leq i \leq n/k} \|s_i\|_p < c_p \cdot \sqrt{\frac{n}{k}} \cdot \frac{1}{2 \cdot c_p} \cdot \sqrt{\frac{k}{n}} \cdot \rho^{(p)}(\mathcal{L}(B)) = \frac{1}{2} \cdot \rho^{(p)}(\mathcal{L}(B))$, as follows from (2). Finally, observe that the difference between v and $\sum_{i=1}^{n/k} \varepsilon_i \cdot s_i$ is a lattice vector, hence

$$\mathrm{dist}_p(v, \mathcal{L}(B)) = \mathrm{dist}_p(\sum_{i=1}^{n/k} \varepsilon_i \cdot s_i, \mathcal{L}(B)) \leq \| \sum_{i=1}^{n/k} \varepsilon_i \cdot s_i \|_p < \frac{1}{2} \cdot \rho^{(p)}(\mathcal{L}(B)),$$

in contradiction to our choice of v.

The analysis for $p = \infty$ is almost identical to the analysis described above. The only difference is in applying Spencer's theorem (Theorem 4) instead of Lemma 2 to find a short ± 1 combination of the s_i's.

Notice that in the ℓ_∞ case the remoteness parameter γ obtained in Theorem 5 does not depend only on the rank n but also on the dimension m. Hence, let us state it again for the special case of full-rank lattices (i.e., $m = n$) which is usually considered.

Theorem 6. *For every $k = k(n) \geq 1$ there exists a deterministic $2^k \cdot s^{O(1)}$ time algorithm for $\mathsf{RSP}_{d,\gamma}^{(\infty)}$ on full-rank lattices with $d(n) = O(\frac{n}{k} \cdot 2^k)$ and $\gamma(n) = O(\sqrt{\frac{n \cdot \log(2k)}{k}})$, where s denotes the input size.*

Now Theorem 2 is easily derived from Theorem 5 by choosing $k = c \log n$ where n is the lattice rank and c is a constant, as stated in the following corollary. We

note that one can obtain a slightly stronger version of this corollary by choosing $k = O(\log s)$ where s is the input size.

Corollary 1. *For every $2 \leq p < \infty$ and every constant $c \geq 1$, there exists a deterministic polynomial-time algorithm for $\mathsf{RSP}_{d,\gamma}^{(p)}$ with $d(n) = n^{O(c)}$ and $\gamma(n) = O(\sqrt{n/(c \log n)})$. In addition, for every constant $c \geq 1$, there exists a deterministic polynomial-time algorithm for $\mathsf{RSP}_{d,\gamma}^{(\infty)}$ on full-rank lattices with $d(n) = n^{O(c)}$ and $\gamma(n) = O(\sqrt{n \cdot \log\log n/(c \log n)})$.*

3.2 Extensions of Theorem 2

In the analysis of our algorithm for RSP we applied Lemma 2 and Theorem 4 which roughly speaking say that every set of vectors has a linear combination with ± 1 coefficients of small ℓ_p norm compared to the maximum ℓ_p norm of the vectors in the set. It turns out that similar questions were studied where the goal is to minimize the ℓ_p norm of the linear combination compared to the maximum ℓ_2 norm of the vectors in the set. This is stated in the following theorem which stems from a paper of Banaszczyk [8] (see also [10, Propositions 24, 25]).

Theorem 7 ([8]). *For every $2 \leq p \leq \infty$ there exists a constant $c_p > 0$ for which the following holds. For every n vectors $x_1, \ldots, x_n \in \mathbb{R}^m$ there exist $\varepsilon_1, \ldots, \varepsilon_n \in \{-1, +1\}$ such that for $2 \leq p < \infty$, $\| \sum_{i=1}^n \varepsilon_i \cdot x_i \|_p \leq c_p \cdot n^{1/p} \cdot \max_{1 \leq i \leq n} \|x_i\|_2$, and for $p = \infty$, $\| \sum_{i=1}^n \varepsilon_i \cdot x_i \|_\infty \leq c_\infty \cdot \sqrt{1 + \log n} \cdot \max_{1 \leq i \leq n} \|x_i\|_2$.*

For $p = \infty$, a famous conjecture of Komlós asserts the following.

Conjecture 1. [Komlós Conjecture] There exists a constant $c > 0$ such that for every n vectors $x_1, \ldots, x_n \in \mathbb{R}^m$ there exist $\varepsilon_1, \ldots, \varepsilon_n \in \{-1, +1\}$ such that $\| \sum_{i=1}^n \varepsilon_i \cdot x_i \|_\infty \leq c \cdot \max_{1 \leq i \leq n} \|x_i\|_2$.

Now we observe that Theorem 7 can be used to prove an additional property of the output of our algorithm for RSP. The use of Lemma 2 and Theorem 4 in the proof implied that at least one of points in the output has large ℓ_p distance from the lattice compared to the covering radius in the ℓ_p norm. However, applying Theorem 7 in the proof yields that at least one of the vectors has large ℓ_2 distance from the lattice, still compared to the covering radius in the ℓ_p norm. Theorems 8 and 9 below follow from the algorithm presented in the proof of Theorem 5 (see Figure 1) for $k = \Theta(\log n)$ and $k = 1$ respectively. We omit the proof details.

Theorem 8. *For every $2 \leq p < \infty$ there exists a constant $c_p > 0$ for which the following holds. For every $c \geq 1$ there exists a deterministic polynomial-time algorithm that given a rank n lattice $\mathcal{L}$ outputs a set of $n^{O(c)}$ points at least one of which has ℓ_2 distance at least $c_p \cdot (\frac{c \log n}{n})^{1/p} \cdot \rho^{(p)}(\mathcal{L})$ from $\mathcal{L}$.*

Theorem 9. *There exists a constant $c > 0$ and a deterministic polynomial-time algorithm that given a rank n lattice $\mathcal{L}$ outputs a set of n points at least one of which has ℓ_2 distance at least $\frac{c}{\sqrt{1+\log n}} \cdot \rho^{(\infty)}(\mathcal{L})$ from $\mathcal{L}$. Assuming Conjecture 1, one of the points has ℓ_2 distance at least $c \cdot \rho^{(\infty)}(\mathcal{L})$ from $\mathcal{L}$.*

4 On the Complexity of the Covering Radius Problem

The following simple lemma, whose proof can be found in the full version of the paper, relates RSP to proving that CRP with certain approximation factors is in NP. The theorem that follows it is an immediate consequence of the lemma and Corollary 1 confirming Theorem 3.

Lemma 3. *For every* $1 \leq p \leq \infty$, $d = d(n)$ *and* $\gamma = \gamma(n)$, *if there exists a deterministic polynomial-time algorithm for* $\mathsf{RSP}^{(p)}_{d,\gamma}$ *then* $\mathsf{GapCRP}^{(p)}_{\gamma}$ *is in* NP.

Theorem 10. *For every* $2 \leq p < \infty$ *and every constant* $c \geq 1$, $\mathsf{GapCRP}^{(p)}_{\gamma}$ *is in* NP *for* $\gamma(n) = \sqrt{n/(c \log n)}$. *In addition, for every constant* $c \geq 1$, $\mathsf{GapCRP}^{(\infty)}_{\gamma}$ *on full-rank lattices is in* NP *for* $\gamma(n) = \sqrt{n \log\log n/(c \log n)}$.

Acknowledgement. We would like to deeply thank Oded Regev for valuable and fruitful discussions.

References

1. Aharonov, D., Regev, O.: Lattice problems in NP intersect coNP. Journal of the ACM 52(5), 749–765 (2005); Preliminary version in FOCS 2004
2. Ajtai, M.: Generating hard instances of lattice problems. In: Complexity of Computations and Proofs. Quad. Mat., vol. 13, pp. 1–32. Dept. Math., Seconda Univ. Napoli, Caserta (2004)
3. Ajtai, M., Kumar, R., Sivakumar, D.: A sieve algorithm for the shortest lattice vector problem. In: Proc. 33rd ACM Symp. on Theory of Computing (STOC), pp. 601–610 (2001)
4. Ajtai, M., Kumar, R., Sivakumar, D.: Sampling short lattice vectors and the closest lattice vector problem. In: Proc. of 17th IEEE Annual Conference on Computational Complexity (CCC), pp. 53–57 (2002)
5. Alon, N., Panigrahy, R., Yekhanin, S.: Deterministic Approximation Algorithms for the Nearest Codeword Problem. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 339–351. Springer, Heidelberg (2009)
6. Babai, L.: On Lovász lattice reduction and the nearest lattice point problem. Combinatorica 6(1), 1–13 (1986)
7. Banaszczyk, W.: Balancing vectors and convex bodies. Studia Math. 106(1), 93–100 (1993)
8. Banaszczyk, W.: Balancing vectors and gaussian measures of n-dimensional convex bodies. Random Struct. Algorithms 12(4), 351–360 (1998)
9. Bansal, N.: Constructive algorithms for discrepancy minimization. In: FOCS, pp. 3–10 (2010)
10. Barthe, F., Guédon, O., Mendelson, S., Naor, A.: A probabilistic approach to the geometry of the ℓ_p^n-ball. The Annals of Probability 33(2), 480–513 (2005)
11. Boppana, R., Håstad, J., Zachos, S.: Does co-NP have short interactive proofs? Information Processing Letters 25, 127–132 (1987)
12. Dadush, D., Peikert, C., Vempala, S.: Enumerative lattice algorithms in any norm via M-ellipsoid coverings. In: FOCS, pp. 580–589 (2011)

13. Dinur, I.: Approximating SVP_∞ to within almost-polynomial factors is NP-hard. Theoretical Computer Science 285(1), 55–71 (2002)
14. Dinur, I., Kindler, G., Raz, R., Safra, S.: Approximating CVP to within almost-polynomial factors is NP-hard. Combinatorica 23(2), 205–243 (2003); Preliminary version in FOCS 1998
15. Goldreich, O., Goldwasser, S.: On the limits of nonapproximability of lattice problems. J. Comput. System Sci. 60(3), 540–563 (2000)
16. Guruswami, V., Micciancio, D., Regev, O.: The complexity of the covering radius problem on lattices and codes. Computational Complexity 14(2), 90–121 (2005); Preliminary version in CCC 2004
17. Haviv, I., Lyubashevsky, V., Regev, O.: A note on the distribution of the distance from a lattice. Discrete and Computational Geometry 41(1), 162–176 (2009)
18. Haviv, I., Regev, O.: Hardness of the covering radius problem on lattices. In: Proc. of 21st IEEE Annual Conference on Computational Complexity (CCC), pp. 145–158 (2006)
19. Haviv, I., Regev, O.: Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In: Proc. 39th ACM Symp. on Theory of Computing (STOC), pp. 469–477 (2007)
20. Kannan, R.: Minkowski's convex body theorem and integer programming. Math. Oper. Res. 12, 415–440 (1987)
21. Khot, S.: Hardness of Approximating the Shortest Vector Problem in Lattices. In: IEEE Symposium on Foundations of Computer Science (FOCS), pp. 126–135 (2004)
22. Lenstra, A., Lenstra, H., Lovász, L.: Factoring polynomials with rational coefficients. Math. Ann. 261, 515–534 (1982)
23. Micciancio, D.: Almost perfect lattices, the covering radius problem, and applications to Ajtai's connection factor. SIAM Journal on Computing 34(1), 118–169 (2004); Preliminary version in STOC 2002
24. Micciancio, D.: Efficient reductions among lattice problems. In: SODA, pp. 84–93 (2008)
25. Micciancio, D.: Inapproximability of the shortest vector problem: Toward a deterministic reduction. Electronic Colloquium on Computational Complexity (ECCC) 19 (2012)
26. Micciancio, D., Goldwasser, S.: Complexity of Lattice Problems: A Cryptographic Perspective. The Kluwer International Series in Engineering and Computer Science, vol. 671. Kluwer Academic Publishers, Boston (2002)
27. Micciancio, D., Regev, O.: Lattice-based cryptography. In: Bernstein, D.J., Buchmann, J. (eds.) Post-quantum Cryprography, pp. 147–191. Springer (2008)
28. Micciancio, D., Voulgaris, P.: A deterministic single exponential time algorithm for most lattice problems based on voronoi cell computations. In: Proc. 42nd ACM Symposium on Theory of Computing (STOC), pp. 351–358 (2010)
29. Milman, V.D., Schechtman, G.: Asymptotic theory of finite dimensional normed spaces. Springer-Verlag New York, Inc., New York (1986)
30. Peikert, C.: Limits on the hardness of lattice problems in ℓ_p norms. Computational Complexity 17(2), 300–351 (2008); Preliminary version in CCC 2007
31. Regev, O., Rosen, R.: Lattice problems and norm embeddings. In: Proc. 38th ACM Symp. on Theory of Computing (STOC), pp. 447–456 (2006)
32. Schnorr, C.-P.: A hierarchy of polynomial time lattice basis reduction algorithms. Theoretical Computer Science 53(2-3), 201–224 (1987)
33. Spencer, J.: Six standard deviations suffice. Trans. Amer. Math. Soc. 289(2), 679–706 (1985)

Approximation Algorithms for Generalized and Variable-Sized Bin Covering

Matthias Hellwig[1] and Alexander Souza[2]

[1] Humboldt University of Berlin, Germany
mhellwig@informatik.hu-berlin.de
[2] Apixxo AG, Switzerland
alex.souza@apixxo.com

Abstract. We consider the Generalized Bin Covering problem: We are given m bin types, where each bin of type i has profit p_i and demand d_i. Furthermore, there are n items, where item j has size s_j. A bin of type i is said to be covered if the set of items assigned to it has total size of at least d_i. For earning profit p_i a bin of type i has to be covered. The objective is to maximize the total profit. Only the cases $p_i = d_i = 1$ (Bin Covering) and $p_i = d_i$ (Variable-Sized Bin Covering) have been treated before. We study two models of bin supply: In the unit supply model, we have exactly one bin of each type, i. e., we have individual bins. By contrast, in the infinite supply model, we have arbitrarily many bins of each type. Both versions of the problem are NP-hard and can not be approximated better than 2 in polynomial time, unless P = NP.

We prove that there is a combinatorial 5-approximation algorithm for Generalized Bin Covering with unit supply, which has running time $O(nm\sqrt{m+n})$. This also transfers to the infinite supply model. Furthermore, for Variable-Sized Bin Covering, in which we have $p_i = d_i$, we show that the natural and fast Next Fit Decreasing (nfd) algorithm is a 9/4-approximation in the unit supply model. The bound is tight for the algorithm and close to being best-possible.

The above results in the unit supply model not hold only asymptotically, but for all instances. This contrasts most of the previous work on Bin Covering, which has been asymptotic. Additionally, we can extend an existing AFPTAS for Bin Covering in order to obtain an AFPTAS for Variable-Sized Bin Covering in the *infinite* supply model.

1 Introduction

Models and Motivation. We study generalizations of the NP-hard classical Bin Covering problem. In this problem we have an infinite supply of unit-sized bins and a collection of items having individual sizes. Each item has to be assigned to a bin. A bin is said to be covered if the total size of the assigned items is at least the size of the bin. We seek an assignment of items that maximizes the number of *covered* bins. In order to distinguish the size of bins and the size of items we refer in the following to the *demand* of a bin instead of its size. This notion also captures that a bin has to be covered in order to contribute to the objective function.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 194–205, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Bin Covering has received considerable attention in the past [1,2,3,5,4,9]. In this paper we study Generalized Bin Covering: We have a set $I = \{1, \ldots, m\}$ of *bin types* and each bin type $i \in I$ has a *profit* p_i and *demand* d_i. We denote the set of items by $J = \{1, \ldots, n\}$ and define that each item $j \in J$ has a *size* s_j. A bin is *covered* or *filled* if the total size of the packed items is at least the demand d_i of the bin, in which case we earn profit p_i. The goal is to maximize the total profit gained. The special case with $p_i = d_i$ is known as Variable-Sized Bin Covering. The special case with $p_i = d_i = 1$ is the classical Bin Covering problem. To the best of our knowledge, the model with general profits and demands has not been studied in the Bin Covering setting before. Furthermore, we consider two models of bin supply: In the *infinite supply model* we have arbitrary many bins of each bin type available. By contrast, we introduce the more general *unit supply model*. In this model we have only one bin per type available. Thus, in the following, we rather speak of individual bins than of bin types. Note that some of these individual bins are allowed to have identical demand. Hence by introducing n copies of each bin type, we can simulate the infinite supply model with the unit supply model. The converse is obviously not true.

For motivating these generalizations, we mention the following application from trucking. Suppose that a moving company receives a collection of inquiries for moving contracts. Each inquiry has a certain volume and yields a certain profit if it is served (entirely). The company has a fleet of trucks, where each truck has a certain capacity. The objective is to decide which inquiries to serve with the available trucks as to maximize total profit. Inquiries map to bins, while trucks map to items in the setting of Generalized Bin Covering. Notice that in particular the unit supply model is essential here, since the inquiries are individual, i. e., are not available arbitrarily often. To the best of our knowledge, all previous work on Bin Covering exclusively considers the infinite supply model and is hence not applicable here. Also note that the Generalized Bin Covering problem applies in particular, if the profits do not necessarily correlate with the volume, but also depend on the types of goods.

Let $\mathcal{I}$ denote the family of all sets of bin types and $\mathcal{J}$ the family of all sets of items. Furthermore, let $\textsc{alg}(I, J)$ and $\textsc{opt}(I, J)$ be the respective profits gained by some algorithm alg and by an optimal algorithm opt on an instance $(I, J) \in \mathcal{I} \times \mathcal{J}$. The *approximation ratio* of an algorithm alg, is defined by $\rho(\textsc{alg}) = \sup\{\textsc{opt}(I, J)/\textsc{alg}(I, J) \mid I \in \mathcal{I}, J \in \mathcal{J}\}$. If $\rho(\textsc{alg}) \leq \rho$ holds for an algorithm alg with running time polynomial in the input size, then it is called a *ρ-approximation*. If there is a $(1 + \varepsilon)$-approximation for every $\varepsilon > 0$, then the respective family of algorithms is called a *polynomial time approximation scheme* (PTAS). If the running of a PTAS is additionally polynomial in $1/\varepsilon$, then it is called a *fully polynomial time approximation scheme* (FPTAS). With $\bar{\rho}(\textsc{alg}) = \lim_{p\to\infty} \sup\{\textsc{opt}(I, J)/\textsc{alg}(I, J) \mid I \in \mathcal{I}, J \in \mathcal{J}, \textsc{opt}(I, J) \geq p\}$ we denote the *asymptotic approximation ratio* of an algorithm alg. The notions of an asymptotic approximation algorithm and of asymptotic (F)PTAS (A(F)PTAS) transfer analogously.

Related Work. To the best of our knowledge, all of the previous work considers the (Variable-Sized) Bin Covering problem in the infinite supply model. Surveys on offline and online versions of these problems are given by Csirik and Frenk [3] and by Csirik and Woeginger [6]. Historically, research (on the offline version) of the Bin Covering problem was initiated by Assmann et al. [1]. They proved that Next Fit is a 2-approximation algorithm and that First Fit Decreasing is an asymptotic 3/2-approximation. They improved on this result by giving an asymptotic 4/3-approximate algorithm. Csirik et al. [2] also obtained asymptotic approximation guarantees of 3/2 and 4/3 with simpler heuristics. The next breakthrough was achieved by Csirik, Johnson, and Kenyon [4] by giving an APTAS for the classical Bin Covering problem. Their algorithm is based on a suitable LP relaxation and a rounding scheme. Later on, Jansen and Solis-Oba [9] improved on the running time and gave an AFPTAS. They reduce the number of variables by approximating the LP formulation of Csirik et al. [4], which yields the desired speed-up. Csirik and Totik [5] gave a lower bound of 2 for every algorithm for online (Variable-Sized) Bin Covering, where items arrive one-by-one. This bound holds also asymptotically. Clearly, the algorithm Next Fit, which uses only the largest bin type, is already an asymptotic 2-approximation.

Our Contribution. In Section 2 we consider Generalized Bin Covering in the unit supply model. Our first main result, stated in Theorem 1, is a 5-approximation algorithm having running time $O(nm\sqrt{m+n})$. The basic idea is to solve a modified version of the problem optimally. Even though the found solution may not be feasible for the original problem, it will enable us to provide a good solution for it. As a side result, which might be interesting in its own right, we obtain an integrality gap of two for a linear program of the modified problem and a corresponding integer linear program.

For Variable-Sized Bin Covering in the infinite supply model, it is not hard to see that any reasonable algorithm (using only the largest bin type) is an asymptotic 2-approximation. The situation changes considerably in the *unit* supply model: Firstly, limitations in bin availability have to be respected. Secondly, the desired approximation guarantees are non-asymptotic. Our main result here is a tight analysis of the Next Fit Decreasing (nfd) algorithm in the unit supply model for Variable-Sized Bin Covering, which can be found in Section 3. Theorem 4 states that nfd yields an approximation ratio of at most $9/4 = 2.25$ with running time $O(n \log n + m \log m)$. Example 1 shows that this bound is tight. The main idea behind our analysis is to classify bins according to their coverage: The bins that nfd covers with single items are – in some sense – optimally covered. If a bin is covered with at least two items, then their total size is at most twice the demand of the covered bin. Hence those bins yield at least half of the achievable profit. Intuitively, the problematic bins are those that are not covered by nfd: An optimal algorithm might recombine leftover items of nfd with other items to cover some of these bins and increase the profit gained. Our analysis gives detailed insight into the limited improvements to which such recombinations can lead. Firstly, our result is interesting in its

own right, since NFD is a natural and fast algorithm. Secondly, it is also close to being best possible, in the following sense. A folklore reduction from PARTITION yields that even the classical BIN COVERING problem is not approximable within a factor of two, unless P = NP. This clearly excludes the possibility of a PTAS for BIN COVERING in any of the models. The reduction crucially uses that there are only two identical bins in the BIN COVERING instance it creates. Then the question arises if one can improve in an asymptotic notion, where the optimal profit diverges. Indeed, for the classical BIN COVERING problem with *infinite supply*, there is an A(F)PTAS [4,9].

Since we have individual bins rather than bin types in the *unit supply* model, there are difficulties for defining meaningful asymptotics for VARIABLE-SIZED BIN COVERING therein. We elaborate more on this issue in a full version of the paper. In Theorem 8 we show that, if there are $m > 2$ bins available and the profit as well as the number of covered bins of an optimal solution diverges, there are instances for which no polynomial-time algorithm can have an approximation ratio smaller than $2-\varepsilon$ for any $\varepsilon > 0$, unless P = NP. Intuitively, we show that in this asymptotic notion one still has to solve a PARTITION instance on two "large" bins. Hence, for this asymptotics there is no APTAS for VARIABLE-SIZED BIN COVERING in the *unit* supply model, unless P = NP. However, this fact does *not* exclude the possibility of an A(F)PTAS for VARIABLE-SIZED BIN COVERING in the *infinite* supply model. Indeed, we can give an A(F)PTAS for VARIABLE-SIZED BIN COVERING with infinite supply. Our algorithm is an extension of the APTAS of Csirik et al. [4] for classical BIN COVERING. We remove bin types with small demands and adjust the LP formulation and the rounding scheme used by [4]. The running-time of the APTAS can be further improved using the involved method of Jansen and Solis-Oba [9] to yield the claimed AFPTAS in Theorem 9.

Notation. For any set $K \subseteq J$ define the *total size* by $s(K) = \sum_{k \in K} s_k$. Note that a bin $i \in I$ is covered by a set $K \subseteq J$, if $s(K) \geq d_i$. As a shorthand, define $s = s(J)$. Any assignment of items to bins is a solution of the GENERALIZED BIN COVERING problem. We will denote such an assignment by a collection of sets $S = (S_i)_{i \in I}$, where the $S_i \subseteq J$ are pairwise disjoint subsets of the set J of items. Denote the profit of a solution S by $p(S) = \sum_{i \in I : s(S_i) \geq d_i} p_i$. The profit of a solution S determined by some algorithm ALG on an instance (I, J) is denoted by $\text{ALG}(I, J) = p(S)$. We may omit the instance (I, J) in calculations, if it is clear to which instance ALG refers to. Furthermore, for a solution S of an algorithm ALG, let $u_{\text{ALG}}(i) = s(S_i)$ be the total size of the items assigned to bin i. If no confusion arises, we will write $u(i)$ instead of $u_{\text{ALG}}(i)$.

2 Generalized Bin Covering

Theorem 1. *There exists a* 5*-approximation for* GENERALIZED BIN COVERING *in the unit supply model, which has running time* $O(nm\sqrt{m+n})$.

It is not hard to see that naive greedy strategies that assign items to most profitable bins or that assign items to bins with the best ratio of profit to demand

do not yield a constant approximation ratio. We give examples in a full version of the paper. There we also give all missing proofs. We start with an informal description of the ideas of our algorithm and define terms formally below.

At the heart of our analysis lies the following observation. In an optimal solution either a not too small fraction of bins is covered with only one item exceeding the demand of the respective bin or a large fraction of bins is covered with more than one item, and all these items are smaller than the demand of the bin they were assigned to. We explain below, why this can be assumed to hold true. In the former case we speak of singular coverage and in the latter of regular coverage. It is easy to see (cf. Observation 2) that a bipartite maximum matching gives a solution being at least as good as the partial optimal solution of singularly covered bins. More difficult to handle is the case when a large fraction of bins is covered regularly in an optimal solution. We address these difficulties by considering an appropriately modified Bin Covering problem. In this problem items are only allowed to be assigned to bins with demand of at most their size. In this situation we say that the items are admissible to the respective bins. Furthermore, it is allowed to split items into parts and these parts may be distributed among the bins to which the whole item is admissible. Intuitively, in this modified problem the profit gained for a bin is the fraction of demand covered multiplied with the profit of the respective bin.

In Lemma 1 we show that the modified problem can be solved optimally in polynomial time by a greedy algorithm ALG*. Algorithm ALG* considers bins in non-increasing order of efficiency, where the efficiency of a bin is defined as the ratio of profit to demand of the respective bin. For each bin i ALG* considers the largest item j, which is admissible to i. If j was not assigned or only a part of j was assigned previously, then j, respectively the remaining part of j, is assigned to i. Then ALG* proceeds with the next smaller item. Once a bin is covered, the item that exceeds this bin is split in order to exactly cover the bin. Note that it can happen that during this procedure bins receive items, but are not covered. Nonetheless, by definition of the modified problem, these bins proportionally contribute to the objective function. The solution found by this algorithm is optimal. We show this by transforming an arbitrary optimal solution to a linear program formulation of the modified problem – for example found by an LP solver – into the solution of ALG* without losing any profit.

In contrast to an arbitrary optimal solution, the solution found by algorithm ALG* has additional structural properties. We crucially use these to transform it via two steps into a good solution for the Generalized Bin Covering problem. We are able to reassemble the split items in Lemma 2 without losing too much profit in the modified model. The solution is further modified in a greedy way such that there are no items on a not covered bin i, which are admissible to another not covered bin i' having larger efficiency. A solution with this property is called maximal with respect to the modified problem.

In Lemma 3 we show how to create a solution for the Generalized Bin Covering problem from a maximal solution obtained in the above way, again by losing only a bounded amount of profit. For this we move items successively

from a not covered bin to the next not covered bin, which has at least the same efficiency. Since the solution was maximal, the bins with higher efficiency are covered. By this procedure all bins are covered, which were not covered in the maximal solution, except the least efficient one. Either this least efficient bin or the remaining ones yield at least half of the profit of all bins in the maximal solution. Therefore, after applying this procedure at most half of the profit is lost in comparison to the maximal solution. But now, all bins that have received items are actually covered after this procedure.

We now start with the proof of Theorem 1. Let $S = (S_1, \ldots, S_m)$ be any solution. During the analysis we can assume that $S_1 \cup \cdots \cup S_m = J$, i. e. all considered algorithms assign all items to some bin. This is justified, since we could add a dummy bin $m+1$ with $p_{m+1} = 0$ and $d_{m+1} = \infty$ for sake of analysis.

A covered bin i is said to be covered *singularly* if $S_i = \{j\}$ for some $j \in J$ with $s_j > d_i$, otherwise it is said to be covered *regularly*. Since we can assume that all items can be assigned to bins, we can also make the following assumptions. A bin i containing an item j with $s_j > d_i$ is singularly covered. For a bin i, which is covered regularly, it holds $s(S_i) \leq 2d_i$. The latter can be assumed to hold true, since the bin i does not contain an item j with $s_j > d_i$ and hence, in case $s(S_i) > 2d_i$, we could remove an item and the bin i would still be covered.

Observation 2. *For an optimal solution O on an instance (I, J) let $I_S \subseteq I$ be the bins covered singularly in O and let $J_S \subseteq J$ be the set of items in O assigned to the bins from I_S. There is an algorithm* ALG *such that for every instance (I, J) there holds $\text{ALG}(I, J) \geq \text{OPT}(I_S, J_S)$. The running time is $O(nm\sqrt{m+n})$.*

Proof (Sketch). Construct a bipartite graph $G = (I \cup J, E)$, where E contains an edge (i, j), if item j can cover bin i singularly. Recall, p_i is the profit of bin i. We set the weight of (i, j) to be p_i. A Maximum Weight Bipartite Matching [8] in G is at least as large as the value of the partial solution of singularly bins covered by OPT in the instance (I, J). □

Consider the following modified Bin Covering problem. An item j may be split by an algorithm into $p_j \geq 1$ parts. We will refer to such an item j as p_j many items $(j, 1), \ldots (j, p_j)$ of positive size, where we may omit the braces in indices. We refer to the (j, l) as the parts of the item j. The size of item part (j, i) of item j is denoted by $s_{j,i}$. Formally it has to hold $s_j = \sum_{i=1}^{p_j} s_{j,i}$ and $s_{j,l} > 0$ for $1 \leq l \leq p_j$.

An item j is said to be admissible to a bin i, if $s_j \leq d_i$. The parts (j, l) of an item j are defined to be admissible to i if and only if j is admissible to i. Item parts can only be assigned to bins to which they are admissible.

For a fixed solution $S = (S_1, \ldots, S_m)$ let S_i be the set of item parts, assigned to bin i. Let $y_i := \min\{s(S_i)/d_i, 1\}$. Intuitively, y_i is the "fill level" of bin i. Note, that this "fill level" of a bin i may be at most one, but nevertheless $s(S_i) > d_i$ is permitted, i. e. the sum of item sizes assigned to bin i may exceed its demand. The profit gained for bin i in the modified problem is $p^*(S_i) := p_i y_i$, which intuitively is the percentage of covered demand multiplied with the profit of the bin, where the maximal profit that can be gained is bounded by p_i. Further for

a solution $S = (S_1, \ldots, S_m)$ on an instance (I, J) let $p^*(S) = \sum_{i \in I} p^*(S_i)$. The goal is to find a solution S that maximizes $p^*(S)$ for a given instance (I, J).

Let the efficiency e_i of bin i be $e_i := p_i/d_i$. Due to space limitations algorithm ALG* for the modified Bin Covering problem is described formally in a full version of the paper. Let ALG$^*(I, J)$ denote as usual the value of its solution for the modified problem on the instance (I, J). Analogously let OPT$^*(I, J)$ be the value of an optimal solution to the modified Bin Covering problem.

Lemma 1. *Algorithm* ALG* *gives a solution of value* ALG$^*(I, J) =$ OPT$^*(I, J)$.

Observation 3. *Let S be a solution with the property $s(S_i) \leq d_i$ for all $i \in I$. Let S^* be a solution having the property $s(S_i^*) \leq 2d_i$ for all $i \in I$. If for all item parts $j \in J$ with $j \in S_i$ and $j \in S_{i'}^*$ there holds $e_i \leq e_{i'}$, then $p^*(S) \leq 2p^*(S^*)$.*

We say that a solution S *contains no split items*, if for all $i \in I$ and $j \in S_i$ there holds $s_{j,1} = s_j$. We call a solution S containing no split items *maximal with respect to the modified* Bin Covering *problem*, if there are no two distinct bins i and i' with $0 < s(S_i) < d_i$ and $0 < s(S_{i'}) < d_{i'}$ and $e_i \leq e_{i'}$, such that there is an item $j \in S_i$, which is admissible to bin i'. Note that this implies the following. If we assign in a maximal solution only one item j from such a bin i to such a bin i', then bin i' is already covered by this single item. This comes from the fact that j is not admissible to i' by the maximality of the solution.

Lemma 2. *Let S be the solution given by* ALG* *for the modified problem. S can be transformed into a solution S^* with the following properties. There holds $p^*(S) \leq 2p^*(S^*)$, S^* contains no split items and S^* is maximal with respect to the modified* Bin Covering *problem.*

Lemma 3. *Let S be a solution containing no split items and being maximal with respect to the modified* Bin Covering *problem. S can be transformed into a solution S^* for the* Generalized Bin Covering *problem with $p^*(S) \leq 2p(S^*)$.*

Proof (of Theorem 1). Let (I, J) be the given instance. Our algorithm works as follows. We use Observation 2 to find a solution S_1. Then we run ALG* on the instance (I, J) and let S be the solution output. We transform solution S into a solution S' as done in Lemma 2 and then solution S' into a solution S_2 as done in Lemma 3. We output the better solution from $\{S_1, S_2\}$. The running time is dominated by the algorithm for Maximum Weight Bipartite Matching [8].

For the proof of the approximation guarantee, fix an optimal solution O to the instance (I, J). Let $I_R \subseteq I$ be the set of bins covered regularly by the solution O and $J_R = \{j \in J \mid \exists i \in I_R : j \in O_i\}$, the set of items in these bins. Let $I_S \subseteq I$ be the set of bins covered singularly by the solution O and $J_S = \{j \in J \mid \exists i \in I_S : j \in O_i\}$, the set of items on these bins. We have OPT$(I, J) =$ OPT$(I_R, J_R) +$ OPT(I_S, J_S). Thus, OPT$(I, J) -$ OPT$(I_R, J_R) =$ OPT(I_S, J_S).

If OPT$(I_R, J_R) < 4/5 \cdot$ OPT(I, J), then OPT$(I_S, J_S) > 1/5 \cdot$ OPT(I, J) by the above. Hence in this case OPT$(I, J) \leq 5$ALG(I, J) using Observation 2.

Otherwise, if OPT$(I_R, J_R) \geq 4/5 \cdot$ OPT(I, J), then we have OPT$(I_S, J_S) \leq 1/5 \cdot$ OPT(I, J). We find the claimed OPT$(I, J) \leq 5 \cdot$ ALG(I, J) as follows. We

have $\text{OPT}(I,J) = \text{OPT}(I_R,J_R) + \text{OPT}(I_S,J_S) \leq \text{OPT}^*(I_R,J_R) + 1/5 \cdot \text{OPT}(I,J)$, where we use $\text{OPT}^*(I_R,J_R) \geq \text{OPT}(I_R,J_R)$ and the assumption of the case. There holds $\text{OPT}^*(I,J) + 1/5 \cdot \text{OPT}(I,J) = \text{ALG}^*(I,J) + 1/5 \cdot \text{OPT}(I,J)$ by Lemma 1. Further $\text{ALG}^*(I,J) + 1/5 \cdot \text{OPT}(I,J) \leq 4 \cdot \text{ALG}(I,J) + 1/5 \cdot \text{OPT}(I,J)$, where we have accounted for transforming the fractional solution to the modified problem into a solution for the Generalized Bin Covering problem with Lemmas 2 and 3. This gives the claim. □

3 Variable-Sized Bin Covering

3.1 Tight Analysis of NFD in the Unit Supply Model

In this subsection we consider the unit supply model and it is assumed that $d_i = p_i$ for all i. The algorithm Next Fit Decreasing (NFD) works as follows. The algorithm considers bins in non-increasing order of demand. For each bin, if the total size of the unassigned items suffices for coverage, it assigns as many items (also non-increasing in size) as necessary to cover the bin. Otherwise, the bin is skipped. Due to lack of space we omit a formal description here. In this section we assume that we have $d_1 \geq \cdots \geq d_m$ and $s_1 \geq \cdots \geq s_n$, as needed by the algorithm.

Example 1. Let $2/3 > \varepsilon > 0$ be arbitrary. The following instance (I,J) yields that NFD gives an approximation not better than $9/4 - 2\varepsilon$. Hence NFD is at least a $9/4$-approximation. Let $I = \{4, 3-2\varepsilon, 3-2\varepsilon, 3-2\varepsilon\}$ and $J = \{2-\varepsilon, 2-\varepsilon, 2-\varepsilon, 1-\varepsilon, 1-\varepsilon, 1-\varepsilon\}$. Observe we have $\text{NFD}(I,J) = 4$ and $\text{OPT}(I,J) = 9 - 6\varepsilon$.

Theorem 4. *Algorithm* NFD *is a* $9/4$*-approximation, which has running time* $O(n \log n + m \log m)$*. The bound on the approximation factor is tight.*

Note that this is almost best possible, since the problem is inapproximable up to a factor of two, unless P = NP.

Proof Techniques. We will use three kinds of arguments. The first type we call a volume argument. If s is the sum of item sizes in the (remaining) instance, we have $\text{OPT} \leq s$. This argument holds independently of the actual demands of bins. Such volume bounds are too weak in general to achieve the claimed bound, thus we need arguments using the structure of bins in the instance, which is the second type of arguments. For example, if the sum of item sizes in the (remaining) instance is αd, $\alpha > 1$ and the demand of the only bin in the instance is d, then it follows $\text{OPT} \leq d$, while we could only conclude $\text{OPT} \leq \alpha d$ with a volume argument. The third type of argument we use are arguments transforming instances. These arguments give that we can w. l. o g. restrict ourselves to analyze instances having certain properties. For example, we may assume that there are no items in the instance with size larger than the largest bin demand.

Proof Outline. Our proof looks at the specific structure of the solution given by NFD and argues based on that, how much better an optimal solution can be. We employ the described techniques in the following way. Firstly, we settle two basic

properties of NFD: A solution of NFD is unique and if a bin is covered with at least twice its demand, then there is only one item assigned to it. These properties will be used implicitly during the analysis. After that, we give transformation arguments, which allow us to restrict ourselves to analyze instances with the following properties. We may assume that NFD covers the first bin (Observation 5), and that the "right-most bins" (i. e. the bins with the least demand – or the smallest bins) are empty (Observation 6), where we will specify this notion in more detail later. We will show that we may assume that the "left-most bins" (i. e. the largest bins) are only assigned items such that they do not exceed twice their demand (Lemma 4). Here "left-most bins" refers to the bins up to the first empty bin.

With these tools at hand we can come to the actual proof. The central notion here is the *well-covered* bin (Definition 1): Consider the right-most (i.e., smallest) empty bin in the instance with the property that all larger bins are assigned items only up to twice their demand. If such a bin exists, then we call the covered bins of these well-covered. The proof will be inductive. The terminating cases are the ones, when there are either at least four well-covered bins (Observation 7) or between two and three well-covered bins and there is a bin among these containing at least three items (Lemma 9). These cases are settled by volume arguments which is also the reason, why they are terminating cases – even if there are additional filled but not well-covered bins in the instance. We are also in terminating cases if the above prerequisites are not met, but there are no filled bins which are not well-covered: Lemma 5 treats the case that all of the at most three well-covered bins contain at most two items and Lemma 6 gives the cases, in which we have exactly one well-covered bin in the instance.

If there are additional filled but not well-covered bins and we cannot apply volume arguments – as in the both last mentioned situations – , we have to look at the instance more closely. Our idea is here to consider a specific not well-covered bin, which will be called the *head of the instance*. We will subdivide such an instance into two parts, which is done by the key lemma of the recursion step, the Decomposition Lemma 10. Therein and in Lemma 7 we show that it is not advantageous to assign items, which NFD assigned to bins with larger demand than the demand of the head of the instance, to bins with smaller demand than the demand of the head of the instance. This allows us in combination with some estimations to consider the left part of the instance and the right part separately. For the left part Lemma 8 and Lemma 10 give that the approximation factor of NFD is at most $9/4$ and the right part of the instance is a smaller instance and we may hence iteratively apply the argumentation.

Observation 5. *Fix an instance* (I, J). *If* $u_{\mathrm{NFD}}(1) = \cdots = u_{\mathrm{NFD}}(i) = 0$ *then* $u_{\mathrm{OPT}}(1) = \cdots = u_{\mathrm{OPT}}(i) = 0$. *Let* $I'' = I \backslash \{1, \ldots, i\}$. *Then* $\mathrm{NFD}(I, J) = \mathrm{NFD}(I'', J)$ *and* $\mathrm{OPT}(I, J) = \mathrm{OPT}(I'', J)$.

By the argument given by the previous observation it is also justified to assume $\mathrm{NFD}(I, J) > 0$. Since otherwise also $\mathrm{OPT}(I, J) = 0$ follows and NFD is optimal. This assumption will always be implicitly used and thus the quotient

$\text{OPT}(I,J)/\text{NFD}(I,J)$ is always defined. Alternatively, if $\text{NFD}(I,J) = \text{OPT}(I,J) = 0$, we could define $\text{OPT}(I,J)/\text{NFD}(I,J) := 1$. Further, we may always assume that there exists an empty bin, otherwise NFD is clearly optimal. We may strengthen this observation such that it suffices to compare instances of NFD to OPT, where the right-most bins are all empty, i. e. there is a non-empty bin i', the bin $i'+1$ is empty and all bins with higher indices, if they exist, are also empty.

Observation 6. *Fix the solution of* NFD *on an instance* (I,J)*. Let* i^* *be a bin with* $u(i^*) = 0$ *and for all* $i > i^*$ *we have* $u(i) > 0$*. Then* $\text{OPT}(I,J)/\text{NFD}(I,J) \leq \text{OPT}(I',J)/\text{NFD}(I',J)$*, where* $I' = I \setminus \{i^*+1,\dots,m\}$*.*

Due to this observation we may assume that $u(m) = 0$ from now on. The next lemma states that we can assume w. l. o g. that all bins i up to the bin with smallest index i^*, such that $u(i^*+1) = 0$, receive only items in such a way, that $u(i) \leq 2d_i$ for $i < i^*$.

Lemma 4. *Let* (I,J) *be an instance and consider a solution of* NFD *for it. Let* i^* *be the smallest index, such that* i^* *is a bin with* $u(i^*) > 0$ *and* $u(i^*+1) = 0$*. Let* $i_1,\dots,i_k \in \{1,\dots,i^*\}$ *be the indices with* $u(i_j) \geq 2d_{i_j}$ *for* $j = 1,\dots,k$ *and let* $j_1,\dots,j_k$ *be the items on these bins. Set* $I' = I \setminus \{i_1,\dots,i_k\}$ *and* $J' = J \setminus \{j_1,\dots,j_k\}$*. Then* $\text{OPT}(I,J)/\text{NFD}(I,J) \leq \text{OPT}(I',J')/\text{NFD}(I',J')$*.*

Definition 1. *Consider a solution of* NFD *for an instance* (I,J)*. Fix a bin* i^**, with* $u(i^*) > 0$*, and let* i' *be the smallest number with* $i' > i^*$ *such that* $u(i') = 0$*, if it exists. We call the bin* i^* *well-covered, if* i' *exists and* $u(i) \leq 2d_i$ *for all* $i = 1,\dots,i'$*.*

Observation 7. *Let* (I,J) *be given. If* NFD *gives a solution with* $k \geq 1$ *well-covered bins, then* $\text{OPT}(I,J)/\text{NFD}(I,J) \leq 2 + 1/k$*.*

Proof. Let $k' \geq k$ be the largest index of a well-covered bin and let $I' = \{i \in \{1,\dots,k'\} \mid u(i) > 0\}$ be the set of well-covered bins. On the one hand we have $\text{NFD}(I,J) \geq \sum_{i\in I'} d_i$ and on the other $\text{NFD}(I,J) \geq kd_{k'}$. Recall, we have for every $i \in I'$ that $u(i) \leq 2d_i$. Let l be the index of the first item, which NFD did not assign to a bin with index k' or smaller. Since $u(k'+1) = 0$ by definition of k', we further have $\sum_{j=l}^{n} s_j < d_{k'+1} \leq d_{k'}$, otherwise NFD would have filled bin $k'+1$. For the sum of item sizes $s = \sum_{j=1}^{n} s_j < \sum_{i\in I'} 2d_i + d_{k'}$. Because $\text{OPT} \leq s$ we can bound $\text{OPT} < \sum_{i\in I'} 2d_i + d_{k'} \leq 2\text{NFD} + 1/k \cdot \text{NFD} = (2+1/k)\text{NFD}$. □

By Observation 5 and Lemma 4 it can be shown that we may assume that there is at least one well-covered bin in the instance. Hence by Observation 7 we can conclude that NFD is at most a 3-approximation. Additionally, for a number of $k \geq 4$ well-covered bins this observation already gives the desired result. Hence we now turn our attention to the cases where $k \leq 3$.

Lemma 5. *Let* (I,J) *be an instance. If* NFD *gives a solution, in which every filled bin is well-covered and contains at most two items, then there holds* $\text{OPT}(I,J)/\text{NFD}(I,J) \leq 2$*.*

Lemma 6. *Let (I, J) be an instance. If* NFD *gives a solution with $k = 1$ well-covered bins and all other bins are empty, then* $\text{OPT}(I, J)/\text{NFD}(I, J) \leq 9/4$.

In order to simplify the following statements we introduce the term head of the instance, which is a distinguished bin. For this, fix a solution of NFD to a given instance (I, J). Let i_0 be the index of the first not well-covered bin with $u(i_0) > 0$ and let i_1 be the smallest index such that $u(i_1+1) = 0$ with $i_1 \geq i_0$. Let $i^* = \max_{i:u(i)>2d_i}\{i \leq i_1\}$. Then, the bin i^* is called the *head* (of the instance).

Lemma 7. *Let (I, J) be an instance on which* NFD *gives a solution with $k = 1$ well-covered bin and which contains a non-empty bin, which is not well-covered. Let i^* be the head of the instance. If there are at least three items in bin 1 in* NFD*'s solution and* OPT *assigns at least one of these items to a bin with index at least i^*, then* $\text{OPT}(I, J)/\text{NFD}(I, J) \leq 9/4$.

Lemma 8. *Let (I, J) be an instance on which* NFD *gives a solution with $k \in \{1, 2, 3\}$ well-covered bins and each of these contains at most two items. Moreover let the solution contain at least one non-empty bin, which is not well-covered, and let i^* be the head of the instance. Define $I' = \{1, \ldots, i^*\}$ and $I'' = I \setminus I'$. Further let J' be the set of items, which reside on a bin from I' in* NFD*'s solution, and let $J'' = J \setminus J'$. Fix an optimal solution O. Let A be the set of items, which reside in O on a bin from I', and let $B = J \setminus A$. Then* $(\text{OPT}(I', A) + \text{OPT}(I'', B \setminus J''))/\text{NFD}(I', J') \leq 2$.

Lemma 9. *Let (I, J) be an instance on which* NFD *gives a solution with $k \geq 2$ well-covered bins. If at least one of these bins contains at least three items, then* $\text{OPT}(I, J)/\text{NFD}(I, J) \leq 9/4$.

Lemma 10 (Decomposition Lemma). *Let (I, J) be an instance on which* NFD *gives a solution with k well-covered bins and at least one not well-covered bin. Let i^* be the head of the instance. Let J' be the set of items residing on the bins $1, \ldots, i^*$ in* NFD*'s solution and $J'' = J \setminus J'$. Further let $I' = \{1, \ldots, i^*\}$ and $I'' = I \setminus I'$. Then* $\text{OPT}(I, J)/\text{NFD}(I, J) \leq \max\{9/4, \text{OPT}(I'', J'')/\text{NFD}(I'', J'')\}$.

Proof (of Theorem 4). First observe that Example 1 yields a lower bound of $9/4$ on the approximation ratio of NFD. Let k be the number of well-covered bins in the solution of NFD. If $k \geq 4$ then Observation 7 already gives the claim. Thus let $k \in \{1, 2, 3\}$. Firstly assume that there is no additional filled bin besides the k well-covered bins. If one of the k bins contains at least three items, then the claim follows by Lemma 6 and Lemma 9. If all k well-covered bins contain at most two items, the statement follows from Lemma 5.

Now let there be $k \in \{1, 2, 3\}$ well-covered bins in the solution of NFD and at least one additional filled bin, which is not well-covered. Define $I' = \{1, \ldots, i^*\}$, $I'' = I \setminus I'$, J' to be the set of items, which are assigned to the bins in I' by NFD and $J'' = J \setminus J'$, where i^* is the head of the instance. Now we can apply Lemma 10. Observe that (I'', J'') is a smaller instance, which has at least one not well-covered bin less. Hence we can apply the analysis recursively to this instance. The recursion terminates if (I'', J'') is an instance, such that the

solution of NFD contains only well-covered bins or only empty bins. Clearly, in the latter case we have that NFD is optimal and in the former we can argue as above. The algorithm can be implemented such that the running-time is dominated by sorting bins and items. □

3.2 Asymptotical Results for Variable-Sized Bin Covering

The details of the following results can be found in the full version of the paper. There we also discuss the difficulty of defining suitable asymptotics for the unit supply model. Here we consider an asymptotics, where the total profit and the number of covered bins in an optimal solution diverge.

Theorem 8. *Consider* Variable-Sized Bin Covering *with unit supply. Let* $2 \leq m \leq n$. *Then there is an instance* (I, J), *with* $|J| = n+m-2$, *such that in an optimal solution* m *bins are covered, but there is no polynomial time algorithm with approximation factor better than* $\rho = 2 - \frac{m-2}{s/2+m-2}$, *unless* $P = NP$.

For the Variable-Sized Bin Covering model with infinite supply the APTAS of Csirik et al. [4] and the method of Jansen and Solis-Oba [9] can be extended. The basic idea is to ignore bin types with small demand. Adjusting the parameters in the algorithms of [4] and [9] and adapting the calculations gives the desired result.

Theorem 9. *There is an AFPTAS for* Variable-Sized Bin Covering *in the infinite supply model.*

References

1. Assmann, S.F., Johnson, D.S., Kleitman, D.J., Leung, J.Y.-T.: On a dual version of the one-dimensional bin packing problem. J. Algorithms 5(4), 502–525 (1984)
2. Csirik, J., Frenk, J., Labbé, M., Zhang, S.: Two simple algorithms for bin covering. Acta Cybernetica 14, 13–25 (1999)
3. Csirik, J., Frenk, J.B.G.: A dual version of bin packing. Algorithms Review 1(2), 87–95 (1990)
4. Csirik, J., Johnson, D.S., Kenyon, C.: Better approximation algorithms for bin covering. In: SODA, pp. 557–566 (2001)
5. Csirik, J., Totik, V.: Online algorithms for a dual version of bin packing. Discrete Applied Mathematics 21, 163–167 (1988)
6. Csirik, J., Woeginger, G.J.: On-line Packing and Covering Problems. In: Fiat, A. (ed.) Online Algorithms 1996. LNCS, vol. 1442, pp. 147–177. Springer, Heidelberg (1998)
7. Grigoriadis, M.D., Khachiyan, L.G.: Coordination complexity of parallel price-directive decomposition. Math. Oper. Res., 321–340 (May 1996)
8. Hopcroft, J.E., Karp, R.M.: An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. SIAM J. C. 2(4), 225–231 (1973)
9. Jansen, K., Solis-Oba, R.: An asymptotic fully polynomial time approximation scheme for bin covering. TCS 306(1-3), 543–551 (2003)
10. Jansen, K., Zhang, H.: Approximation algorithms for general packing problems with modified logarithmic potential function. In: ICTCS, Montreal, Canada, pp. 255–266 (2002)

Approximating Minimum Linear Ordering Problems

Satoru Iwata[1], Prasad Tetali[2,3], and Pushkar Tripathi[3]

[1] Research Institute for Mathematical Sciences, Kyoto University
[2] School of Mathematics, Georgia Institute of Technology*
[3] School of Computer Science, Georgia Institute of Technology

Abstract. This paper addresses the Minimum Linear Ordering Problem (MLOP): Given a nonnegative set function f on a finite set V, find a linear ordering on V such that the sum of the function values for all the suffixes is minimized. This problem generalizes well-known problems such as the Minimum Linear Arrangement, Min Sum Set Cover, Minimum Latency Set Cover, and Multiple Intents Ranking. Extending a result of Feige, Lovász, and Tetali (2004) on Min Sum Set Cover, we show that the greedy algorithm provides a factor 4 approximate optimal solution when the cost function f is supermodular. We also present a factor 2 rounding algorithm for MLOP with a monotone submodular cost function, using the convexity of the Lovász extension. These are among very few constant factor approximation algorithms for NP-hard minimization problems formulated in terms of submodular/supermodular functions. In contrast, when f is a symmetric submodular function, the problem has an information theoretic lower bound of 2 on the approximability.

Feige, Lovász, and Tetali (2004) also devised a factor 2 LP-rounding algorithm for the Min Sum Vertex Cover. In this paper, we present an improved approximation algorithm with ratio 1.79. The algorithm performs multi-stage randomized rounding based on the same LP relaxation, which provides an answer to their open question on the integrality gap.

1 Introduction

In this paper we introduce the Minimum Linear Ordering Problem (MLOP), which generalizes several known problems such as the Minimum Linear Arrangement (MLA) and Min Sum Set Cover (MSSC) problems. Each of these problems has been extensively studied in isolation. In this paper we initiate a systematic study of these problems under the general umbrella of submodular and supermodular set functions.

An instance of the MLOP consists of a ground set V of cardinality n and a cost function $f : 2^V \to \mathbb{R}_+$. The objective is to find a linear ordering (bijection) $\sigma : V \to \{1, \ldots, n\}$ such that $\sum_i f(S_i)$ is minimized, where for any linear ordering σ of V, we define $S_i = S_i(\sigma) = \{v \mid \sigma(v) \geq i\}$. We consider three broad classes of cost functions f: supermodular, monotone submodular, and symmetric submodular functions.

* Supported in part by NSF DMS-1101447 and CCR 0910584.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 206–217, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1.1 Results and Techniques

For the case when the cost function is a *supermodular* function we establish the following theorem.

Theorem 1. *For any instance of MLOP with a supermodular cost function, the greedy algorithm yields a factor* 4 *approximation to the optimal linear ordering.*

The proof is based on a scaling technique that is similar to dual fitting. We use a histogram to represent any solution to the problem and show that the histogram corresponding to the greedy solution when scaled appropriately fits within the histogram for the optimal solution.

We also consider a special case of this problem, the min sum vertex cover problem. In this problem we are given a graph $G(V, E)$ and the objective is to arrange the vertices of G in a linear ordering σ, such that $\sum_{(u,v)\in E} \min\{\sigma(u), \sigma(v)\}$ is minimized. We achieve the following result with regard to this problem.

Theorem 2. *There exists a Las Vegas algorithm that approximates the min sum vertex cover to within a factor of* 1.79*.*

This algorithm is based on a multi-stage rounding of the natural linear programming relaxation. In our rounding technique, we randomly and independently round each of the variables in the optimal linear programming solution. However we do not perform this rounding simultaneously for all variables. Instead, we round the variables in several stages. Using this technique, we are able to achieve an approximation factor better than 2. In doing so we also answer an open question posed by [10], showing that the integrality gap of the natural LP relaxation is indeed less than 2.

We also consider the MLOP when the cost function is submodular. For *monotone* submodular functions, we obtain the following result.

Theorem 3. *For any monotone submodular function defined over a ground set of size n, there exists a deterministic algorithm for the corresponding MLOP that achieves a factor of* $2 - \frac{2}{n+1}$*.*

The algorithm is based on the Lovász extension for the given submodular function. The Lovász extensions provide a means of extending the techniques of linear programing to discrete set functions such as the ones considered here. We use the Lovász extension for the given submodular function to define a convex program that is solvable using the ellipsoid method. We then give a *deterministic* procedure to round the optimal solution to this program to get the desired integral solution.

Finally, we also consider the case when the cost function is a general submodular function, which may not be monotone. For this setting, we show an information theoretic lower bound of 2. In particular, we have the following result.

Theorem 4. *For any constant $\epsilon > 0$, there exists a family of instances of the MLOP with symmetric submodular cost functions such that no algorithm making*

polynomially many queries achieve a factor better than $2 - \epsilon$*, even if it is given infinite computational time.*

We achieve this by constructing two families of submodular cost functions that are indistinguishable, with high probability, after polynomial number of value queries, but have different optimal objective values. The ratio of the optimal values under these functions gives the desired factor. Note that the bound is information theoretic, i.e., it holds even if the algorithm is given infinite computational time, but is constrained to make only polynomially many value queries.

1.2 Prior Work

Submodular functions have been the subject of intense study over the last four decades with regards to combinatorial optimization. Special instances of the above mentioned problems have received considerable attention from the point of view of approximation algorithms. However, we are not aware of any work that has studied these problems under a unified framework of submodular/supermodular functions. We will now review some of the problems that have previously been studied in this area.

Feige, Lovász, and Tetali [10] introduced the min sum set cover (MSSC), which is a special instance of MLOP with a supermodular cost function. In this problem, we are given a hyper-graph $H = (V, E)$. For a linear ordering $\sigma : V \to \{1, \ldots, n\}$ and a hyper-edge $e \in E$, let $\hat{\sigma}(e)$ denote the minimum of $\sigma(v)$ among all the vertices v in e. The goal of the min sum set cover problem is to find a linear ordering σ that minimizes $\sum_{e \in E} \hat{\sigma}(e)$. They gave a factor 4 approximation algorithm for this problem and showed that the factor was essentially tight.

They also considered the min sum vertex cover problem described in Section 3, and gave an LP-rounding-based factor 2 approximation algorithm for the problem. This result was not provably tight and the integrality gap of the LP-relaxation was left as an open question. In subsequent work, Barenholz, Feige, and Peleg [5] provided a small improvement (with a rather technically involved analysis) by way of obtaining a 1.99995 factor approximation, and raised the question of further improving the bound. Answering this question, Theorem 2 below provides a substantial improvement, with an alternative rounding and a simpler analysis, in giving a 1.79 factor approximation.

A recent paper of Azar, Gamzu, and Yin [2] discusses a generalization of the MSSC problem in the context of reranking of search results by a search engine. In this so-called Multiple Intents Ranking (MIR) problem, we are given a hyper-graph $H = (V, E)$ with each hyper-edge $e \in E$ having a vector of nonnegative reals $w(e) = \langle w_1(e), \ldots, w_{r(e)}(e) \rangle$, where $r(e)$ denotes the number of vertices contained in e. For a linear ordering $\sigma : V \to \{1, \ldots, n\}$ and a hyper-edge $e \in E$, let $\hat{\sigma}_i(e)$ denote the i-th smallest $\sigma(v)$ among all the vertices v in e. Then the objective is to find a linear ordering σ that minimizes $\sum_{e \in E} \sum_{i=1}^{r(e)} w_i(e) \hat{\sigma}_i(e)$. Azar, Gamzu, and Yin [2] presented an $O(\log r)$-approximation algorithm, where $r = \max_{e \in E}$. They also provided a 4-approximation algorithm for monotone

non-increasing weight vectors and 2-approximation algorithm for monotone non-decreasing weight vectors. The former case includes MSSC, while the latter case generalizes the minimum latency set cover problem introduced by Hassin and Levin [12]. We refer the reader to [1] and [3] for recent developments on MIR.

We were also informed of a recent (unpublished), further generalization due to Im, Nagarajan, and van der Zwaan [14]; these authors study a so-called Minimum Latency Submodular Cover (MLSC), which generalizes the submodular ranking work of Azar and Gamzu on one hand and the Latency Covering Steiner Tree on the other. See the manuscript [14] on the arxiv, for details.

In the Minimum Linear Arrangement (MLA) problem which is a special case of submodular MLOP, we are asked to arrange the vertices of a given graph $G(V, E)$ in a linear ordering σ, so that $\sum_{(u,v)\in E} |\sigma(v) - \sigma(u)|$ is minimized. Rao and Richa [18] gave a $O(\log n)$ factor algorithms for this problem which was later improved to an $O(\sqrt{\log n} \log\log n)$ factor algorithm by Feige and Lee [9] and Charikar et. al. in [6]. The problem has also been studied on special instances and polynomial time algorithms are known for some special graphs; refer to [13] for a detailed exposition.

On the hardness front, Devanur, Khot, Saket, and Vishnoi [8] showed that the problem is hard to approximate to within any constant factor under the Unique Games Conjecture and proved that the integrality gap for the SDP relaxation of this problem is bounded from below by $O(\log\log n)$. The problem has also received considerable attention from the point of view of experimental analysis and heuristics (refer [4], [17]).

Finally, there has been recent interest in studying minimization problems with submodular cost functions [11,19,15]. However almost all the problems previously considered turn out to be quite intractable and have large polynomial lower bounds. Exceptions include the submodular vertex cover problem [11,15] and the submodular multiway partition [7].

1.3 Preliminaries

A set function f is said to be submodular if $f(X)+f(Y) \geq f(X\cup Y)+f(X\cap Y)$ holds for every $X, Y \subseteq V$. Supermodular functions are defined in a similar way; f is supermodular if $f(X)+f(Y) \leq f(X\cup Y)+f(X\cap Y)$ for every $X, Y \subseteq V$. We define a function f to be monotone if $f(X) \leq f(Y)$ for $X, Y \subseteq V$ with $X \subseteq Y$. It is called symmetric if $f(X) = f(V \setminus X)$ for every $X \subseteq V$. We assume that f is normalized, i.e., $f(\emptyset) = 0$. Note that a normalized nonnegative supermodular function is monotone, as $X \subseteq Y$ implies $f(X) \leq f(X)+f(Y\setminus X) \leq f(Y)+f(\emptyset) = f(Y)$.

Finally, a word is in order about the representation of the cost function f, since it is defined over an exponentially large domain. We use the standard *value oracle model*: that the cost function is given by a value oracle that when queried with a set S returns the value $f(S)$.

2 Supermodular Linear Ordering

This section is devoted to the linear ordering problem with supermodular cost function f. In this setting, we are given a supermodular set function f over a ground set V of size n and we are required to arrange the elements of V in a linear ordering σ such that $\sum_i f(S_i)$ is minimized, where $S_i = \{v \mid \sigma(v) \geq i\}$.

We first claim that the min sum set cover (MSSC) problem, considered in [10], is a special instance of this problem. For each $X \subseteq V$ in the hyper-graph $H(V, E)$, let $f(X)$ denote the number of edges included in X. Then f is a supermodular function. Note that $e \in E$ is included in S_i if and only if $i \leq \hat{\sigma}(e)$. Therefore, we have $\sum_{i=1}^{n} f(S_i) = \sum_{e \in E} \hat{\sigma}(e)$. Thus the min sum set cover problem is a very special case of our setting with f being a nonnegative supermodular function.

More generally, the multiple intents ranking problem is a special case of MLOP where each hyper-edge $e \in E$ has a weight $w(e) = \langle w_1(e), \ldots, w_{r(e)}(e) \rangle$. For each $X \subseteq V$ and e, let $f_e(X)$ denote the sum of the last $|X \cap e|$ components of $w(e)$, and put $f(X) = \sum_{e \in E} f_e(X)$. Then $\sum_{i=1}^{n} f(S_i) = \sum_{e \in E} \sum_{j=1}^{r(e)} w_j(e) \hat{\sigma}_j(e)$ holds for any linear ordering σ. If the weight vector $w(e)$ is monotone non-increasing, i.e., $w_1(e) \geq \cdots \geq w_{r(e)}(e)$, then f_e is a supermodular function. Thus the multiple intents ranking problem with non-increasing weight vectors is a special case of MLOP with supermodular cost functions.

In contrast, if $w(e)$ is monotone non-decreasing, then f_e is monotone submodular. Thus the multiple intents ranking problem with non-decreasing weight vectors reduces to the MLOP with monotone submodular cost functions, which will be discussed in Section 4.1.

2.1 Greedy Algorithm

We will consider the following greedy algorithm for this problem. We try to iteratively build the ordering by augmenting the current solution with the element such that the cost of the remaining elements is the smallest possible. The greedy algorithm for the supermodular linear ordering problem begins by setting $T_1 = V$. Then for $i = 1, \ldots, n$, select $v \in T_i$ that minimizes $f(T_i \setminus \{v\})$ and set $\sigma(v) = i$ and $T_{i+1} = T_i \setminus \{v\}$.

2.2 Analysis

We now prove that the greedy algorithm provides an approximate solution within a ratio of 4. Let $S_1, \ldots, S_n$ be the subsets given by $S_i = \{v \mid \sigma(v) \geq i\}$ with an optimal solution σ. Consider a histogram that consists of n columns. The i-th column has width $f(S_i) - f(S_{i+1})$, corresponding to the interval between $f(S_{i+1})$ and $f(S_i)$, and height i. The area of this diagram is equal to the optimal value of the problem denoted by OPT.

With reference to the subsets $T_1, \ldots, T_n$ generated by the greedy algorithm, construct another histogram that also consists of n columns. The i-th column has width $f(T_i) - f(T_{i+1})$, corresponding to the interval between $f(T_{i+1})$ and

$f(T_i)$, and height $p_i = \frac{f(T_i)}{f(T_i)-f(T_{i+1})}$. The area under this histogram is equal to the objective value of the greedy solution denoted by $GREEDY$.

Shrink the second diagram by a factor of 2. We now intend to show that this shrunk version of the second diagram is completely included in the first diagram. To see this, it suffices to check that $(f(T_i)/2, p_i/2)$ lies in the first histogram for each $i \in [n] = 1, 2, \ldots, n$.

For each fixed i, put $k = \lceil p_i/2 \rceil$. Then we now claim that $f(S_k) \geq f(T_i)/2$. In fact, by the procedure of the greedy algorithm, we have $f(T_i \setminus \{v\}) \geq f(T_{i+1})$ for each $v \in V \setminus T_i$. This implies that

$$\begin{aligned} f(S_k) \geq f(T_i \cap S_k) &\geq f(T_i) - \sum_{v \in T_i \setminus S_k} [f(T_i) - f(T_i \setminus \{v\})] \\ &\geq f(T_i) - |T_i \setminus S_k| \cdot [f(T_i) - f(T_{i+1})] \\ &\geq f(T_i) - (k-1)[f(T_i) - f(T_{i+1})] \\ &\geq f(T_i) - \frac{p_i}{2}[f(T_i) - f(T_{i+1})] = \frac{f(T_i)}{2}, \end{aligned}$$

The second inequality follows from the supermodularity of f. Thus, the second histogram is contained in the first one, which implies $GREEDY/4 \leq OPT$.

3 Min Sum Vertex Cover Problem

The Min Sum Vertex Cover (MSVC) problem is a special instance of the Min Sum Set Cover (MSSC) problem in which the given hyper-graph is a graph. We are given a graph $G(V, E)$ and the objective is to arrange the vertices of G in a linear order σ, such that the following sum is minimized, $\sum_{(u,v)\in E} \min\{\sigma(u), \sigma(v)\}$. We present a factor 1.79 approximation algorithm for MSVC.

3.1 Randomized Rounding Algorithm

We begin with the following LP relaxation of the problem. We will use $t \in [n]$ to index the positions in the ordering. Let $x_v(t)$ denote whether vertex v is present at position t and let $y_{uv}(t)$ depict if edge (u, v) is *not* covered by the vertices in the first t positions.

$$\begin{aligned} \text{Minimize} \quad & \sum_{(uv)\in E} \sum_{t=1}^{n} y_{uv}(t) \\ \text{subject to} \quad & 1 - \sum_{s \leq t} x_u(s) - \sum_{s \leq t} x_v(s) \leq y_{uv}(t) && (u,v) \in E,\ \forall t, && \text{(1a)} \\ & \sum_{s=1}^{n} x_u(s) + \sum_{s=1}^{n} x_v(s) \geq 1 && \forall (u,v) \in E, && \text{(1b)} \\ & x_u(t) \geq 0 && \forall u \in V,\ \forall t && \text{(1c)} \\ & y_{uv}(t) \geq 0 && \forall (u,v) \in E,\ \forall t. && \text{(1d)} \end{aligned}$$

The LP can be solved by standard means and let (x^*, y^*) be the optimal solution to the LP. Next, we will define a rounding scheme to round (x^*, y^*) to an integer solution by assigning every vertex a unique integer value (position) on the real line.

Overview of the Algorithm: Note that in the above linear program, if we (independently and) randomly round every vertex by interpreting $x_v^*(.)$ as a probability distribution, we are not even guaranteed a feasible solution. In [10], the authors fix this problem by scaling the solution x^* *prior* to rounding as follows. Let t_v be the largest value of t for which $\sum_{s<t} x_v(t) < 1/2$. For every vertex v, they introduce a new variable $z_v(t)$ where $z_v(t) = 2x_v(t)$ for $t < t_v$, and $z_v(t_v) = 1 - \sum_{t<t_v} z_v(t)$, and $z_v(t) = 0$ for $t > t_v$. By equation (1b), for every edge uv there exists $w \in \{u, v\}$ is such that $\sum_{s=1}^{n} x_w^*(s) \geq 1/2$, which implies $\sum_{s=1}^{n} z_w^*(s) = 1$. Therefore rounding independently, according to z, will surely yield a feasible solution. One can show that the expected value of the rounded solution is at most the optimal value of the objective.

The only shortcoming of the above rounding scheme is that owing to the scaling step, on an average 2 vertices can get rounded to the same position. Intuitively, this can be rectified by *stretching* the real-axis by a factor of 2 to accommodate the extra vertices as shown in Figure 3.1(a). This gives us one a 2 approximate algorithm.

We beat this factor by interleaving the rounding and stretching subroutines in multiple phases. In each phase our algorithm independently rounds on a subset of vertices and assigns them positions on the real-line. Then we stretch the real-line by a constant factor before starting the next phase.

Algorithm Description: Let us fix some notation that would be useful in describing the algorithm. The algorithm proceeds in several phases indexed by r and in each phase it assigns positions for some vertices on the real line. We will use $p(v)$ to denote position of vertex v. For any positive integer r, let $z_v^r(t) = z_v(t')$, if $t = \alpha^{(r-1)}t'$ for some constant $1 < \alpha \leq 2$ to be determined later; else, let $z_v^r(t) = 0$. That is, for $r > 1$, z^r is obtained from z^1 by stretching the real-line by a factor of α^{r-1}. Refer to Figure 3.1(b).

At the start of the algorithm assign a phase number β_v to every vertex v according to the distribution $\Pr[\beta_v = r] = 2^{-r}$ and let $S_r = \{v \mid \beta_v = r\}$. In the r-th phase all vertices in S_r are assigned positions as follows - for every vertex $v \in S_r$, randomly (and independently) assign v to position t with probability $z_v^r(t)$ i.e. set $p(v)$ to t. The algorithm terminates when all vertices are assigned a position on the real-line.

For position t, let $n_t = \sum_v p(v)$. To recover the ordering of vertices, replace position t by n_t time slots and allocate the vertices v for which $p(v) = t$ to these time slots in a random order.

3.2 Analysis

For the sake of conciseness, for any edge $uv \in E$ for a given execution of the algorithm, let us use Γ_{uv} to denote $|\{w \mid p(w) < \min\{p(u), p(v)\}\}| = \sum_{s<t} n_s$ i.e. Γ_{uv} is the number of vertices that are placed to the left of $argmin\{p(u), p(v)\}$. Thus the expected contribution of edge uv to the objective value is $\mathbb{E}[1 + \Gamma_{uv}]$. This can be further simplified using conditional expectation as shown below.

$$\begin{aligned}
\mathbb{E}[1+\Gamma_{uv}] &= \sum_t \Pr[\min\{p(u),p(v)\} = t] \times \mathbb{E}[1+\Gamma_{uv} | \min\{p(u),p(v)\} = t] \\
&= \sum_t t \Pr[\min\{p(u),p(v)\} = t] \times \frac{\mathbb{E}[1+\Gamma_{uv} \mid \min\{p(u),p(v)\} = t]}{t} \\
&\leq \sum_t t \Pr[\min\{p(u),p(v)\} = t] \times \max_t \left\{ \frac{\mathbb{E}[1+\Gamma_{uv} \mid \min\{p(u),p(v)\} = t]}{t} \right\} \\
&\leq \mathbb{E}[\min\{p(u),p(v)\}] \times \max_t \left\{ \frac{\mathbb{E}[1+\Gamma_{uv} \mid \min\{p(u),p(v)\} = t]}{t} \right\} \qquad (2a)
\end{aligned}$$

In Lemmas 1 and 2, we bound both quantities in (2a).

Lemma 1. *For any edge $(u, v) \in E$,*

$$\mathbb{E}[\min\{p(u),p(v)\}] \leq \frac{3}{4-\alpha} \sum_t y_{uv}(t)\,.$$

Proof. As a warm up let us calculate the expected value of $\min\{p(u), p(v)\}$ given that the edge is covered during the first phase. For any $u \in V$, define $F_u(t) = \sum_{s<t} z_u^1(s)$, i.e., $F_u(t)$ is the probability that u is placed at a position to the left of t given that $\beta_u = 1$.

$$\begin{aligned}
&\mathbb{E}[\min\{p(u),p(v)\} \mid (u,v) \text{ is covered in phase 1}] \\
&= \frac{\sum_t (1-F_u(t))}{3} + \frac{\sum_t (1-F_v(t))}{3} + \frac{\sum_t (1-F_u(t))(1-F_v(t))}{3} \qquad (3a) \\
&= 1/3 \sum_t 3 - 2F_u(t) - 2F_v(t) + F_u(t)F_v(t) \qquad (3b) \\
&\leq 1/3 \sum_t 3 - 2F_u(t) - 2F_v(t) + \frac{F_u(t)+F_v(t)}{2} \qquad (3c) \\
&= \sum_t 1 - \frac{F_u(t)+F_v(t)}{2} \qquad (3d)
\end{aligned}$$

The first term in (3a) corresponds to the case when $r_u = 1$ and $r_v > 1$; similarly the second term corresponds to the case when $r_v = 1$ and $r_u > 1$, and the third term is the case when both $r_u = 1$ and $r_v = 1$. We get (3c) from (3b) since both F_u and F_v are bounded by 1. Finally, since $\Pr[uv \text{ is covered in phase 1}] = 3/4$, we have $\Pr[\min\{p(u),p(v)\} \,\&\, (u,v) \text{ is covered in phase 1}] = 3/4 \sum_t 1 - \frac{F_u(t)+F_v(t)}{2}$.

Edge (u, v) is not covered by the start of the r-th phase if both $\beta_u \geq r$ and $\beta_v \geq r$. This happens with probability $2^{-2(r-1)}$. Also since by the start of the r-th phase z_u^1 and z_v^1 have been stretched by a factor of α^{r-1}, the expected position where edge (u, v) is covered, if it is covered in the r-th phase, is $\alpha^{r-1} \sum_t 1 - \frac{F_u(t)+F_v(t)}{2}$. Once again, as above, the probability that uv is covered in the r^{th} phase, given that it was not covered in the first $r-1$ phases, is $3/4$. Combining these two facts we get,

$$\begin{aligned}
\mathrm{E}\left[\min\left\{\bar{x}_u, \bar{x}_v\right\}\right] &= \sum_{r=1}^{\infty} 4^{-(r-1)} \frac{3\alpha^{r-1}}{4} \sum_t 1 - \frac{F_u(t) + F_v(t)}{2} \\
&= 3/4 \sum_{r=1}^{\infty} (\alpha/4)^{r-1} \sum_t \left\{ 1 - \sum_{s<t} \left(x_u(t) + x_v(t)\right) \right\} \\
&\leq 3/4 \sum_{r=0}^{\infty} (\alpha/4)^{r} \sum_t y_{uv}(t) \leq 3/4 \sum_t y_{uv}(t) \sum_{r=0}^{\infty} (\alpha/4)^{r} = \frac{3}{4-\alpha} \sum_t y_{uv}(t)\,.
\end{aligned}$$

The last equation follows from the linear programming constraint (1a).

Lemma 2. $\max_t \left\{ \dfrac{\mathbb{E}\left[1 + \Gamma_{uv} \mid \min\left\{p(u), p(v)\right\} = t\right]}{t} \right\} \leq 2\alpha/(2\alpha - 1)\,.$

Proof. For any position t,

$$\begin{aligned}
\mathbb{E}\left[1 + \Gamma_{uv} \mid \min\left\{p(u), p(v)\right\} = t\right] &= 1 + \sum_{r=1}^{\infty} \sum_{w \notin \{u,v\}} \sum_{s<t} \Pr\left[\beta_w = r\right] z_w^r(s) \\
&= 1 + \sum_{r=1}^{\infty} 2^{-r} \sum_{w \notin \{u,v\}} \sum_{s<t} z_w^r(s) \leq 1 + \sum_{r=1}^{\infty} 2^{-r} \frac{2(t-1)}{\alpha^{r-1}} \qquad (5a) \\
&\leq t \sum_{r=0}^{\infty} (2\alpha)^{-r} = 2t\alpha/(2\alpha - 1)\,. \qquad (5b)
\end{aligned}$$

The first part of (5a) follows from the distribution from which we choose β_w and the second part is derived from the definition of z^r. Finally (5b) holds for $\alpha < 2$ and dividing throughout by t gives the desired result.

Substituting the results from Lemmas 1 and 2 in to (2a) we find that for an arbitrary edge uv, the expected contribution to the objective is at most $\frac{6\alpha}{(4-\alpha)(2\alpha-1)} \sum_t y_{uv}(t)$. For $\alpha = \sqrt{2}$, this is approximately equal to $1.79 \sum_t y_{uv}(t)$. Summing over all edges and noting that $\sum_{(uv)\in E} \sum_t y_{uv}(t)$ is a lower bound on the optimal solution, we conclude that the above algorithm approximates min sum vertex cover to within a factor of at most 1.79.

4 Submodular Linear Ordering

4.1 Monotone Submodular Functions

In this section, we discuss the minimum linear ordering problem with f being a monotone submodular function. The algorithm is based on a continuous extension for the submodular function called the Lovász extension defined below.

Definition 1. *For a set function $f : 2^V \to \mathbb{R}$ with $f(\emptyset) = 0$, its extension $\hat{f} : {\mathbb{R}_+}^V \to \mathbb{R}$ is defined by $\hat{f}(x) = \sum_{i=1}^{n} \lambda_i f(S_i)$, where $V = S_1 \supsetneq S_2 \supsetneq \cdots \supsetneq S_n \supsetneq \emptyset$ is a chain such that $\sum \lambda_i 1_{S_i} = x$ and $\lambda_i \geq 0$.*

Alternatively, one can define $\hat{f}$ by $\hat{f}(x) = \mathbb{E}\left[f(\{i : x_i > \lambda\}\right]$, where λ is uniformly random in $[0, 1]$. Note that the value $\hat{f}(x)$ is easy to compute, provided that an oracle access to f is available. Lovász [16] showed that $\hat{f}$ is convex if and only if f is submodular.

Consider the following convex optimization problem, which can be solved in polynomial time by the ellipsoid method.

$$\begin{aligned} \langle \mathrm{CP} \rangle \qquad & \text{Minimize} \quad \hat{f}(x) \\ & \text{subject to} \quad \sum_{v \in S} x(v) \geq |S|(|S|+1)/2, \qquad \forall S \subseteq V. \end{aligned}$$

For a linear ordering σ, let x^σ denote a vector defined by $x^\sigma(v) = \sigma(v)$. Then x^σ is a feasible solution, and its objective value is $\hat{f}(x^\sigma) = \sum_{i=1}^{n} f(S_i)$. Thus $\langle \mathrm{CP} \rangle$ serves as a relaxation problem. Let x^* be an optimal solution of $\langle \mathrm{CP} \rangle$. We now consider a deterministic rounding procedure that returns a linear ordering σ so that $x^*(u) \leq x^*(v)$ implies $\sigma(u) \leq \sigma(v)$.

Lemma 3. *For each $v \in V$, we have $k \leq (2 - \frac{2}{k+1})x^*(v)$, where $k = \sigma(v)$.*

Proof. Consider the subset $S = \{u \mid \sigma(u) \leq k\}$. By the feasibility of x^*, we have $\sum_{v \in S} x^*(v) \geq k(k+1)/2$. Since $x^*(u) \leq x^*(v)$ for every $u \in S$, this implies $x^*(v) \geq (k+1)/2$. Hence we obtain $k \leq (2 - \frac{2}{k+1})x^*(v)$.

Theorem 5. *The algorithm constructs a linear ordering whose objective value is no more than $2 - \frac{2}{n+1}$ times the optimal one.*

Proof. Recall that $x^\sigma(v) = \sigma(v)$ for each $v \in V$. It follows from Lemma 3 that $x^\sigma(v) \leq \left(2 - \frac{2}{k+1}\right) x^*(v) \leq \left(2 - \frac{2}{n+1}\right) x^*(v)$, where $k = \sigma(v)$. Since $\hat{f}$ is monotone non-decreasing and $\hat{f}(\alpha x) = \alpha \hat{f}(x)$ holds for any $\alpha > 0$, this implies $\hat{f}(x^\sigma) \leq \left(2 - \frac{2}{n+1}\right) \hat{f}(x^*)$. Therefore, $\sum_{i=1}^{n} f(S_i)$ is at most $2 - \frac{2}{n+1}$ times the optimal value.

4.2 Symmetric Submodular Functions

We now focus on the linear ordering problem with f being a symmetric submodular function which includes the minimum linear arrangement problem over graphs. Given a graph $G(V, E)$ with edge capacity $c : E \to \mathbb{R}_+$, the minimum linear arrangement problem asks for finding a linear ordering $\sigma : V \to \{1, \ldots, n\}$ that minimizes $\sum_{(u,v) \in E} c(u, v)|\sigma(u) - \sigma(v)|$. Let $\kappa : 2^V \to \mathbb{R}_+$ denote the cut capacity function. For a linear ordering $\sigma : V \to \{1, \ldots, n\}$, we have

$$\sum_{i=1}^{n} \kappa(S_i) = \sum_{i=1}^{n} \sum_{(u,v) \in E,\, u \in S_i,\, v \notin S_i} c(u, v) = \sum_{(u,v) \in E} c(u, v)\, |\sigma(u) - \sigma(v)|.$$

Thus the minimum linear arrangement problem is a special case of MLOP with f being a cut function of a graph which is a symmetric submodular function.

Next, we show an unconditional information theoretic lower bound on the approximation factor for MLOP with symmetric submodular functions. This is done by defining two symmetric submodular functions f_1 and f_2 such that they achieve the same value on 'most' of the queries but have different optimal values.

Concretely, let $\delta > 0$ such that $\delta^2 = \frac{1}{n}\omega(\log n)$ and $\beta = \frac{n}{4}(1+\delta)$. Let R be a subset of V of size $\frac{n}{2}$ then for any $S \subseteq V$, define $f_1(S) = \min\left(|S|, \frac{n}{2}\right) - \frac{|S|}{2}$ and $f_2(S) = \min\left(|S|, \frac{n}{2}, \beta + |S \cap R|, \beta + |S \cap \bar{R}|\right) - \frac{|S|}{2}$.

It can be shown that both f_1 and f_2 are nonnegative and submodular, and using a result by Svitkina and Fleischer [19], we can bound the probability of distinguishing them using polynomially many value queries.

Lemma 4 ([19]). *For R chosen uniformly at random from among all subsets of V of size $\frac{n}{2}$, any algorithm that makes a polynomial number of oracle queries has probability at most $n^{\omega(1)}$ of distinguishing the functions f_1 and f_2.*

It can be shown the ratio of the optimal values of the linear arrangements under f_1 and f_2 is $2 - o(1)$, which coupled with Lemma 4, yields the following theorem. We defer the details of the proof to the full version of the paper.

Theorem 6. *For every constant $\epsilon > 0$, there exists a family of instances of the NM-MLOP such that no (computationally unbounded) algorithm making polynomially many queries to the cost function can achieve a factor better than $2 - \epsilon$.*

Acknowledgment. We thank Zoya Svitkina for suggesting Theorem 4.

References

1. Azar, Y., Gamzu, I.: Ranking with submodular valuations. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, pp. 1070–1079. Society for Industrial and Applied Mathematics, Philadelphia (2011)
2. Azar, Y., Gamzu, I., Yin, X.: Multiple intents re-ranking. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, pp. 669–678. ACM, New York (2009)
3. 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, SODA 2010, pp. 1539–1545. Society for Industrial and Applied Mathematics, Philadelphia (2010)
4. Bar-Yehuda, R., Even, G., Feldman, J., Naor, J.S.: Computing an optimal orientation of a balanced decomposition tree for linear arrangement problems. J. Graph Algorithms Appl. 5, 1–27 (2001)
5. Barenholz, U., Feige, U., Peleg, D.: Improved approximation for min-sum vertex cover (2006)
6. Charikar, M., Hajiaghayi, M.T., Karloff, H., Rao, S.: l22 spreading metrics for vertex ordering problems. Algorithmica 56(4), 577–604 (2010)

7. Chekuri, C., Ene, A.: Approximation algorithms for submodular multiway partition. In: Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, pp. 807–816. IEEE Computer Society, Washington, DC (2011)
8. Devanur, N.R., Khot, S.A., Saket, R., Vishnoi, N.K.: Integrality gaps for sparsest cut and minimum linear arrangement problems. In: Proceedings of the 38th Annual ACM Symposium on Theory of Computing, STOC 2006, pp. 537–546. ACM, New York (2006)
9. Feige, U., Lee, J.R.: An improved approximation ratio for the minimum linear arrangement problem. Inf. Process. Lett. 101, 26–29 (2007)
10. Feige, U., Lovász, L., Tetali, P.: Approximating min sum set cover. Algorithmica 40, 219–234 (2004)
11. Goel, G., Karande, C., Tripathi, P., Wang, L.: Approximability of combinatorial problems with multi-agent submodular cost functions. In: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, pp. 755–764. IEEE Computer Society, Washington, DC (2009)
12. Hassin, R., Levin, A.: An Approximation Algorithm for the Minimum Latency Set Cover Problem. In: Brodal, G.S., Leonardi, S. (eds.) ESA 2005. LNCS, vol. 3669, pp. 726–733. Springer, Heidelberg (2005)
13. Horton, S.B.: The Optimal Linear Arrangement Problem: Algorithms and Approximation. PhD thesis, Georgia Institute of Technology, Atlanta, GA, USA (1997), AAI9735901
14. Im, S., Nagarajan, V., van der Zwaan, R.: Minimum latency submodular cover (manuscript, 2012)
15. Iwata, S., Nagano, K.: Submodular function minimization under covering constraints. In: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, pp. 671–680. IEEE Computer Society, Washington, DC (2009)
16. Lovász, L.: Submodular functions and convexity. In: Mathematical Programming — The State of the Art, pp. 235–257. Springer (1983)
17. Petit, J.: Experiments on the minimum linear arrangement problem. J. Exp. Algorithmics 8 (December 2003)
18. Rao, S., Richa, A.W.: New approximation techniques for some ordering problems. In: Proceedings of the Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 1998, pp. 211–218. Society for Industrial and Applied Mathematics, Philadelphia (1998)
19. Svitkina, Z., Fleischer, L.: Submodular approximation: Sampling-based algorithms and lower bounds. SIAM J. Comput. 40(6), 1715–1737 (2011)

New Approximation Results for Resource Replication Problems

Samir Khuller[1,⋆], Barna Saha[2], and Kanthi K. Sarpatwar[1,⋆⋆]

[1] Department of Computer Science
University of Maryland (College Park)
{samir,kasarpa}@cs.umd.edu
[2] AT&T Shannon Research Laboratory
barna@research.att.com

Abstract. We consider several variants of a basic resource replication problem in this paper, and propose new approximation results for them. These problems are of fundamental interest in the areas of P2P networks, sensor networks and ad hoc networks, where optimal placement of replicas is the main bottleneck on performance. We observe that the threshold graph technique, which has been applied to several k-center type problems, yields simple and efficient approximation algorithms for resource replication problems. Our results range from positive (efficient, small constant factor, approximation algorithms) to extremely negative (impossibility of existence of any algorithm with non-trivial approximation guarantee, i.e., with positive approximation ratio) for different versions of the problem.

1 Introduction

Problems related to data placement and replication are of fundamental interest both in the area of large scale distributed networking systems as well as centralized storage systems. The performance of distributed systems such as P2P file sharing systems, wireless ad hoc networks, sensor networks etc., where resources are shared among clients, can be significantly impacted by placement of the replicated resources [16,17,2]. On the other hand, centralized storage systems, such as in netflix, might have data distributed across different data centers so that it is necessary to keep data closer to the demand to prevent over loading the network. Demand patterns for data can also vary widely, especially in the context of video on demand distribution.

There is a lot of research on centralized storage systems [9] that addresses the problem of data layout when all the storage units are centrally located in a single location and thus the "distance" of each client from any storage unit is the same. However, in modern storage management systems, this assumption is

⋆ Supported by NSF Awards CCF-0728839 and CCF-0937865, and a Google Research Award.
⋆⋆ Supported by NSF Grant CCF-0728839.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 218–230, 2012.
© Springer-Verlag Berlin Heidelberg 2012

not valid. Companies rent storage space all over the world from different data centers in different locations. Since the most interesting objective functions are NP-hard, it is of interest to consider efficient approximation algorithms.

The basic framework is the following: given a collection of k data items, we wish to distribute the k data items to a collection of n nodes modeled by a graph, where the vertices are embedded in a metric space. In the basic model, each node wishes to access each of the k data items and the goal is to minimize the *maximum* distance any node has to travel to access all k items. For this problem, Ko and Rubenstein [16] give a distributed algorithm based on a local search idea and also show that this algorithm delivers a solution with a worst case approximation guarantee of 3. We note that the algorithm is not guaranteed to run in polynomial time, however, in practice its convergence was reasonably quick. In a followup piece of work [17], Ko and Rubenstein introduced a generalization of the basic problem in which each node only required a *subset* of the items. For this problem, they develop a heuristic, however, for this heuristic, unlike the other case, there is no approximation guarantee any more.

In this paper we consider both the questions described above, along with several other generalizations and provide polynomial time approximation algorithms for them. In particular we develop a simple algorithm with a 3-approximation for the basic model, and this can be implemented in a distributed setting. We also develop a more involved centralized 3-approximation scheme for the general problem as well. However, we do not know how to implement this algorithm in a distributed setting as yet. In addition, we consider further generalizations where we need to provide excellent service to a given fraction of the clients and not all the clients. This is motivated from the fact that there may be a few outliers, and it may be extremely costly to provide all data items to the outliers. Here, the two problems deviate in difficulty immediately. For the basic problem we can still provide a constant approximation, but for the general problem, somewhat surprisingly, it turns out that, assuming $P \neq NP$, there is no polynomial time algorithm with any non-trivial approximation guarantee. We give a polynomial time reduction of the *densest k-subgraph* [8] to the feasibility version of the general problem.

Following the works of Ko and Rubenstein [16,17], in this paper, we consider the "min-max" objective function for data placement problems. A different objective function of minimizing average data-access cost was studied by Baev et al. [1,2] under the assumption that each client only requires a particular data item. A generalization of this problem with load and capacity constraints on servers was considered by Guha et al. [11] and Meyerson et al. [20] (called the *page-placement* problem). They developed bicriteria approximation algorithms for this problem where load and/or capacity are violated by a small factor.

Our Contributions. The following is a summary of our results.

– In Section 2, we consider the basic replication problem where each client needs all k data items (*basic resource replication*) and its generalization where each client might need a subset of data items (*subset resource replication*). For the first problem, we give a distributed polynomial time

3-approximation algorithm and show that there does not exist any polynomial time algorithm achieving a $2-\epsilon$ (for any $\epsilon > 0$) approximation (Theorem 1 and Theorem 3). For the later, we give the first polynomial time 3-approximation algorithm (in a centralized setting) along with matching hardness (Theorem 2 and Theorem 3).
- In Section 3, we consider the outlier version of the basic as well as subset resource replication problem. For the former, we give a polynomial time 3-approximation algorithm while for the latter, somewhat surprisingly, we show that there does not exist any non-trivial approximation guarantee (in polynomial time). We also consider the case where each resource can be replicated at most K times and give polynomial time 5-approximation algorithm for it.
- In Section 4, we consider another natural generalization of the basic resource replication problem where each node has an upper bound (load) on the number of clients it can serve. We give polynomial time 4-approximation algorithm for this version when load $L \geq 2k-1$ (k is the number of resources). A simple counting argument shows that this problem is infeasible if $L < k$. This implies our 4-approximation algorithm is a bicriteria approximation algorithm and the load capacity is not violated by more than a factor of 2.

2 Resource Replication Problem

2.1 Basic Resource Replication Problem

The following problem, which we call the *Basic Resource Replication* (BRR) problem, was first studied by Ko and Rubenstein [16]. The input consists of:

- set of nodes or vertices, $V = \{v_1, v_2 \dots v_n\}$
- a metric space defined by the function $d : V \times V \to \mathbb{R}^+ \cup \{0\}$
- set of resources or colors $\mathcal{C} = \{C_1, C_2, C_3, \dots, C_k\}$.

We seek to find an *optimal* mapping $\phi : V \to \mathcal{C}$ of colors to vertices. The objective function for optimality is defined in the following way. Define $d_r(v)$ to be the shortest distance between a vertex assigned the color C_r[1] and the vertex v. The goal of the *Basic Resource Replication* (BRR) problem is the following -

$$\min_{\phi} \max_{\substack{v \in V \\ C_r \in \mathcal{C}}} d_r(v).$$

This is the central problem of the work of Ko and Rubenstein [16] who give a distributed algorithm with a 3-approximation guarantee. Unfortunately, their algorithm has no proven polynomial running time bound. We give a simple distributed polynomial time 3-approximation algorithm for this problem. All the algorithms in this work use a technique called *threshold graph construction*

[1] We may abuse the notation and use same expression, $d_r(v)$, when r represents a color.

introduced by Edmonds and Fulkerson [7] and used extensively for k-center type problems [10,15,14,13]. We observe that the use of this approach enables the design of very simple and efficient algorithms for several resource replication problems. Given $\delta \in \mathbb{R}^+ \cup \{0\}$, the *threshold graph*, denoted by G_δ, is constructed by adding edges between every pair of vertices u, v which are at distance at most δ. The algorithm for BRR works in the following way. For each vertex v, we determine the distance of the $(k-1)^{th}$ closest neighbor - and denote by δ_L the maximum of these distances. We construct the threshold graph G_{δ_L} which has minimum degree at least $k-1$. Also δ_L must be a lower bound on the optimal δ (δ_{OPT}) - because δ_L is the least value such that the threshold graph has degree at least $k-1$ and $G_{\delta_{OPT}}$ has minimum degree at least $k-1$. Now in the graph $G^2_{\delta_L}$ which is the graph formed by squaring G_{δ_L}, we compute a maximal independent set $\mathcal{I}$. Finally, for each vertex in $\mathcal{I}$, we color the vertex with C_1 and pick $k-1$ vertices from its list of neighbors in G_δ and assign them a distinct color from the remaining $k-1$ colors. Due to space restrictions, we defer the details of the algorithm and discussion of this problem (along with a few other generalizations) to full version[2].

Theorem 1. *There is a distributed, polynomial time, 3-approximation algorithm for the problem of BRR.*

2.2 Subset Resource Replication Problem

In BRR model each client requires all the data items. But in general each client might be interested in a subset of resources instead of all the resources. The servers might also have capacity to hold several data items. This substantially more generalized version of resource replication problem, which we call *Subset Resource Replication* Problem (SRR) was considered by Ko and Rubenstein in a subsequent paper [17]. Formally, the problem has the following input

- a set of vertices $V = \{v_1, v_2 \dots v_n\}$, a metric $d : V \times V \to \mathbb{R}^+ \cup \{0\}$ and a set of colors $\mathcal{C} = \{C_1, C_2 \dots C_k\}$.
- every vertex $v \in V$ has a subset $\mathcal{C}_v \subseteq \mathcal{C}$ of "required" colors and a non-negative integer s_v as the storage capacity - that is we can assign s_v colors to vertex v.

The goal is to assign a list of colors $\phi(v) \subseteq \mathcal{C}$ to each vertex v, such that $|\phi(v)| \leq s_v$, with the following objective -

$$\delta = \min_{\phi} \max_{\substack{v \in V \\ r \in \mathcal{C}_v}} d_r(v)$$

where $d_r(v)$ is the shortest distance from v to a vertex having C_r on its list of colors. Ko and Rubenstein [17] extended their basic approach to this problem but had no guarantee on either the approximation ratio or the running time. We give the first centralized polynomial time 3-approximation algorithm (Algorithm 1)

[2] http://www.cs.umd.edu/~samir/grant/approx12-full.pdf

for the problem. Later, in Theorem 3, we will prove that this is the best possible approximation one can expect, assuming $P \neq NP$.

We again use the threshold graph technique. The optimal distance δ has to be the distance between one of the $O(n^2)$ pairs of vertices. Hence, it has only polynomial number of possible values and we can assume that the value of δ is known (trying out all possible values of δ will only add a polynomial factor). Assuming δ is known, we construct the threshold graph G_δ. We now square the graph G_δ to obtain G_δ^2, i.e., add an edge between two vertices $u, v \in V$ if they are at a distance at most two in G_δ. Consider a color r and let $H_r \subseteq G_\delta^2$ be the induced subgraph on vertices that need color r (among possibly other colors). Let $\mathcal{I}_r$ be a maximal independent set in the subgraph H_r. The following is a key observation about an optimal solution.

Observation. *For every vertex $v \in \mathcal{I}_r$, the optimal solution must assign a unique copy of r in the neighborhood of v in G_δ.* (†)

Indeed, in G_δ the neighborhoods corresponding to vertices in $\mathcal{I}_r$ must be mutually disjoint. If neighborhoods corresponding to vertices $u, v \in \mathcal{I}_r$ intersect, then there must exist an edge between u, v in G_δ^2 - which is impossible, as $\mathcal{I}_r$ forms an independent set in this graph. Since, every vertex must be satisfied by some copy in its neighborhood in G_δ, our observation holds. If for every vertex $v \in \mathcal{I}_r$, $d_r(v) \leq \delta$ then every vertex $u \in H_r$ has $d_r(u) \leq 3 \times \delta$. Thus to find a 3-approximation, we focus on satisfying vertices of such independent sets $\mathcal{I}_r$, for each color $r \in \mathcal{C}$. We cast this as a b-matching problem [6] on the graph $B = (X, Y)$ - where X is the union of independent sets $\mathcal{I}_r$, $\forall r \in \mathcal{C}$ (i.e., if a vertex belongs to s independent sets of the form I_r, we add s copies of the vertex to X) and Y is a copy of V with $b-$matching bounds s_v on each vertex $v \in V$. We add an edge across the partitions, if its end points are at distance at most δ from each other. From observation (†), there must exist a b-matching that saturates all the vertices of X.

Algorithm 1. A 3-approximation algorithm for SRR

1: Guess the optimal value δ. Construct the graph G_δ^2
2: **for all** colors c **do**
3: Let H_c be the subgraph of G_δ^2 induced by the set of vertices that require color c.
4: Compute $\mathcal{I}_c$, any maximal independent set of H_c.
5: **end for**
6: Let X denote the union of copies of each $\mathcal{I}_c$ (i.e., if a vertex is contained in s independent sets of form $\mathcal{I}_c$, we add s copies of that vertex to X). Let Y be a copy of set of vertices in V with non-zero storage capacities.
7: Construct the bipartite graph $B = (X, Y)$: add an edge between $x \in X$ and $y \in Y$ if the nodes they represent are at distance at most δ.
8: Compute a maximum b-matching in B with bounds : 1 on vertices of X and respective storage capacities on the nodes of Y.
9: For every node $v \in Y$, let $S_v \subseteq X$ be matched subset of nodes, assign the list of colors L_v of nodes of S_v to v.

Theorem 2. *Algorithm 1 is a 3-approximation for the Subset Resource Replication problem.*

Proof. We start by proving that (assuming δ is the optimal solution), the maximum b-matching, found in step 7 of Algorithm 1, completely saturates X. It is sufficient to show that there exists of b-matching which saturates X (which implies the maximum matching also does so). In the optimal coloring, which satisfies every vertex within distance δ, let L_v^{opt} denote the list of colors placed on $v \in V$ (for feasibility, $|L_v^{opt}| \leq s_v$, where s_v is the storage capacity of v). For a color i and a vertex v, we denote the copy of v in $\mathcal{I}_i$ by v_i. We note that for every v requiring a color i, there exists a vertex $u \in Y$ which is within distance δ of v and has i in its list of colors L_u^{opt}. We now claim that the following edge set forms a b-matching which saturates X. The edge set, denoted by bM, consists one edge for each $v_i \in X$, namely $\overline{v_i u}$, where u is some vertex within distance δ of v_i such that $i \in L_u^{opt}$. We only have to show that bM is a feasible b-matching, because it saturates X by its definition.

In order to prove that bM is a feasible $b-$matching, we show that the number of edges incident on each vertex is within the allocated bounds - s_v for $v \in Y$ and 1 for $v_i \in X$. The latter bound is trivially verified. To prove that the bounds s_v are not violated, we observe that no two vertices of X with same color index i, say v_i and w_i, are matched to the same vertex $u \in Y$ with respect to bM. Indeed, this would imply that v and w are adjacent in G_δ^2, which is a contradiction to the fact that they belong to a maximal independent set (in some induced subgraph of G_δ^2). Thus the number of edges of bM incident on u, is at most $|L_u^{opt}| \leq s_u$. Hence, bM is a valid b-matching which saturates all the vertices of X.
To finish the proof, we now show that every node v requiring a color i finds a node hosting i at distance at most 3δ. Indeed, there exists some $u_i \in X$, such that u is a neighbor of v in H_i (note that the distance between such u and v is at most 2δ). Now, if $\overline{u_i w} \in bM$, w is the vertex hosting i at distance at most 3δ. Hence, Algorithm 1 is a 3-approximation algorithm for the subset resource replication problem.

2.3 Hardness of BRR and SRR

We now prove some lower bounds on the above problems. The following theorem shows that Algorithm 1 provides the best possible guarantee for the SRR problem, while there is a small gap between the algorithm and the lower bound proven for the BRR problem. We state the theorem here; for lack of space, the proof is given in full version.

Theorem 3. *Assuming $P \neq NP$, for any given constant $\epsilon > 0$, there is no polynomial time algorithm which guarantees an approximation ratio better than*

- $(2 - \epsilon)$ *for basic resource replication problem.*
- $(3 - \epsilon)$ *for subset resource replication problem.*

3 Robust Resource Replication Problem

The objective of minimizing the maximum distance over all vertices may result in a much larger distance if there are few distant "outliers". Even a good approximation algorithm, in this case, will raise δ to a very high value and many nodes could get a bad solution. It is therefore natural to study outlier version of such problems. In such a model, the objective remains the same but we are allowed to ignore a few far away vertices (the outliers). Several well known problems have been studied under the "outlier" model like outlier versions of k-center problem [5] (called *robust k-centers*), scheduling with outliers [4,12,21], outlier versions of facility location type problems [5,19]. In this section, we initiate the problem of *robust basic resource replication* (RBRR) or the resource replication problem with outliers. In the RBRR problem, the input is the same as the BRR problem along with a lower bound M - which is the number of vertices that have to be satisfied. Formally, the input instance $\mathcal{I} = (V, \mathcal{C}, M, d)$ is defined as following.

- A set of vertices $V = \{v_1, v_2, \ldots v_n\}$, a metric $d : V \times V \to \mathbb{R}^+ \cup \{0\}$ and a set of colors $\mathcal{C} = \{C_1, C_2 \ldots C_k\}$.
- A lower bound $M \in \mathbb{N}$.

The objective function of the Robust Basic Resource Replication problem is defined as-

$$\min_{\substack{\phi \\ S \subseteq V \\ |S| \geq M}} \quad \max_{v \in S} \quad \max_{C_r \in \mathcal{C}} \quad d_r(v)$$

A simple extension to the BRR algorithm results in a 3-approximation algorithm for this problem. Due to space restrictions, we defer our discussion to full version. Instead, we focus on a more interesting generalization of the Robust Basic Resource Replication problem called the K-Robust Basic Resource Replication (K-RBRR) problem. In this problem we only allow K copies of each resource, while the rest of input and output structure remains the same as RBRR. This problem is a natural generalization of the robust K-center problem- the former problem has k resources and latter has only one. The robust K-center problem is the outlier version of K-center problem and was studied, along with several other outlier variants of facility location type problems by Charikar et al. [5]. One variant of particular interest to our work is the robust K-supplier problem, for which [5] gives a 3 -approximation algorithm. The robust K-supplier is the outlier variant of K-supplier problem. In the K-supplier problem, we have a set of suppliers and a set of clients, embedded in a metric. The goal is to choose K suppliers which can hold a resource (there is only one resource here) such that the maximum "client to nearest resource distance" is minimized over all clients. In the robust K-supplier problem, we have the same objective but we may satisfy only M clients. We use the 3-approximation algorithm of [5] as a sub-routine and obtain a 5-approximation algorithm for K-RBRR problem. For the sake of completeness, we briefly describe the algorithm from [5] here.
For a given value δ, the algorithm of [5] proceeds in the following way.

- For each supplier v, construct G_v as the set of clients within distance δ and E_v as the set of clients within distance 3δ of v.
- Repeat the following steps k times:
  - Greedily pick a supplier v as a center whose set G_v covers most number of yet uncovered clients. (†)
  - Mark all the clients in E_v as covered.
- If at least M vertices are not satisfied return NO, or else return the set of centers.

For a proof on why this algorithm guarantees a 3-approximation, we refer the reader to [5]. We make a small modification to the above algorithm before using it as a sub-routine. In the step (†), if there are no more clients to be covered we can stop (this will clearly not affect the performance or feasibility of the algorithm). Otherwise, there is at least one new uncovered client which is now covered by v. We pick one such newly covered client arbitrarily and label it $U(v)$. Note that this process assigns a distinct client to each supplier.

We can now describe our Algorithm 2 to solve the K-RBRR problem. We make the following claims about Algorithm 2 but defer the proofs to full version.

Claim. If δ is optimal distance value for an instance of K-RBRR, it is a feasible distance for the K-supplier instance in the step 2 of Algorithm 2.

Claim. The set $\mathcal{I}$ formed in the step 3 of Algorithm 2 is an independent set in G_δ^2.

Algorithm 2. A 5-approximation for K-RBRR

1: Guess optimal distance value δ and construct G_δ. Mark the nodes of degree $\geq k-1$. Let these "high" degree vertices form a set V_c.
2: With V_c as the set of clients, $V_s = V$ as the set of suppliers, distance between copies remaining the same as the original vertices, we solve the robust K-supplier problem [5] with δ as the input distance. Let $S \subseteq V_s$ be the set of centers returned. By Claim 3, S is well defined.
3: Let $\mathcal{I} = \{U(v) : v \in S\}$. By Claim 3, $\mathcal{I}$ is an independent set such that each member has degree $\geq k-1$ in G_δ.
4: **for** $v \in \mathcal{I}$ **do**
5: Pick $k-1$ neighbors of v in G_δ. Assign each of these vertices along with v, one color each of the k colors.
6: **end for**

Theorem 4. *Algorithm 2 is a 5-approximation for the K-RBRR.*

Proof. We defer the proof to full version.

Let us now consider the Robust Subset Resource Replication (RSRR) problem. In this problem, we are provided with the input for the SRR problem along with a lower bound M on the number of vertices that must be satisfied with their requirement. The objective function is -

$$\min_{\phi} \max_{\substack{S \subset V \\ |S| \geq M}} \max_{\substack{v \in S \\ r \in \mathcal{C}_v}} d_r(v)$$

Given that the outlier version of BRR and its extension with bound on each color has simple constant factor approximation algorithms, it is a natural question to ask whether similar bounds can be obtained for Robust SRR. But, quite surprisingly, we show not only there does not exist any constant factor approximation algorithm for Robust SRR, but in fact, assuming $P \neq NP$, there is no polynomial time algorithm that provides any nontrivial approximation guarantee. In Theorem 5, we prove that deciding if a given instance of RSRR is feasible, is NP hard. We give a polynomial time reduction of the well-studied densest k subgraph [8] problem to the problem of deciding the feasibility of RSRR. In the decision version of the densest k-subgraph problem, we have an instance of the form $\mathcal{I} = (G, k, L)$ and the goal is to decide if there is a subgraph of G with exactly k vertices and at least L edges.

Theorem 5. *Assuming $P \neq NP$, there is no polynomial time algorithm which gives a positive approximation ratio for Robust Subset Resource Replication problem.*

Proof. **Reduction.** Given an instance of densest k-subgraph problem $\mathcal{I} = (G = (V, E), k, L)$, $|V| = n$, $|E| = m$ where the problem is to decide if there is a subgraph on k vertices with at least L edges - we construct an instance of RSRR, $\mathcal{I}' = (G', M, \mathcal{C}, \{\mathcal{C}_v : \forall v \in G'\})$ as follows. First, color the vertices in V with distinct colors $c_1, c_2 \ldots c_n$. The vertex set of G' has 3 parts - V_1, V_2, V_3. V_1 has k vertices and V_2 has m vertices corresponding to the edges of G. The distance between any two vertices $u \in V_1, v \in V_2$ is 1. Each vertex $v \in V_2$ has a set of m^2 vertices, G_v, associated with itself. The distance between any vertex pair of $v \cup G_v$ is 1. Rest of the distances are computed using the shortest path metric. The set $\{\mathcal{C}_v : \forall v \in G'\}$ is specified in the following way - Each vertex $u \in V_1$ requires 0 colors and hence are trivially satisfied. Each vertex $v \in V_2$ requires colors $\{a_v, c_i, c_j\}$ where a_v is a color associated uniquely with vertex v and c_i, c_j are the colors of the end points of the edge in G associated with v. Each vertex $w \in G_v$ requires colors $\{a_v, b_v^i : i \in [1 : m^2]\}$. Each one of $a_v, b_v^i : v \in V_2, i \in [1, m^2]$ is a distinct color. Set $M = m^3 + L + k$, the lower bound of the number of vertices that must be satisfied.

Claim: I is an YES instance of densest k subgraph problem if and only if I' is a feasible solution of Robust Subset Resource Replication problem. In other words, we prove that the feasibility question of Robust Subset Resource Replication problem is NP-hard. This would imply that there is no approximation algorithm for this problem.

Proof of the Claim. Let I be an YES instance of the densest subgraph problem and let $H = \{v_1, v_2 \ldots v_k\}$ be the k vertices that induce L edges in G. We present a feasible coloring for I' as following -

- The k vertices of V_1 are colored with the k colors of H
- Each vertex $v \in V_2$ is colored with its associated color a_v.
- For each vertex $v \in V_2$, its m^2 associated vertices G_v are colored with m^2 colors of type b_v^i.

It is straightforward to check that the above coloring satisfies $M = m^3 + k + L$ vertices - all the vertices of V_3 are satisfied, all the vertices of V_1 are satisfied and at least L vertices of V_2 are satisfied. Now, we consider the other direction. Let there be a coloring of vertices of G' which certifies that I' is a feasible instance. We first observe that, all the m^3 colors of type b_v^i and the m colors of type a_v must be used - otherwise, there will be at least m^2 vertices out of $m^3 + m + k$ vertices which go unsatisfied and hence the bound M is not met. Since, we are only interested in the feasibility question, we can assume that m^2 vertices of G_v are colored with m^2 colors of type b_v^i and the m vertices $v \in V_2$ are colored with color a_v. Now at least L vertices of V_2 must be satisfied and the k vertices of V_1 must be colored with k colors from $\{c_1, c_2 \ldots c_n\}$ - say $\{c_1, c_2 \ldots c_k\}$. We observe that the union of colors required by the L vertices, apart from their associated colors, must be $\{c_1, c_2 \ldots c_k\}$. Hence, the L edges in G corresponding to these L vertices in V_2 must be completely incident on the vertices in V corresponding to these k colors. This implies the existence of k vertices in G that induce L edges. Hence the theorem.

4 Capacitated Basic Resource Replication Problem

Another desired quality of an assignment scheme in client-server type problems is load balancing [18,15,3]. In this setting, we are not allowed to "overload" a server by assigning more than a bounded number of clients. Bar-Ilan, Kortsarz and Peleg [3], Khuller and Sussman [15] study the load balancing version of the k-center problem which is called the *capacitated k-center problem*. Khuller and Sussman [15] provide the current best approximation ratio of 5 for this problem. We initiate the study of basic resource replication problem in the load balancing setting. We call it the *capacitated basic resource replication problem* (CBRR). In this problem, the input instance is defined as $\mathcal{I} = (V, \mathcal{C} = \{C_1, C_2 \ldots C_k\}, d, L)$ and the goal is the same as the basic resource replication problem with an additional restriction that a vertex with a certain color is not allowed to serve more than L other vertices (including itself). We give a 4-approximation algorithm (Algorithm 3) for this problem, provided $L \geq 2k - 1$. We prove in the full version that, for a feasible solution, L has to be $\geq k$. By using this fact, we observe that Algorithm 3 is in fact a bicriteria approximation algorithm - it gives an approximation guarantee of 4 while exceeding the load by a factor of 2 at most.

Algorithm 3 starts by guessing the optimal δ and constructs the threshold graph G_δ. Let $\mathcal{I}$ be some maximal independent set of G_δ^2. We divide all the vertices into three levels - *level* 0, *level* 1 and *level* 2. All the elements in $\mathcal{I}$ are at *level* 0. All vertices not in $\mathcal{I}$ but adjacent (with respect to G_δ) to some element in $\mathcal{I}$ are at *level* 1. Finally all the vertices not in *level* 0 or *level* 1 are in *level* 2.

For each element v at *level* 0, its empire $Empire(v)$ consists of itself along with all the adjacent(with respect to G_δ) *level* 1 vertices. Since $\mathcal{I}$ is independent in G_δ^2, all the empires defined so far are mutually disjoint. Finally, all the *level* 2 vertices are adjacent to at least one *level* 1 vertex. For each *level* 2 vertex, we pick one such *level* 1 vertex arbitrarily and assign the former to the same empire as the latter. Thus we have assigned every vertex to exactly one empire.

In the next step, we consider one empire at a time and split it into "blocks" of vertices. Every block consists of exactly k vertices, except the last block which might have less than k vertices. A key property of vertices in a block is the following - any two vertices are at a distance of at most 4δ from each other. We now color each block of size exactly k using all k colors (since the degree of each vertex is at least $k-1$ in G_δ, every empire has at least one block of size exactly k). A vertex in a block only serves other vertices in the same block, hence the load is not more than k currently on any vertex. The vertices of the final block (which might have $\leq k$ vertices) are now served by some block of size exactly size k. Thus the load on each vertex is at most $2k-1$.

Algorithm 3. A 4-approximation for CBRR

Guess the optimal value δ. Construct the graph G_δ and G_δ^2.
Let $\mathcal{I}$ be a maximal independent set in G_δ^2.
for all $v \in V$ **do**
  if $v \in \mathcal{I}$ **then**
    $Empire(v) = \{v\}$
  end if
  if $v \notin \mathcal{I}$ **then**
    if v has a vertex $u \in \mathcal{I}$ at distance δ. **then**
      Such a vertex is unique owing to the property that $\mathcal{I}$ is an independent set. Add v to the empire of u, $Empire(u) = Empire(u) \cup \{v\}$.
    else if v has a vertex in $\mathcal{I}$ at distance 2δ. **then**
      Pick one such vertex u arbitrarily and add v to the empire of u.
    end if
  end if
end for
for all $v \in \mathcal{I}$ **do**
  Each vertex v has degree at least $k-1$ in G_δ. Hence, $|Empire(v)| \geq k$. Divide $Empire(v)$ into blocks, all of which have size exactly k - except possibly the last one which has size at most k.
  Color each block of size exactly k using k colors, arbitrarily. The final block, whose size is at most k, has its color requirement satisfied from one such block. Since there is at least one block of size exactly k, such an assignment is valid.
end for

Theorem 6. *Algorithm 3 is a 4-approximation algorithm for the problem of Capacitated Basic Resource Replication problem where the allowed load* $L \geq 2k-1$.

Proof. We defer the proof to full version.

To conclude, we study several variants of the resource replication problem and prove that most of them are approximable within a small constant. A striking anomaly is the problem of RSRR, which somewhat surprisingly is hard to approximate within any non-trivial bound. Our work leaves several open problems. It would be interesting to close the gap between the approximation factor and the lower bound of the BRR problem. Extending the capacitated version to SRR, obtaining a true approximation factor for CBRR for all values of load, improving the approximation factor for K-RBRR etc. are few other future directions to consider.

References

1. Baev, I.D., Rajaraman, R.: Approximation algorithms for data placement in arbitrary networks. In: SODA, pp. 661–670 (2001)
2. Baev, I.D., Rajaraman, R., Swamy, C.: Approximation algorithms for data placement problems. SIAM J. Comput. 38(4), 1411–1429 (2008)
3. Bar-Ilan, J., Kortsarz, G., Peleg, D.: How to allocate network centers. J. Algorithms 15(3), 385–415 (1993)
4. Charikar, M., Khuller, S.: A robust maximum completion time measure for scheduling. In: SODA, pp. 324–333 (2006)
5. Charikar, M., Khuller, S., Mount, D.M., Narasimhan, G.: Algorithms for facility location problems with outliers. In: SODA, pp. 642–651 (2001)
6. Edmonds, J.: Paths, trees, and flowers. In: Gessel, I., Rota, G.-C. (eds.) Classic Papers in Combinatorics, Modern Birkhuser Classics, pp. 361–379. Birkhuser, Boston (1987)
7. Edmonds, J., Fulkerson, D.R.: Bottleneck extrema. Journal of Combinatorial Theory 8(3), 299–306 (1970)
8. Feige, U., Peleg, D., Kortsarz, G.: The dense k-subgraph problem. Algorithmica 29(3), 410–421 (2001)
9. Golubchik, L., Khanna, S., Khuller, S., Thurimella, R., Zhu, A.: Approximation algorithms for data placement on parallel disks. ACM Transactions on Algorithms 5(4) (2009)
10. Gonzalez, T.F.: Clustering to minimize the maximum intercluster distance. Theor. Comput. Sci. 38, 293–306 (1985)
11. Guha, S., Munagala, K.: Improved algorithms for the data placement problem. In: SODA, pp. 106–107 (2002)
12. Gupta, A., Krishnaswamy, R., Kumar, A., Segev, D.: Scheduling with Outliers. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 149–162. Springer, Heidelberg (2009)
13. Hochbaum, D.S., Shmoys, D.B.: A best possible heuristic for the k-center problem. Mathematics of Operations Research 10(2), 180–184 (1985)
14. Hochbaum, D.S., Shmoys, D.B.: A unified approach to approximation algorithms for bottleneck problems. J. ACM 33(3), 533–550 (1986)
15. Khuller, S., Sussmann, Y.J.: The capacitated k-center problem. SIAM J. Discrete Math. 13(3), 403–418 (2000)
16. Ko, B.-J., Rubenstein, D.: Distributed, self-stabilizing placement of replicated resources in emerging networks. In: ICNP, pp. 6–15 (2003)
17. Ko, B.-J., Rubenstein, D.: Distributed server replication in large scale networks. In: NOSSDAV, pp. 127–132 (2004)

18. Korupolu, M.R., Greg Plaxton, C., Rajaraman, R.: Analysis of a local search heuristic for facility location problems. In: SODA, pp. 1–10 (1998)
19. Krishnaswamy, R., Kumar, A., Nagarajan, V., Sabharwal, Y., Saha, B.: The matroid median problem. In: SODA, pp. 1117–1130 (2011)
20. Meyerson, A., Munagala, K., Plotkin, S.A.: Web caching using access statistics. In: SODA, pp. 354–363 (2001)
21. Saha, B., Srinivasan, A.: A new approximation technique for resource-allocation problems. In: ICS, pp. 342–357 (2010)

Maximum Matching in Semi-streaming with Few Passes*

Christian Konrad[1,2], Frédéric Magniez[1], and Claire Mathieu[3]

[1] LIAFA, Université Paris Diderot, Paris, France
[2] LRI, Université Paris-Sud, Orsay, France
[3] DI/ENS CNRS, Paris, France and CS Dept, Brown University, Providence
konrad@lri.fr, frederic.magniez@univ-paris-diderot.fr, claire@cs.brown.edu

Abstract. We present three semi-streaming algorithms for Maximum Bipartite Matching with one and two passes. Our one-pass semi-streaming algorithm is deterministic and returns a matching of size at least $1/2 + 0.005$ times the optimal matching size in expectation, assuming that edges arrive one by one in (uniform) random order. Our first two-pass algorithm is randomized and returns a matching of size at least $1/2+0.019$ times the optimal matching size in expectation (over its internal random coin flips) for any arrival order. These two algorithms apply the simple Greedy matching algorithm several times on carefully chosen subgraphs as a subroutine. Furthermore, we present a two-pass deterministic algorithm for any arrival order returning a matching of size at least $1/2 + 0.019$ times the optimal matching size. This algorithm is built on ideas from the computation of semi-matchings.

1 Introduction

Streaming. Classical algorithms assume random access to the input. This is a reasonable assumption until one is faced with massive data sets as in bioinformatics for genome decoding, Web databases for the search of documents, or network monitoring. The input may then be too large to fit into the computer's memory. Streaming algorithms sequentially scan the input piece by piece in one pass, while using sublinear memory space. The analysis of Internet traffic [1] was one of the first applications of such algorithms. A similar but slightly different situation arises when the input is recorded on an external storage device where only sequential access is possible, such as optical disks, or even hard drives. Then a small number of passes, ideally constant, can be performed.

By sublinear memory one ideally means memory that is polylogarithmic in the size of the input. Nonetheless, polylogarithmic memory is often too restrictive for graph problems: as shown in [2], deciding basic graph properties such as bipartiteness or connectivity of an n-vertex graph requires $\Omega(n)$ space. Muthukrishnan [3] initially mentioned massive graphs as typical examples where one

* Supported by the French ANR Defis program under contract ANR-08-EMER-012 (QRAC project). Christian Konrad is supported by a Fondation CFM-JP Aguilar grant. Claire Mathieu is supported by NSF grant CCF-0964037.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 231–242, 2012.
© Springer-Verlag Berlin Heidelberg 2012

assumes a *semi-external* model, that is, not the entire graph but the vertex set can be stored in memory. In that model, a graph is given by a stream of edges arriving in arbitrary order. A *semi-streaming* algorithm has memory space $O(n \operatorname{polylog} n)$, and the graph vertices are known in advance.

Matching. In this paper we focus on an iconic graph problem: finding large matchings. In the semi-streaming model, the problem was primarily addressed by Feigenbaum, Kannan, McGregor, Suri and Zhang [4]. In the meantime a variety of semi-streaming matching algorithms for particular settings exist (unweighted/weighted, bipartite/general graphs). Most works, however, consider the multipass scenario [5,6] where the goal is to find a $(1-\epsilon)$ approximation while minimizing the number of passes. The techniques are based on finding augmenting paths, and, recently, linear programming was also applied. Ahn and Guha [5] provide an overview of the current best algorithms. In this paper, we also take the augmenting paths route.

In the one-pass setting, in the unweighted case, the greedy matching algorithm is still the best known algorithm. (We note that in the weighted case, progress was made [7] [8], but when the edges are unweighted those algorithms are of no help.) The greedy matching constructs a matching in the following online fashion: starting with an empty matching M, upon arrival of edge e, it adds e to M if $M+e$ remains a matching. A *maximal matching* is a matching that can not be enlarged by adding another edge to it. It is well-known that the cardinality of maximal matchings is at least half of the cardinality of maximum matchings. By construction, since the greedy matching is maximal, M is a $1/2$-approximation of any maximum matching M^*, that is $|M| \geq |M^*|/2$. The starting point of this paper was to address the following long standing open problem: **Is the greedy algorithm best possible, or is it possible to get an approximation ratio better than** $1/2$**?**

A very recent result [9] rules out the possibility of any one-pass semi streaming algorithm for Maximum Bipartite Matching (MBM) with approximation ratio better than $2/3$, since that would require memory space $n^{1+\Omega(1/\log\log n)}$. Nevertheless, there is still room between $1/2$ and $2/3$. To get an approximation ratio better than $1/2$, prior semi-streaming algorithms require at least 3 passes, for instance the algorithm of [6] can be used to run in 3 passes providing a matching strictly better than a $1/2$ approximation.

Random Order of Edge Arrivals. The behavior of the greedy matching algorithm has been studied in a variety of settings. The most relevant reference [10] considers a (uniform) random order of edge arrivals. Here, Dyer and Frieze showed that the expected approximation ratio is still $1/2$ for some graphs (their example can be extended to bipartite graphs), but can be better for particular graph classes such as planar graphs and forests.

In the context of streaming algorithms, random order arrival has been first studied for the problems of sorting and selecting in limited space by Munro and Paterson [11]. Then Guha and McGregor [12] gave an exponential separation between random and adversarial order models. One justification of the random order model is to understand why some problems do not admit a memory efficient streaming algorithms in theory, while in practice, heuristics are often sufficient.

Other Related Work. MBM was studied in the online setting, where nodes from one side arrive in adversarial order together with all their incident edges. The well-known randomized algorithm by Karp and Vazirani [13] achieves an optimal approximation ratio of $(1-1/e)$ for bipartite graphs where all nodes from one side are known in advance. This barrier was broken recently by assuming that, although the graph is worst-case, the arrival order is random according to some distribution [14,15]. Estimating the *size* of maximum matchings was further studied in the sublinear time model. For graphs with degrees bounded by d, it is possible to estimate the size of a maximum matching up to an additive ϵ in time independent of n [16,17]. The algorithm partially explores balls of limited radius around some vertices, an approach that cannot be adapted to one-pass streaming. Furthermore, estimations with one-sided error require linear time.

Our Results. In this paper we present algorithms for settings in which we can beat 1/2. We design semi-streaming algorithms for MBM with one and two passes. Our one-pass semi-streaming algorithm is deterministic and achieves an expected approximation ratio $1/2+0.005$ for any graph (**Theorem 1**) assuming that the edges arrive one by one in (uniform) random order. This is the first analysis of a graph algorithm in the semi-streaming model assuming random order arrival of edges. Our two two-pass semi-streaming algorithm do not need the random order assumption. We present a randomized two-pass algorithm with expected approximation ratio $1/2+0.019$ against its internal random coin flips, for any graph and for any arrival order (**Theorem 3**). Furthermore, we present a deterministic counterpart with the same approximation ratio for any graph and any arrival order (**Theorem 4**).

Techniques. The one-pass algorithm as well as the randomized two-pass algorithm each apply three times the greedy matching algorithm on different and carefully chosen subgraphs. The deterministic two-pass algorithm is slightly more complicated as it uses a subroutine that computes a particular semi-matching besides the greedy algorithm.

General idea: If we had three passes at our disposal (see for instance Algorithm 2 in [4]), we could use one pass to build a maximal matching M_0 between the two sides A and B of the bipartition, a second pass to find a matching M_1 between the A vertices matched in M_0 and the B vertices that are free w.r.t. M_0 whose combination with edges of M_0 forms paths of length 2. Finally, a third pass to find a matching M_2 between B vertices vertices matched in M_0 and A vertices that are free w.r.t. M_0 whose combination with M_0 and M_1 forms paths of length 3 that can be used to augment matching M_0. All our algorithms simulate these 3 passes in less passes.

One pass, deterministic, random order: To simulate this with a single pass, we split the sequence of arrivals $[1, m]$ into three phases $[1, \alpha m]$, $(\alpha m, \beta m]$, and $(\beta m, m]$ and build M_0 during the first phase, M_1 during the second phase, and M_2 during the third phase. Of course, we see only a subset of the edges for each phase, but thanks to the random order arrival, these subsets are random, and, intuitively, we loose only a constant fraction in the sizes of the constructed

matchings. As it turns out, the intuition can be made rigorous, as long as the first matching M_0 is maximal or close to maximal. We observe that, if the greedy algorithm, executed on the entire sequence of edges, produces a matching that is not much better than a $1/2$ approximation of the optimal maximum matching, then that matching *is built early on.* More precisely (Lemma 4), if the greedy matching on the entire graphs is no better than a $1/2 + \epsilon$ approximation, then after seeing a mere one third of the edges of the graph, the greedy matching is already a $1/2 - \epsilon$ approximation, so it is already close to maximal.

Two passes, randomized, any order: Assume a bipartite graph (A, B, E) comprising a perfect matching. If A' is a small random subset of A, then, regardless of the arrival order, the greedy algorithm that constructs a greedy matching between A' and B (that is, the greedy algorithm restricted to the edges that have an endpoint in A') will find a matching that is near-perfect, that is, almost every vertex of A' is matched (see Theorem 2 for a slightly more general version of this statement). This property of the greedy algorithm may be of independent interest. Then, in one pass we compute a greedy matching M_0 and also via the greedy algorithm independently and in parallel a matching M_1 between a subset $A' \subset A$ and the B vertices. It turns out that $M_0 \cup M_1$ comprise many length 2 paths that can be completed to 3-augmenting paths by a third matching M_2 that we compute in the second pass.

Two passes, deterministic, any order: Again, assume a bipartite graph (A, B, E) comprising a perfect matching and some integer λ. Add now greedily edges ab to a set S if the degree of a in S is yet 0, and the degree of b is smaller than λ. This algorithm computes an *incomplete semi-matching* with a degree limitation λ on the B nodes. In the first pass, we run this algorithm in parallel to the greedy matching algorithm for constructing M_0. S replaces the computation of M_1. We will show that many length 2 paths in $M_0 \cup S$ can be completed to 3-augmenting paths in the second pass via a further greedy matching M_2.

Extension to General Graphs. All algorithms presented in this paper can be generalized to non-bipartite graphs. These generalizations, however, require more technically involved analyses while the main ideas are already captured in the bipartite versions. For this reason, and for the sake of a clean presentation, we only present here the bipartite versions while the algorithms for general graphs are postponed to the full version of this paper.

When searching for augmenting paths in general graphs, algorithms have to cope with the fact that a candidate edge for an augmenting path may form an undesired triangle with the edge to augment and an optimal edge. In this case, the candidate edge can block the entire augmenting path. McGregor [18] overcomes this problem by repeatedly sampling bipartite graphs from the input graph. Such a strategy is not needed for our randomized algorithms. Indeed, one can show that undesired triangles only appear with small probability allowing our techniques to still work. For our deterministic two-pass algorithm, a direct combinatorial argument can be used to bound the number of those *bad* triangles.

2 Preliminaries

Notations and Definitions. Let $G = (A, B, E)$ be a bipartite graph with $V = A \cup B$, $n = |V|$ vertices and $m = |E|$ edges. For $e \in E$ with end points $a \in A$ and $b \in B$, we denote e also by ab. The input G is given as an edge sequence arriving one by one in some order. Let $\Pi(G)$ be the set of all edge sequences of G.

Definition 1 (Semi-Streaming Algorithm). *A k-pass* semi-streaming algorithm $\boldsymbol{A}$ *with processing time per edge t is an algorithm such that, for every input stream $\pi \in \Pi(G)$ encoding a graph G with n vertices: (1) $\boldsymbol{A}$ performs in total at most k passes on stream π, (2) $\boldsymbol{A}$ maintains a memory space of size $O(n \operatorname{polylog} n)$, (3) $\boldsymbol{A}$ has running time $O(t)$ per edge.*

For a subset of edges F, we denote by $\operatorname{opt}(F)$ a matching of maximum size in the graph G restricted to edges F. We may write $\operatorname{opt}(G)$ for $\operatorname{opt}(E)$, and we use $M^* = \operatorname{opt}(G)$. We say that an algorithm $\mathbf{A}$ computes a *c-approximation of the maximum matching* if $\mathbf{A}$ outputs a matching M such that $|M| \geq c \cdot |\operatorname{opt}(G)|$. We consider two potential sources of randomness: from the algorithm and from the arrival order. Nevertheless, we will always consider worst case against the graph. For each situation, we relax the notion of c-approximation so that the expected approximation ratio is c, that is $\mathbb{E}\,|M| \geq c \cdot |\operatorname{opt}(G)|$ where the expectation can be taken either over the internal random coins of the algorithm, or over all possible arrival orders.

For simplicity, we assume that A, B and $m = |E|$ are given in advance to our semi-streaming algorithms. Moreover, for two sets S_1, S_2 we denote by $S_1 \oplus S_2$ the symmetric difference $(S_1 \setminus S_2) \cup (S_2 \setminus S_1)$ of the two sets.

For an input stream $\pi \in \Pi(G)$, we write $\pi[i]$ for the i-th edge of π, and $\pi[i, j]$ for the subsequence $\pi[i]\pi[i+1] \ldots \pi[j]$. In this notation, a parenthesis excludes the smallest or respectively largest element: $\pi(i, j] = \pi[i+1, j]$, and $\pi[i, j) = \pi[i, j-1]$. If i, j are real, $\pi[i, j] := \pi[\lceil i \rceil, \lfloor j \rfloor]$, and $\pi[i] := \pi[\lceil i \rceil]$. Given a subset $S \subseteq V$, $\pi|_S$ is the largest subsequence of π such that all edges in $\pi|_S$ are among vertices in S.

For a set of vertices S and a set of edges F, let $S(F)$ be the subset of vertices of S covered by F. Furthermore, we use the abbreviation $\overline{S(F)} := S \setminus S(F)$. For $S_A \subseteq A$ and $S_B \subseteq B$, we write $\operatorname{opt}(S_A, S_B)$ for $\operatorname{opt}(G|_{S_A \times S_B})$, that is a maximum matching in the subgraph of G induced by vertices $S_A \cup S_B$.

Maximal Matchings and the Greedy Matching Algorithm. Formally, the *greedy matching algorithm* Greedy *on stream* π is defined as follows: Starting with an empty matching M, upon arrival of an edge $\pi[i]$, Greedy inserts $\pi[i]$ into M if $\pi[i]$ does not intersect any edges in M, that is, if $V(M) \cap \{\pi[i]\} = \emptyset$. Denote by $\text{Greedy}(\pi)$ the matching M after the stream π has been fully processed. By maximality, $|\text{Greedy}(\pi)| \geq \frac{1}{2}|\operatorname{opt}(G)|$. Greedy can be seen as a semi-streaming algorithm for MBM with expected approximation ratio $\frac{1}{2}$ and $O(1)$ processing time per edge. We now state some preliminary properties (proofs omitted). Lemma 1 shows that a maximal matching that is far from the optimal matching in value is also far from the optimal matching in Hamming distance.

Lemma 1. *Let $M^* = \text{opt}(G)$, and let M be a maximal matching of G. Then $|M \cap M^*| \leq 2(|M| - \frac{1}{2}|M^*|)$.*

Lemma 2 shows that maximal matchings that are small in size contain many edges that are 3-*augmentable*. Given a maximum matching $M^* = \text{opt}(G)$, and a maximal matching M, we say that an edge $e \in M$ is 3-augmentable if the removal of e from M allows the insertion of two edges f, g from $M^* \setminus M$ into M.

Lemma 2. *Let $\epsilon \geq 0$. Let $M^* = \text{opt}(G)$, let M be a maximal matching of G st. $|M| \leq (\frac{1}{2} + \epsilon)|M^*|$. Then M contains at least $(\frac{1}{2} - 3\epsilon)|M^*|$ 3-augmentable edges.*

3 One-Pass Algorithm on Random Order

Algorithm. Here is a key observation in the random order setting: if Greedy performs badly on some input graph G, then most edges of Greedy appear within the first constant fraction of the stream, see Lemma 4. Our strategy is hence to run Greedy on a first part of the stream, and then, on the remaining part of the stream, we focus on searching for 3-augmenting paths.

Let M_0 denote the matching computed by Greedy on the first part of the stream. Assume that Greedy performs badly on the input graph G. Lemma 2 tells us that almost all of the edges of M_0 are 3-augmentable. To find 3-augmenting paths, in the next part of the stream we run Greedy to compute a matching M_1 between $B(M_0)$ and $\overline{A(M_0)}$. The edges in M_1 serve as one of the edges of 3-augmenting paths (from the B-side of M_0). In Lemma 5, we show that we find a constant fraction of those. In the last part of the stream, again by the help of Greedy, we compute a matching M_2 that completes the 3-augmenting paths. Lemma 8 shows that by this strategy we find many 3-augmenting paths. Then, either a simple Greedy matching performs well on G, or else we can find many 3-augmenting paths and use them to improve M_0: see the main theorem, Theorem 1 whose proof is deferred to the end of this section.

Algorithm 1. One-pass deterministic bipartite matching algorithm

1: $\alpha \leftarrow 0.4312, \beta \leftarrow 0.7595$
2: $M_G \leftarrow \text{Greedy}(\pi)$
3: $M_0 \leftarrow \text{Greedy}(\pi[1, \alpha m])$, matching obtained by running Greedy on the first $\lfloor \alpha m \rfloor$ edges
4: $F_1 \leftarrow$ complete bipartite graph between $B(M_0)$ and $\overline{A(M_0)}$
5: $M_1 \leftarrow \text{Greedy}(F_1 \cap \pi(\alpha m, \beta m])$, matching obtained by running Greedy on edges $\lfloor \alpha m \rfloor + 1$ to βm that intersect F_1
6: $A' \leftarrow \{a \in A \mid \exists b \in B(M_1) : ab \in M_0\}$
7: $F_2 \leftarrow$ complete bipartite graph between A' and $\overline{B(M_0)}$
8: $M_2 \leftarrow \text{Greedy}(F_2 \cap \pi(\beta m, m])$, matching obtained by running Greedy on edges $\lfloor \beta m \rfloor + 1$ to m that intersect F_2
9: $M \leftarrow$ matching obtained from M_0 augmented by $M_1 \cup M_2$
10: **return** larger of the two matchings M_G and M

Our algorithm only uses memory space $O(n \log n)$. Indeed, the subsets F_1 and F_2 can be compactly represented by two n-bit arrays, and checking if an edge of π belongs to one of them can be done in time $O(1)$ via that representation.

Theorem 1. *Algorithm 1 is a deterministic one-pass semi-streaming algorithm for* MBM *with approximation ratio* $\frac{1}{2} + 0.005$ *against (uniform) random order for any graph, and can be implemented with* $O(1)$ *processing time per edge.*

Analysis. We use the notations of Algorithm 1. Consider α and β as variables with $0 \leq \alpha \leq \frac{1}{2} < \beta < 1$.

Lemma 3. $\forall e = ab \in E : \Pr[a \text{ and } b \notin V(M_0)] \leq (\frac{1}{\alpha} - 1) \Pr[e \in M_0]$.

Proof. Observe: $\Pr[a \text{ and } b \notin V(M_0)] + \Pr[e \in M_0] = \Pr[a \text{ and } b \notin V(M_0 \setminus \{e\})]$, because the two events on the left hand side are disjoint and their union is the event on the right hand side.

Consider the following probabilistic argument. Take the execution for a particular ordering π. Assume that a and $b \notin V(M_0 \setminus \{e\})$ and let t be the arrival time of e. If we modify the ordering by changing the arrival time of e to some time $t' \leq t$, then we still have a and $b \notin V(M_0 \setminus \{e\})$. Thus[1]

$$\Pr[a \text{ and } b \notin V(M_0 \setminus \{e\})] \leq \Pr[a \text{ and } b \notin V(M_0 \setminus \{e\}) | e \in \pi[1, \alpha m]].$$

Now, the right-hand side equals $\Pr[e \in M_0 | e \in \pi[1, \alpha m]]$, which simplifies into $\Pr[e \in M_0] / \Pr[e \in \pi[1, \alpha m]]$ since e can only be in M_0 if it is one of the first αm arrivals. The we conclude the Lemma by the random order assumption $\Pr[e \in \pi[1, \alpha m]] = \alpha$. □

Lemma 4. *If* $\mathbb{E}_\pi |M_G| \leq (\frac{1}{2} + \epsilon)|M^*|$, *then* $\mathbb{E}_\pi |M_0| \geq |M^*|(\frac{1}{2} - (\frac{1}{\alpha} - 2)\epsilon)$.

Proof. Rather than directly analyzing the number of edges $|M_0|$, we analyze the number of vertices matched by M_0, which is equivalent since $|V(M_0)| = 2(|M_0|)$.

Fix an edge $e = ab$ of M^*. Either $e \in M_0$, or at least one of a, b is matched by M_0, or neither a nor b are matched. Summing over all $e \in M^*$ gives

$$|V(M_0)| \geq 2|M^* \cap M_0| + |M^* \setminus M_0| - \sum_{e=ab \in M^*} \chi[a \text{ and } b \notin V(M_0)],$$

where $\chi[X] = 1$ if the event X happens, otherwise $\chi[X] = 0$. Taking expectations and using Lemma 3,

$$\begin{aligned} \mathop{\mathbb{E}}_{\pi}(|V(M_0)|) &\geq 2 \mathop{\mathbb{E}}_{\pi} |M^* \cap M_0| + \mathop{\mathbb{E}}_{\pi} |M^* \setminus M_0| - (\frac{1}{\alpha} - 1) \mathop{\mathbb{E}}_{\pi} |M^* \cap M_0| \\ &= |M^*| - (\frac{1}{\alpha} - 2) \mathop{\mathbb{E}}_{\pi} |M^* \cap M_0|. \end{aligned}$$

[1] Formally, we define a map f from the uniform distribution on all orderings to the uniform distribution on all orderings such that $e \in \pi[1, \alpha m]$: if $e \in \pi[1, \alpha m]$ then $f(\pi) = \pi$ and otherwise $f(\pi)$ is the permutation obtained from π by removing e and re-inserting it at a position picked uniformly at random in $[1, \alpha m]$.

Since M_0 is just a subset of the edges of M_G, using Lemma 1 and linearity of expectation, $\mathbb{E}_\pi |M^* \cap M_0| \leq \mathbb{E}_\pi |M^* \cap M_G| \leq 2(\mathbb{E}_\pi |M_G| - \frac{1}{2}|M^*|) \leq 2\epsilon|M^*|$. Combining gives the Lemma. □

Lemma 5. *Assume that* $\mathbb{E}_\pi |M_G| \leq (\frac{1}{2} + \epsilon)|M^*|$. *Then:*

$$\mathbb{E}_\pi |\mathrm{opt}(\overline{A(M_0)}, B(M_0))| \geq |M^*|(\tfrac{1}{2} - (\tfrac{1}{\alpha} + 2)\epsilon).$$

Proof. The size of a maximum matching between $\overline{A(M_0)}$ and $B(M_0)$ is at least the number of augmenting paths of length 3 in $M_0 \oplus M^*$. By Lemma 2, in expectation, the number of augmenting paths of length 3 in $M_G \oplus M^*$ is at least $(\frac{1}{2} - 3\epsilon)|M^*|$. All of those are augmenting paths of length 3 in $M_0 \oplus M^*$, except for at most $|M_G| - |M_0|$. Hence, in expectation, M_0 contains $(\frac{1}{2} - 3\epsilon)|M^*| - (\mathbb{E}_\pi |M_G| - \mathbb{E}_\pi |M_0|)$ 3-augmentable edges. Lemma 4 concludes the proof. □

Lemma 6. $\mathbb{E}_\pi |M_1| \geq \frac{1}{2}(\beta - \alpha)(\mathbb{E}_\pi |\mathrm{opt}(\overline{A(M_0)}, B(M_0))| - \frac{1}{1-\alpha})$.

Proof. Since Greedy computes a maximal matching which is at least half the size of a maximum matching, $\mathbb{E}_\pi |M_1| \geq \frac{1}{2}\mathbb{E}_\pi |\mathrm{opt}(\overline{A(M_0)}, B(M_0)) \cap \pi(\alpha m, \beta m]|$.

By independence between M_0 and the ordering within $(\alpha m, m]$, we see that even if we condition on M_0, we still have that $\pi(\alpha m, \beta m]$ is a random uniform subset of $\pi(\alpha m, m]$. Thus:

$$\mathop{\mathbb{E}}_{\pi} |\mathrm{opt}(\overline{A(M_0)}, B(M_0)) \cap \pi(\alpha m, \beta m]| = \tfrac{\beta-\alpha}{1-\alpha} \mathop{\mathbb{E}}_{\pi} |\mathrm{opt}(\overline{A(M_0)}, B(M_0)) \cap \pi(\alpha m, m]|.$$

We use a probabilistic argument similar to but slightly more complicated than the proof of Lemma 3. We define a map f from the uniform distribution on all orderings to the uniform distribution on all orderings such that $e \in \pi(\alpha m, m]$: if $e \in \pi(\alpha m, m]$ then $f(\pi) = \pi$ and otherwise $f(\pi)$ is the permutation obtained from π by removing e and re-inserting it at a position picked uniformly at random in $(\alpha m, m]$; in the latter case, if this causes an edge $f = a'b'$, previously arriving at time $\lfloor \alpha m \rfloor + 1$, to now arrive at time $\lfloor \alpha m \rfloor$ and to be added to M_0, we define $M_0' = M_0 \setminus \{f\}$; in all other cases we define $M_0' = M_0$. Thus, if in π we have $e \in \mathrm{opt}(\overline{A(M_0)}, B(M_0))$, then in $f(\pi)$ we have $e \in \mathrm{opt}(\overline{A(M_0')}, B(M_0'))$. Since the distribution of $f(\pi)$ is uniform conditioned on $e \in \pi(\alpha m, m]$:

$$\frac{\Pr[e \in \mathrm{opt}(\overline{A(M_0')}, B(M_0')) \text{ and } e \in \pi(\alpha m, m]]}{\Pr[e \in \pi(\alpha m, m]]} \geq \Pr[e \in \mathrm{opt}(\overline{A(M_0)}, B(M_0))],$$

Using $\Pr[e \in \pi(\alpha m, m]] = 1 - \alpha$ and summing over e:

$$\mathbb{E}_\pi |\mathrm{opt}(\overline{A(M_0')}, B(M_0')) \cap \pi(\alpha m, m]| \geq (1 - \alpha)\, \mathbb{E}_\pi |\mathrm{opt}(\overline{A(M_0)}, B(M_0))|.$$

Since M_0' and M_0 differ by at most one edge, $|\mathrm{opt}(\overline{A(M_0)}, B(M_0))| \geq |\mathrm{opt}(\overline{A(M_0')}, B(M_0'))| - 1$, and the Lemma follows. □

Lemma 7. *If* $\mathop{\mathbb{E}}_{\pi} |M_G| \leq (\frac{1}{2} + \epsilon)|M^*|$, *then* $\mathop{\mathbb{E}}_{\pi} |\mathrm{opt}(A', \overline{B(M_0)}| \geq \mathop{\mathbb{E}}_{\pi} |M_1| - 4\epsilon|M^*|$.

Proof. $|\mathrm{opt}(A', \overline{B(M_0)}|$ is at least $|M_1|$ minus the number of edges of M_0 that are not 3-augmentable. Since M_0 is a subset of M_G, the latter term is bounded

by the number of edges of M_G that are not 3-augmentable, which by Lemma 2 is in expectation at most $(\frac{1}{2}+\epsilon)|M^*| - (\frac{1}{2}-3\epsilon)|M^*| = 4\epsilon|M^*|$. □

Lemma 8. $\mathbb{E}_\pi |M_2| \geq \frac{1}{2}((1-\beta)\,\mathbb{E}_\pi |\mathrm{opt}(A', \overline{B(M_0)})| - 1)$.

The proof of Lemma 8 has a similar flavor as the proofs of Lemmas 3 and 6 and it uses a similar probabilistic argument (proof omitted). We now present the proof of the main theorem, Theorem 1.

Proof (of Theorem 1). Assume that $\mathbb{E}_\pi |M_G| \leq (\frac{1}{2}+\epsilon)|M^*|$. By construction, every $e \in M_2$ completes a 3−augmenting path, hence $|M| \geq |M_0| + |M_2|$. In Lemma 4 we show that $\mathbb{E}_\pi |M_0| \geq |M^*|(\frac{1}{2} - (\frac{1}{\alpha} - 2)\epsilon)$. By Lemmas 8 and 7, $|M_2|$ can be related to $|M_1|$:

$$\mathop{\mathbb{E}}_{\pi} |M_2| \geq \frac{1}{2}(1-\beta) \mathop{\mathbb{E}}_{\pi} |\mathrm{opt}(A', \overline{B(M_0)})| - \frac{1}{2} \geq \frac{1}{2}(1-\beta)(\mathop{\mathbb{E}}_{\pi} |M_1| - 4\epsilon|M^*|) - \frac{1}{2}.$$

By Lemmas 6 and 5, $|M_1|$ can be related to $|M^*|$:

$$\begin{aligned} \mathop{\mathbb{E}}_{\pi} |M_1| &\geq \tfrac{1}{2}(\beta-\alpha) \mathop{\mathbb{E}}_{\pi} |\mathrm{opt}(\overline{A(M_0)}, B(M_0)| - O(1) \\ &\geq \tfrac{1}{2}(\beta-\alpha)(|M^*|(\frac{1}{2} - (\frac{1}{\alpha}+2)\epsilon)) - O(1). \end{aligned}$$

Combining,
$\mathop{\mathbb{E}}_{\pi} |M| \geq |M^*|(\frac{1}{2} - (\frac{1}{\alpha}-2)\epsilon + \frac{1}{2}(1-\beta)(\frac{1}{2}(\beta-\alpha)(\frac{1}{2} - (\frac{1}{\alpha}+2)\epsilon) - 4\epsilon)) - O(1)$.
The expected value of the output of the Algorithm is at least $\min_\epsilon \max\{(\frac{1}{2}+\epsilon)|M^*|, \mathbb{E}_\pi |M|\}$. We set the right hand side of the above Equation equal to $(\frac{1}{2}+\epsilon)|M^*|$. By a numerical search we optimize parameters α, β. Setting $\alpha = 0.4312$ and $\beta = 0.7595$, we obtain $\epsilon \approx 0.005$ which proves the Theorem. □

4 Randomized Two-Pass Algorithm on Any Order

The algorithm relies on a property of the Greedy algorithm that we discuss before the presentation of the algorithm. This property may be of independent interest.

Matching Many Vertices of a Random Subset of A. For a fixed parameter $0 < p \leq 1$, consider an independent random sample of vertices $A' \subseteq A$ such that $\Pr[a \in A'] = p$, for all $a \in A$. Theorem 2 (proof omitted) shows that the greedy algorithm restricted to the edges with an endpoint in A' will output a matching of expected approximation ratio $p/(1+p)$, compared to a maximum matching $\mathrm{opt}(G)$ over the full graph G. Since, in expectation, the size of A' is $p|A|$, one can roughly say that a fraction of $1/(1+p)$ of vertices in $|A'|$ has been matched.

Theorem 2. *Let $0 < p \leq 1$, let $G = (A, B, E)$ be a bipartite graph. Let A' be an independent random sample $A' \subset A$ such that $\Pr[a \in A'] = p$, for all $a \in A$. Let F be the complete bipartite graph between A' and B Then for any input stream $\pi \in \Pi(G)$: $\mathbb{E}_{A'} |\mathrm{Greedy}(F \cap \pi)| \geq \frac{p}{1+p}|\mathrm{opt}(G)|$.*

Application: a Randomized Two-Pass Algorithm. Based on Theorem 2, we design our randomized two-pass algorithm. Assume that Greedy(π) returns a matching that is close to a $\frac{1}{2}$ approximation. In order to apply Theorem 2, we pick an independent random sample $A' \subseteq A$ such that $\Pr[a \in A'] = p$ for all a. In a first pass, our algorithm computes a Greedy matching M_0 of G, and a Greedy matching M' between vertices of A' and B. M' then contains some edges that form parts of 3-augmenting paths for M_0.Let $M_1 \subset M'$ be the set of those edges. It remains to complete these length 2 paths $M_0 \cup M_1$ in a second pass by a further Greedy matching M_2. Theorem 3 states then that if Greedy(π) is close to a $\frac{1}{2}$ approximation, then we find many 3-augmenting paths.

Algorithm 2. Two-pass randomized bipartite matching algorithm

1: Let $p \leftarrow \sqrt{2} - 1$.
2: Take an independent random sample $A' \subseteq A$ st. $\Pr[a \in A'] = p$, for all $a \in A$
3: Let F_1 be the set of edges with one endpoint in A'.
4: **First pass:** $M_0 \leftarrow$ Greedy(π) and $M' \leftarrow$ Greedy($F_1 \cap \pi$)
5: $M_1 \leftarrow \{e \in M' \,|\, e$ goes between $B(M_0)$ and $\overline{A(M_0)}\}$
6: $A_2 \leftarrow \{a \in A(M_0) : \exists b, c : ab \in M_0$ and $bc \in M_1\}$.
7: Let $F_2 \leftarrow \{da : d \in \overline{B(M_0)}$ and $a \in A_2\}$.
8: **Second pass:** $M_2 \leftarrow$ Greedy($F_2 \cap \pi$)
9: Augment M_0 by edges in M_1 and M_2 and store it in M
10: **return** the resulting matching M

Theorem 3. *Algorithm 2 is a randomized two-pass semi-streaming algorithm for* MBM *with expected approximation ratio* $\frac{1}{2} + 0.019$ *in expectation over its internal random coin flips for any graph and any arrival order, and can be implemented with* $O(1)$ *processing time per edge.*

Proof. By construction, each edge in M_2 is part of a 3-augmenting path, hence the output has size: $|M| = |M_0| + |M_2|$. Define ϵ to be such that $|M_0| = (\frac{1}{2} + \epsilon)|\text{opt}(G)|$. Since M_2 is a maximal matching of F_2, we have $|M_2| \geq \frac{1}{2}|\text{opt}(F_2)|$. Let M^* be a maximum matching of G. Then $|\text{opt}(F_2)|$ is greater than or equal to the number of edges ab of M_0 such that there exists an edge bc of M_1 and an edge da of M^* that altogether form a 3-augmenting path of M_0:

$$\begin{aligned}|\text{opt}(F_2)| &\geq |\{ab \in M_0 \,|\, \exists c : bc \in M_1 \text{ and } \exists d : da \in M^*\}| \\ &\geq |\{ab \in M_0 \,|\, \exists c : bc \in M_1\}| - |\{ab \in M_0 \,|\, ab \text{ not 3-augmentable}\}|.\end{aligned}$$

Lemma 2 gives $|\{ab \in M_0 \,|\, ab$ is not 3-augmentable with $M^*\}| \leq 4\epsilon|\text{opt}(G)|$. It remains to bound $|\{ab \in M_0 \,|\, \exists c : bc \in M_1\}|$ from below. By definition of M' and of $M_1 \subseteq M'$, and by maximality of M_0,

$$\begin{aligned}|\{ab \in M_0 \,|\, \exists c : bc \in M_1\}| &= |M'| - |\{ab \in M' \,|\, a \in A(M_0)\}| \\ &\geq |M'| - |A(M_0) \cap A'|.\end{aligned}$$

Taking expectations, by Theorem 2 and by independence of M_0 from A':

$$\mathop{\mathbb{E}}_{A'} |M'| - \mathop{\mathbb{E}}_{A'} |A(M_0) \cap A'| \geq \frac{p}{1+p}|\text{opt}(G)| - p(\frac{1}{2} + \epsilon)|\text{opt}(G)|.$$

Combining:

$$\mathop{\mathbb{E}}_{A'} |M| \geq (\frac{1}{2} + \epsilon)|\text{opt}(G)| + \frac{1}{2}\left(|\text{opt}(G)|p(\frac{1}{1+p} - \frac{1}{2} - \epsilon) - 4\epsilon|\text{opt}(G)|.\right)$$

For ϵ small, the right hand side is maximized for $p = \sqrt{2} - 1$. Then $\epsilon \approx 0.019$ minimizes $\max\{|M|, |M_0|\}$ which proves the theorem.

5 Deterministic Two-Pass Algorithm on Any Order

The deterministic two-pass algorithm, Algorithm 4, follows the same line as its randomized version, Algorithm 2. In a first pass we compute a Greedy matching M_0 and some additional edges S via Algorithm 3. If M_0 is not much more than a $\frac{1}{2}$-approximation then S contains edges that serve as parts of 3-augmenting paths. These are completed via a Greedy matching in the second pass.

The way we compute the edge set S is now different. Before, S was a matching M' between B and a random subset A' of A. Now, S is not a matching but a relaxation of matchings as follows. Given an integer $\lambda \geq 2$, an *incomplete λ-bounded semi-matching* S of a bipartite graph $G = (A, B, E)$ is a subset $S \subseteq E$ such that $\deg_S(a) \leq 1$ and $\deg_S(b) \leq \lambda$, for all $a \in A$ and $b \in B$. This notion is closely related to semi-matchings. A semi-matching matches all A vertices to B vertices without limitations on the degree of a B vertex. However, since we require that the B vertices have constant degree, we loosen the condition that all A vertices need to be matched. In Lemma 9 (proof omitted) we show that Algorithm 3, a straightforward greedy algorithm, computes an incomplete λ-bounded semi-matching that covers at least $\frac{\lambda}{\lambda+1}|M^*|$ vertices of A.

Lemma 9. *Let $S = \text{SEMI}(\lambda)$ be the output of Algorithm 3 for some $\lambda \geq 2$. Then S is an incomplete λ-bounded semi-matching such that $|A(S)| \geq \frac{\lambda}{\lambda+1}|M^*|$.*

Algorithm 3. incomplete λ bounded semi-matching SEMI(λ)

```
S ← ∅
while ∃ edge ab in stream
  if deg_S(a) = 0 and deg_S(b) ≤ λ − 1 then S ← S ∪ {ab}
return S
```

Now, assume that the greedy matching algorithm computes a M_0 close to a $\frac{1}{2}$-approximation. Then, for $\lambda \geq 2$ there are many A vertices that are not matched in M_0 but are matched in S. Edges incident to those in S are candidates for the construction of 3-augmenting paths. This argument can be made rigorous, leading to Algorithm 4 where λ is set to 3, in Theorem 4 (proof omitted).

Theorem 4. *Algorithm 4 is a deterministic two-pass semi-streaming algorithm for* MBM *with approximation ratio $\frac{1}{2} + 0.019$ for any graph and any arrival order and can be implemented with $O(1)$ processing time per edge.*

Algorithm 4. two-pass deterministic bipartite matching algorithm

First pass: $M_0 \leftarrow \text{Greedy}(\pi)$ and $S \leftarrow \overline{\text{SEMI}(3)}$
$M_1 \leftarrow \{e \in S \,|\, e \text{ is between } B(M_0) \text{ and } \overline{A(M_0)}\}$
$A_2 \leftarrow \{a \in A(M_0) \,|\, \exists bc : ab \in M_0 \text{ and } bc \in M_1\}$
$F \leftarrow \{e \,|\, e \text{ goes between } A_2 \text{ and } \overline{B(M_0)}\}$
Second pass: $M_2 \leftarrow \text{Greedy}(\pi \cap F)$
Augment M_0 by edges in M_1 and M_2 and store it in M
return M

References

1. Alon, N., Matias, Y., Szegedy, M.: The space complexity of approximating the frequency moments. J. of Computer and System Sciences 58(1), 137–147 (1999)
2. Feigenbaum, J., Kannan, S., Mcgregor, A., Suri, S., Zhang, J.: Graph distances in the streaming model: the value of space. In: SODA, pp. 745–754 (2005)
3. Muthukrishnan, S.: Data streams: Algorithms and applications. In: Foundations and Trends in Theoretical Computer Science. Now Publishers Inc. (2005)
4. Feigenbaum, J., Kannan, S., McGregor, A., Suri, S., Zhang, J.: On graph problems in a semi-streaming model. Theoretical Computer Science 348(2-3), 207–216 (2005)
5. Ahn, K.J., Guha, S.: Linear Programming in the Semi-streaming Model with Application to the Maximum Matching Problem. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 526–538. Springer, Heidelberg (2011)
6. Eggert, S., Kliemann, L., Munstermann, P., Srivastav, A.: Bipartite matching in the semi-streaming model. Algorithmica 63(1-2), 490–508 (2012)
7. Zelke, M.: Weighted matching in the semi-streaming model. Algorithmica 62(1-2), 1–20 (2012)
8. Epstein, L., Levin, A., Mestre, J., Segev, D.: Improved approximation guarantees for weighted matching in the semi-streaming model. In: STACS, pp. 347–358 (2010)
9. Goel, A., Kapralov, M., Khanna, S.: On the communication and streaming complexity of maximum bipartite matching. In: SODA (2012)
10. Dyer, M., Frieze, A.: Randomized greedy matching. Random Structures & Algorithms 2(1), 29–46 (1991)
11. Munro, J., Paterson, M.: Selection and sorting with limited storage. Theoretical Computer Science 12, 211–219 (1980)
12. Guha, S., McGregor, A.: Stream order and order statistics: Quantile estimation in random-order streams. SIAM J. of Computing 38(1), 2044–2059 (2009)
13. Karp, R., Vazirani, U., Vazirani, V.: An optimal online bipartite matching algorithm. In: STOC, pp. 352–358 (1990)
14. Karande, C., Mehta, A., Tripathi, P.: Online bipartite matching with unknown distributions. In: STOC, pp. 587–596 (2011)
15. Mahdian, M., Yan, Q.: Online bipartite matching with random arrivals: an approach based on strongly factor-revealing LPs. In: STOC, pp. 597–605 (2011)
16. Nguyen, H.N., Onak, K.: Constant-time approximation algorithms via local improvements. In: FOCS, pp. 327–336 (2008)
17. Yoshida, Y., Yamamoto, M., Ito, H.: An improved constant-time approximation algorithm for maximum matchings. In: STOC, pp. 225–234 (2009)
18. McGregor, A.: Finding Graph Matchings in Data Streams. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX and RANDOM 2005. LNCS, vol. 3624, pp. 170–181. Springer, Heidelberg (2005)

Improved Inapproximability for TSP

Michael Lampis*

KTH Royal Institue of Technology
mlampis@kth.se

Abstract. The Traveling Salesman Problem is one of the most studied problems in computational complexity and its approximability has been a long standing open question. Currently, the best known inapproximability threshold known is $\frac{220}{219}$ due to Papadimitriou and Vempala. Here, using an essentially different construction and also relying on the work of Berman and Karpinski on bounded occurrence CSPs, we give an alternative and simpler inapproximability proof which improves the bound to $\frac{185}{184}$.

1 Introduction

The Traveling Salesman Problem (TSP) is one of the most widely studied algorithmic problems and deriving optimal approximability results for it has been a long-standing question. Recently, there has been much progress in the algorithmic front, after more than thirty years, at least in the important special case where the instance metric is derived from an unweighted graph, often referred to as Graphic TSP. The $\frac{3}{2}$-approximation algorithm by Christofides was the best known until Gharan et al. gave a slight improvement [6] for Graphic TSP. Then an algorithm with approximation ratio 1.461 was given by Mömke and Svensson [10]. With improved analysis on their algorithm Mucha obtained a ratio of $\frac{13}{9}$ [11], while the best currently known algorithm has ratio 1.4 and is due to Sebö and Vygen [16].

Nevertheless, there is still a huge gap between the guarantee of the best approximation algorithms we know and the best inapproximability results. The TSP was first shown MAXSNP-hard in [15], where no explicit inapproximability constant was derived. The work of Engerbretsen [5] and Böckenhauer et al. [4] gave inapproximability thresholds of $\frac{5381}{5380}$ and $\frac{3813}{3812}$ respectively. Later, this was improved to $\frac{220}{219}$ in [14] by Papadimitriou and Vempala[1]. No further progress has been made on the inapproximability threshold of this problem in the more than ten years since [13].

Overview: Our main objective in this paper is to give a different, less complicated inapproximability proof for TSP than the one given in [13,14]. The proof of [14] is very much optimized to achieve a good constant: the authors reduce

* Research supported by ERC Grant 226203.

[1] The reduction of [14] was first presented in [13], which (erroneously) claimed a better bound.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 243–253, 2012.
© Springer-Verlag Berlin Heidelberg 2012

directly from MAX-E3-LIN2, a constraint satisfaction problem (CSP) for which optimal inapproximability results are known, due to Håstad [7]. They take care to avoid introducing extra gadgets for the variables, using only gadgets that encode the equations. Finally they define their own custom expander-like notion on graphs to ensure consistency between tours and assignments. Then the reduction is performed in essentially one step.

Here on the other hand we take the opposite approach, choosing simplicity over optimization. We also start from MAX-E3-LIN2 but go through two intermediate CSPs. The first step in our reduction gives a set of equations where each variable appears at most five times (this property will come in handy in the end when proving consistency between tours and assignments). In this step, rather than introducing something new we rely heavily on machinery developed by Berman and Karpinski to prove inapproximability for bounded occurrence CSPs [1,2,3]. As a second step we reduce to MAX-1-IN-3-SAT. The motivation is that the 1-IN-3 predicate nicely corresponds to the objectives of TSP, since we represent clauses by gadgets and the most economical solution will visit all gadgets once but not more than once. Another way to view this step is that we use MAX-1-IN-3-SAT as an aid to design a TSP gadget for parity. Finally, we give a reduction from MAX-1-IN-3-SAT to TSP.

This approach is (at least arguably) simpler than the approach of [14], since some of our arguments can be broken down into independent pieces, arguing about the inapproximability of intermediate, specially constructed CSPs. We also benefit from re-using out-of-the box the amplifier construction of [3]. Interestingly, putting everything together we end up obtaining a slightly better constant than the one currently known, implying that there may still be some room for further improvement. Though we are still a long way from an optimal inapproximability result, our results show that there may still be hope for better bounds with existing tools. Exploring how far these techniques can take us with respect to TSP (and also its variants, see for example [8]) may thus be an interesting question.

The main result of this paper is given below and it follows directly from the construction in section 4.1 and Lemmata 1,2 [2].

Theorem 1. *For all $\epsilon > 0$ there is no polynomial-time $(\frac{92.3}{91.8} - \epsilon)$-approximation algorithm for TSP, unless P=NP.*

2 Preliminaries

In the metric Traveling Salesman Problem (TSP) we are given as input an edge-weighted undirected graph $G(V, E)$. Let $d(u, v)$, for $u, v \in V$ denote the shortest-path distance from u, v. The objective is to find an ordering $v_1, v_2, \ldots, v_n$ of the vertices such that $\sum_{i=1}^{n-1} d(v_i, v_{i+1}) + d(v_n, v_1)$ is minimized.

[2] Due to space constraints, some proofs have been omitted. A full version of this paper can be found in [9].

Another, equivalent view of the TSP is the following: given an edge-weighted graph $G(V, E)$ we seek to find a multi-set E_T consisting of edges from E such that the graph induced by E_T spans V, is Eulerian and the sum of the weights of all edges in E_T is minimized. It is not hard to see that the two formulations are equivalent. We will make use of this second formulation because it makes some arguments on our construction easier.

We generalize the Eulerian multi-graph formulation as follows: a multi-set E_T of edges from E is a quasi-tour iff the degrees of all vertices in the multi-graph $G_T(V, E_T)$ are even. The cost of a quasi-tour is defined as $\sum_{e \in E_T} w(e) + 2(c(G_T) - 1)$, where $c(G_T)$ denotes the number of connected components of the multi-graph. It is not hard to see that a TSP tour can also be considered a quasi-tour with the same cost (since for a normal tour $c(G_T) = 1$), but in a weighted graph there could potentially be a quasi-tour that is cheaper than the optimal tour.

2.1 Forced Edges

As mentioned, we will view TSP as the problem of selecting edges from E to form a minimum-weight multi-set E_T that makes the graph Eulerian. It is easy to see that no edge will be selected more than twice, since if an edge is selected three times we can remove two copies of it from E_T and the graph will still be Eulerian while we have improved the cost.

In our construction we would like to be able to stipulate that some edges are to be used at least once in any valid tour. We can achieve this with the following trick: suppose that there is an edge (u, v) with weight w that we want to force into every tour. We sub-divide this edge a large number of times, say $p - 1$, that is, we remove the edge and replace it with a path of p edges going through new vertices of degree two. We then redistribute the original edge's weight to the p newly formed edges, so that each has weight w/p. Now, any tour that fails to use two or more of the newly formed edges must be disconnected. Any tour that fails to use exactly one of them can be augmented by adding two copies of the unused edge. This only increases the cost by $2w/p$, which can be made arbitrarily small by giving p an appropriately large value. Therefore, we may assume without loss of generality that in our construction we can force some edges to be used at least once. Note that these arguments apply also to quasi-tours.

3 Intermediate CSPs

In this section we will design and prove inapproximability for a family of instances of MAX-1-IN-3-SAT with some special structure. We will use these instances (and their structure) in the next section where we reduce from MAX-1-IN-3-SAT to TSP.

Let I_1 be a system of m linear equations mod 2, each consisting of exactly three variables. Let n be the total number of variables appearing in I_1 and let the variables be denoted as $x_i, i \in [n]$. Let B be the maximum number of times

any variable appears. We will make use of the following seminal result due to Håstad:

Theorem 2 ([7])
For all $\epsilon > 0$ there exists a B such that given an instance I_1 as above it is NP-hard to decide if there is an assignment that satisfies at least $(1-\epsilon)m$ equations or all assignment satisfy at most $(\frac{1}{2}+\epsilon)m$ equations.

3.1 Bounded Occurences

In I_1 each variable appears at most a constant number of times B, where B depends on ϵ. We would like to reduce the maximum number of occurrences of each variable to a small absolute constant. For this, one typically uses some kind of expander or amplifier construction. Here we will rely on a construction due to Berman and Karpinski that reduces the number of occurrences to 5.

Theorem 3 ([3])
Consider the family of bipartite graphs $G(L, R, E)$, where $|L| = B, |R| = 0.8B$, all vertices of L have degree 4, all vertices of R have degree 5 and B is a sufficiently large multiple of 5. If we select uniformly at random a graph from this family then with high probability it has the following property: for any $S \subseteq L \cup R$ such that $|S \cap L| \leq \frac{|L|}{2}$ the number of edges in E with exactly one endpoint in S is at least $|S \cap L|$.

We now use the above construction to construct a system of equations where each variable appears exactly 5 times. First, we may assume that in I_1 the number of appearances of each variable is a multiple of 5 (otherwise, repeat all equations five times). Also, by repeating all the equations we can make sure that all variables appear at least B' times, where B' is a sufficiently large number to make Theorem 3 hold.

For each variable x_i in I_1 we introduce the variables $x_{(i,j)}, j \in [d(i)]$ and $y_{(i,j)}, j \in [0.8d(i)]$ where $d(i)$ is the number of appearances of x_i in the original instance. We call $X_i = \{x_{(i,j)} \mid j \in [d(i)]\} \cup \{y_{(i,j)} \mid j \in [0.8d(i)]\}$ the cloud that corresponds to x_i. Construct a bipartite graph with the property described in Theorem 3 with $L = [d(i)], R = [0.8d(i)]$ (since $d(i) < B$ is a constant that depends only on ϵ this can be done in constant time by brute force). For each edge $(j,k) \in E$ introduce the equation $x_{(i,j)} + y_{(i,k)} = 1$. Finally, for each equation $x_{i_1} + x_{i_2} + x_{i_3} = b$ in I_1, where this is the j_1-th appearance of x_{i_1}, the j_2-th appearance of x_{i_2} and the j_3-th appearance of x_{i_3} replace it with the equation $x_{(i_1,j_1)} + x_{(i_2,j_2)} + x_{(i_3,j_3)} = b$.

Denote this instance by I_2 and we have $|I_2| = 13m$, with $12m$ equations having size 2. A consistent assignment to a cloud X_i is an assignment that sets all $x_{(i,j)}$ to b and all $y_{(i,j)}$ to $1-b$. By standard arguments using the graph of Theorem 3 we can show that an optimal assignment to I_2 is consistent (in each inconsistent cloud let S be the vertices with the minority assignment; flipping all variables of S cannot make the solution worse). From this it follows that it is NP-hard to

distinguish if the maximum number of satisfiable equations is at least $(13-\epsilon)m$ or at most $(12.5+\epsilon)m$.

3.2 MAX-1-IN-3-SAT

In the MAX-1-IN-3-SAT problem we are given a collection of clauses $(l_i \vee l_j \vee l_k)$, each consisting of at most three literals, where each literal is either a variable or its negation. A clause is satisfied by a truth assighment if exactly one of its literals is set to True. The problem is to find an assignment that satisfies the maximum number of clauses.

We would like to produce a MAX-1-IN-3-SAT instance from I_2. Observe that it is easy to turn the size two equations $x_{(i,j)}+y_{(i,k)}=1$ to the equivalent clauses $(x_{(i,j)} \vee y_{(i,k)})$. We only need to worry about the m equations of size three.

If the k-th size-three equation of I_2 is $x_{(i_1,j_1)} + x_{(i_2,j_2)} + x_{(i_3,j_3)} = 1$ we introduce three new auxilliary variables $a_{(k,i)}, i \in [3]$ and replace the equation with the three clauses $(x_{(i_1,j_1)} \vee a_{(k,1)} \vee a_{(k,2)})$, $(x_{(i_2,j_2)} \vee a_{(k,2)} \vee a_{(k,3)})$, $(x_{(i_3,j_3)} \vee a_{(k,1)} \vee a_{(k,3)})$. If the right-hand-side of the equation is 0 then we add the same three clauses except we negate $x_{(i_1,j_1)}$ in the first clause. We call these three clauses the cluster that corresponds to the k-th equation.

It is not hard to see that if we fix an assignment to $x_{(i_1,j_1)}, x_{(i_2,j_2)}, x_{(i_3,j_3)}$ that satisfies the k-th equation of I_2 then there exists an assignment to $a_{(k,1)}, a_{(k,2)}, a_{(k,3)}$ that satisfies the whole cluster. Otherwise, at most two of the clauses of the cluster can be satisfied. Furthermore, in this case there exist three different assignments to the auxilliary variables that satisfy two clauses and each leaves a different clause unsatisfied.

From now on, we will denote by M the set of (main) variables $x_{(i,j)}$, by C the set of (checker) variables $y_{(i,j)}$ and by A the set of (auxilliary) variables $a_{(k,i)}$. Call the instance of MAX-1-IN-3-SAT we have constructed I_3. Note that it consists of $15m$ clauses and $8.4m$ variables.

4 TSP

4.1 Construction

We now describe a construction that encodes I_3 into a TSP instance $G(V,E)$. Rather than viewing this as a generic construction from MAX-1-IN-3-SAT to TSP, we will at times need to use facts that stem from the special structure of I_3. In particular, the fact that variables can be partitioned into sets M, C, A, such that variables in $M \cup C$ appear five times and variables in A appear twice; the fact that most clauses have size two and they involve one positive variable from M and one positive variable from C; and also the fact that clauses of size three come in clusters as described in the construction of I_3.

As mentioned, we assume that in the graph $G(V,E)$ we may include some forced edges, that is, edges that have to be used at least once in any tour. The graph includes a central vertex, which we will call s. For each variable in

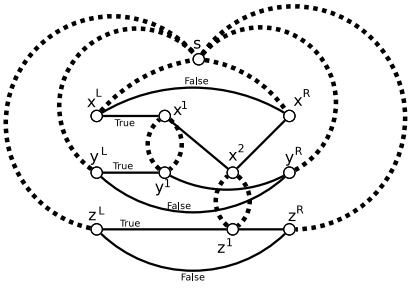

Fig. 1. Example construction for the clause $(x \vee y) \wedge (x \vee z)$. Forced edges are denoted by dashed lines. There are two terminals for each variable and two gadgets that represent the two clauses. The True edges incident on the terminals are re-routed through the gadgets where each variable appears positive. The False edges connect the terminals directly since no variable appears anywhere negated.

$x \in M \cup C \cup A$ we introduce two new vertices named x^L and x^R, which we will call the left and right terminal associated with x. We add a forced edge from each terminal to s. For terminals that correspond to variables in $M \cup C$ this edge has weight $7/4$, while for variables in A it has weight $1/2$. We also add two (parallel) non-forced edges between each pair of terminals representing the same variable, each having a weight of 1 (we will later break down at least one from each pair of these, so the graph we will obtain in the end will be simple). Informally, these two edges encode an assignment to each variable: we arbitrarily label one the True edge and the other the False edge, the idea being that a tour should pick exactly one of these for each variable and that will give us an assignment. We will re-route these edges through the clause gadgets as we introduce them, depending on whether each variable appears in a clause positive or negative.

Now, we add some gadgets to encode the size-two clauses of I_3. Let $(x_{(i,j_1)} \vee y_{(i,j_2)})$ be a clause of I_3 and suppose that this is the k_1-th clause that contains $x_{(i,j_1)}$ and the k_2-th clause that contains $y_{(i,j_2)}$, $k_1, k_2 \in [5]$. Then we add two new vertices to the graph, call them $x^{k_1}_{(i,j_1)}$ and $y^{k_2}_{(i,j_2)}$. Add two forced edges between them, each of weight $3/2$ (recall that forced edges represent long paths, so these are not really parallel edges). Finally, re-route the True edges incident on $x^L_{(i,j_1)}$ and $y^L_{(i,j_2)}$ through $x^{k_1}_{(i,j_1)}$ and $y^{k_2}_{(i,j_2)}$ respectively. More precisely, if the True edge incident on $x^L_{(i,j_1)}$ connects it to some other vertex u, remove that edge from the graph and add an edge from $x^L_{(i,j_1)}$ to $x^{k_1}_{(i,j_1)}$ and an edge from $x^{k_1}_{(i,j_1)}$ to u. All these edges have weight one and are non-forced (see Figure 1).

We use a similar gadget for clauses of size three. Consider a cluster $(x_{(i_1,j_1)} \vee a_{(k,1)} \vee a_{(k,2)})$, $(x_{(i_2,j_2)} \vee a_{(k,2)} \vee a_{(k,3)})$, $(x_{(i_3,j_3)} \vee a_{(k,1)} \vee a_{(k,3)})$ and suppose for simplicity that this is the fifth appearance for all the main variables of the cluster. Then we add the new vertices $x^5_{(i_1,j_1)}, x^5_{(i_2,j_2)}, x^5_{(i_3,j_3)}$ and also the vertices $a^1_{(k,1)}, a^2_{(k,1)}, a^1_{(k,2)}, a^2_{(k,2)}$ and $a^1_{(k,3)}, a^2_{(k,3)}$. To encode the first clause we add two forced edges of weight $5/4$, one from $x^5_{(i_1,j_1)}$ to $a^1_{(k,1)}$ and one from $x^5_{(i_1,j_1)}$ to

$a^1_{(k,2)}$. We also add a forced edge of weight 1 from $a^1_{(k,1)}$ to $a^1_{(k,2)}$, thus making a triangle with the forced edges (see Figure 2). We re-route the True edge from $a^L_{(k,1)}$ through $a^1_{(k,1)}$ and $a^2_{(k,1)}$. We do similarly for the other two auxilliary variables and the main variables. Finally, for a cluster where $x_{(i_1,j_1)}$ is negated, we use the same construction except that rather than re-routing the True edge that is incident on $x^L_{(i_1,j_1)}$ we re-route the False edge. This completes the construction.

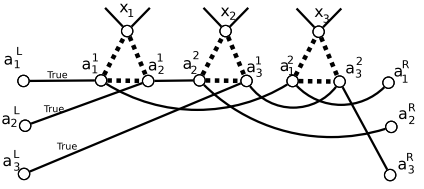

Fig. 2. Example construction fragment for the cluster $(x_1 \vee a_1 \vee a_2) \wedge (x_2 \vee a_2 \vee a_3) \wedge (x_3 \vee a_1 \vee a_3)$. The False edges which connect each pair of terminals and the forced edges that connect terminals to s are not shown.

4.2 From Assignment to Tour

Let us now prove one direction of the reduction and in the process also give some intuition about the construction. Call the graph we have constructed $G(V, E)$.

Lemma 1. *If there exists an assignment to the variables of I_3 that leaves at most k equations unsatisfied, then there is a tour of G with cost at most $T = L + k$, where $L = 91.8m$.*

Proof. Observe that by construction we may assume that all the unsatisfied clauses of I_3 are in the clusters and that at most one clause in each cluster is unsatisfied, otherwise we can obtain a better assignment. Also, if an unsatisfied clause has all literals set to False we can flip the value of one of the auxilliary variables without increasing the number of violated clauses. Thus, we may assume that all clauses have a True literal. Also, we may assume that no clause has all literals set to True: suppose that a clause does, then both auxilliary variables of the clause are True. We set them both to False, gaining one clause. If this causes the two other clauses of the cluster to become unsatisfied, set the remaining auxilliary variable to True. We conclude that all clauses have either one or two True literals.

Our tour uses all forced edges exactly once. For each variable x set to True in the assignment the tour selects the True edge incident on the terminal corresponding to x. If the edge has been re-routed all its pieces are selected, so that we have selected edges that make up a path from x^L to x^R. Otherwise, if x is set to False in the assignment the tour selects the corresponding False path.

Observe that this is a valid quasi-tour because all vertices have even degree (for each terminal we have selected the forced edge plus one more edge, for gadget vertices we have selected the two forced edges and possibly the two edges through which True or False was re-routed). Also, observe that the tour must be connected, because each clause contains a True literal, therefore for each gadget two of its external edges have been selected and they are part of a path that leads to the terminals.

The cost of the tour is at most $F + N + M + k$, where F is the total cost of all forced edges in the graph and N, M are the total number of variables and clauses respectively in I_3. To see this, notice that there are $2N$ terminals, and there is one edge incident on each and there are M clause gadgets, $M - k$ of which have two selected edges incident on them and k of which have four. Summing up, this gives $2N + 2M + 2k$, but then each unit-weight edge has been counted twice, meaning that the non-forced edges have a total cost of $N + M + k$.

Finally, we have $N = 8.4m$, $M = 15m$ and $F = 3 \times 12m + \frac{7}{2} \times 3m + \frac{7}{2} \times 5.4m + 1 \times 3m = 68.4m$, where the terms are respectively the cost of size-two clause gadgets, the cost of size-three clause gadgets, the cost of edges connecting terminals to s for the main variables and for the auxilliary variables. We have $F + N + M = 91.8m$. □

4.3 From Tour to Assignment

We would like now to prove the converse of Lemma 1, namely that if a tour of cost $L + k$ exists then we can find an assignment that leaves at most k clauses unsatisfied. Let us first give some high-level intuition and in the process justify the weights we have selected in our construction.

Informally, we could start from a simple base case: suppose that we have a tour such that all edges of G are used at most once. It is not hard to see that this then corresponds to an assignment, as in the proof of Lemma 1. So, the problem is how to avoid tours that may use some edges twice.

To this end, we first give some local improvement arguments that make sure that the number of problematic edges, which are used twice, is limited. However, arguments like these can only take us so far, and we would like to avoid having too much case analysis.

We therefore try to isolate the problem. For variables in $M \cup C$ which the tour treats honestly, that is, variables which are not involved with edges used twice, we directly obtain an assignment from the tour. For the other variables in $M \cup C$ we pick a random value and then extend the whole assignment to A in an optimal way. We want to show that the expected number of unsatisfied clauses is at most k.

The first point here is that if a clause containing only honest variables turns out to be violated, the tour must also be paying an extra cost for it. The difficulty is therefore concentrated on clauses with dishonest variables.

By using some edges twice the tour is paying some cost on top of what is accounted for in L. We would like to show that this extra cost is larger than the number of clauses violated by the assignment. It is helpful to think here that it

is sufficient to show that the tour pays an additional cost of $\frac{5}{2}$ for each dishonest variable, since main variables appear 5 times.

A crucial point now is that, by a simple parity argument, there has to be an even number of violations (that is, edges used twice) for each variable (Lemma 4). This explains the weights we have picked for the forced edges in size-three gadgets ($\frac{5}{4}$) and for edges connecting terminals to s ($\frac{7}{4} = \frac{5}{4} + \frac{1}{2}$ or $\frac{5}{4}$ extra to the cost already included in L for fixing the parity of the terminal vertex). Two such violations give enough extra cost to pay for the expected number of unsatisfied clauses containing the variable.

At this point, we could also set the weights of forced edges in size-two gadgets to $\frac{5}{2}$, which would be split among the two dishonest variables giving $\frac{5}{4}$ to each. Then, any two violations would have enough additional cost to pay for the expected unsatisfied clauses. However, we are slightly more careful here: rather than setting all dishonest variables in $M \cup C$ independently at random, we pick a random but consistent assignment for each cloud. This ensures that all size-two clauses with violations will be satisfied. Thus, it is sufficient for violations in them to have a cost of $\frac{3}{2}$: the amount "paid" to each variable is now $\frac{3}{4} = \frac{5}{4} - \frac{1}{2}$, but the expected number of unsatisfied clauses with this variable is also decreased by $\frac{1}{2}$ since one clause is surely satisfied.

Due to space constraints we provide here only a sketch of the rest of the proof. Recall that if a tour of a certain cost exists, then there exists also a quasi-tour of the same cost. It suffices then to prove the following:

Lemma 2. *If there exists a quasi-tour of G with cost at most $L + k$ then there exists an assignment to the variables of I_3 that leaves at most k clauses unsatisfied.*

In order to prove Lemma 2 it is helpful to first make some easy observations. First, all (non-forced) edges of weight one are used at most once. Second, in each gadget there is at most one forced edge that is used twice. Third, for each variable x, at least one of the forced edges that connect x^L, x^R to s is used exactly once.

Given a tour E_T, we will say that a variable x is honestly traversed in that tour if all the forced edges that involve it are used exactly once (this includes the forced edges incident on x^L, x^R and x^i, $i \in [5]$).

Let us now give two more useful facts.

Lemma 3. *There exists an optimal tour where all forced edges between two different vertices that correspond to two variables in A are used exactly once.*

Lemma 4. *In an optimal tour, if a variable is dishonest then it must be dishonest twice. More precisely, the number of forced edges that involve the variable (either inside gadgets or connecting terminals to s) and are used twice must be even.*

Observe that it follows from Lemmata 3,4 that if all the main variables involved in a cluster are honest then the auxilliary variables of that cluster are also honest. This holds because if the main variables are honest then by Lemma 3 no forced edge inside the gadgets of the cluster is used twice, so by Lemma 4 and the fact that at least one of the forced edges incident on the terminals is used once, the auxilliary variables are honest.

We would like now to be able to extract a good assignment even if a tour is not honest, thus indirectly proving that honest tours are optimal. This is done in the (omitted) proof of Lemma 2 by the random assignment method we have already sketched.

5 Conclusions

We have given an alternative and (we believe) simpler inapproximability proof for TSP, also modestly improving the known bound. We believe that the approach followed here where the hardness proof goes explicitly through bounded occurrence CSPs is more promising than the somewhat ad-hoc method of [14], not only because it is easier to understand but also because we stand to gain almost "automatically" from improvements in our understanding of the inapproximability of bounded occurrence CSPs. In particular, though we used the 5-regular amplifiers from [3], any such amplifier would work essentially "out of the box", and any improved construction could imply an improvement in our bound. Nevertheless, the distance between the upper and lower bounds on the approximability of TSP remains quite large and it seems that some major new idea will be needed to close it.

References

1. Berman, P., Karpinski, M.: On Some Tighter Inapproximability Results (Extended Abstract). In: Wiedermann, J., Van Emde Boas, P., Nielsen, M. (eds.) ICALP 1999. LNCS, vol. 1644, pp. 200–209. Springer, Heidelberg (1999)
2. Berman, P., Karpinski, M.: Efficient amplifiers and bounded degree optimization. Electronic Colloquium on Computational Complexity (ECCC) 8(53) (2001)
3. Berman, P., Karpinski, M.: Improved approximation lower bounds on small occurrence optimization. Electronic Colloquium on Computational Complexity (ECCC) 10(008) (2003)
4. Böckenhauer, H.-J., Hromkovič, J., Klasing, R., Seibert, S., Unger, W.: An Improved Lower Bound on the Approximability of Metric TSP and Approximation Algorithms for the TSP with Sharpened Triangle Inequality. In: Reichel, H., Tison, S. (eds.) STACS 2000. LNCS, vol. 1770, pp. 382–394. Springer, Heidelberg (2000)
5. Engebretsen, L.: An explicit lower bound for TSP with distances one and two. Algorithmica 35(4), 301–318 (2003)
6. Gharan, S.O., Saberi, A., Singh, M.: A randomized rounding approach to the traveling salesman problem. In: Ostrovsky [12], pp. 550–559
7. Håstad, J.: Some optimal inapproximability results. Journal of the ACM (JACM) 48(4), 798–859 (2001)

8. Karpinski, M., Schmied, R.: On approximation lower bounds for TSP with bounded metrics. CoRR, abs/1201.5821 (2012)
9. Lampis, M.: Improved Inapproximability for TSP. CoRR, abs/1206.2497 (2012)
10. Mömke, T., Svensson, O.: Approximating graphic TSP by matchings. In: Ostrovsky [12], pp. 560–569
11. Mucha, M.: 13/9-approximation for graphic TSP. In: Dürr, C., Wilke, T. (eds.) STACS. LIPIcs, vol. 14, pp. 30–41. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik (2012)
12. Ostrovsky, R. (ed.): IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25. IEEE (2011)
13. Papadimitriou, C.H., Vempala, S.: On the approximability of the traveling salesman problem (extended abstract). In: Yao, F.F., Luks, E.M. (eds.) STOC, pp. 126–133. ACM (2000)
14. Papadimitriou, C.H., Vempala, S.: On the approximability of the traveling salesman problem. Combinatorica 26(1), 101–120 (2006)
15. Papadimitriou, C.H., Yannakakis, M.: The traveling salesman problem with distances one and two. Mathematics of Operations Research, 1–11 (1993)
16. Sebö, A., Vygen, J.: Shorter tours by nicer ears: CoRR, abs/1201.1870 (2012)

Approximation Algorithm for Non-boolean MAX k-CSP

Konstantin Makarychev[1] and Yury Makarychev[2,⋆]

[1] Microsoft Research
[2] Toyota Technological Institute at Chicago

Abstract. In this paper, we present a randomized polynomial-time approximation algorithm for MAX k-CSP$_d$. In MAX k-CSP$_d$, we are given a set of predicates of arity k over an alphabet of size d. Our goal is to find an assignment that maximizes the number of satisfied constraints.

Our algorithm has approximation factor $\Omega(kd/d^k)$ (when $k \geq \Omega(\log d)$). This bound is asymptotically optimal assuming the Unique Games Conjecture. The best previously known algorithm has approximation factor $\Omega(k \log d/d^k)$.

We also give an approximation algorithm for the boolean MAX k-CSP$_2$ problem with a slightly improved approximation guarantee.

1 Introduction

We design an approximation algorithm for the MAX k-CSP$_d$, the maximum constraint satisfaction problem with k-ary predicates and domain size d. In this problem, we are given a set of variables $\{x_u\}_{u\in X}$ and a set of predicates $\mathcal{P}$. Each variable x_u takes values in $[d] = \{1, \dots, d\}$. Each predicate $P \in \mathcal{P}$ depends on at most k variables. Our goal is to assign values to variables so as to maximize the number of satisfied constraints.

There has been a lot of interest in finding the approximability of MAX k-CSP$_d$ in the complexity community motivated by the connection of MAX k-CSP$_d$ to k-bit PCPs. Let us briefly overview known results. Samorodnitsky and Trevisan [9] showed that the boolean MAX k-CSP$_2$ problem cannot be approximated within a factor of $\Omega(2^{2\sqrt{k}}/2^k)$ if $P \neq NP$. Later Engebretsen and Holmerin [5] improved this bound to $\Omega(2^{\sqrt{2k}}/2^k)$. For non-boolean MAX k-CSP$_d$, Engebretsen [4] proved a hardness result of $2^{O(\sqrt{d})}/d^k$. Much stronger inapproximability results were obtained assuming the Unique Games Conjecture (UGC). Samorodnitsky and Trevisan [10] proved the hardness of $O(k/2^k)$ for the boolean MAX k-CSP$_2$. Austrin and Mossel [1] and, independently, Guruswami and Raghavendra [6] proved the hardness of $O(kd^2/d^k)$ for non-boolean MAX k-CSP$_d$. Moreover, Austrin and Mossel [1] proved the hardness of $O(kd/d^k)$ for every d and infinitely many k; specifically, their result holds for d and k such that $k = (d^t - 1)/(d - 1)$ for some $t \in \mathbb{N}$. Very recently, Håstad strengthened the

⋆ Yury Makarychev is supported in part by the NSF Career Award CCF-1150062.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 254–265, 2012.
© Springer-Verlag Berlin Heidelberg 2012

result of Austrin and Mossel and showed the hardness of $O(kd/d^k)$ for every d and $k \geq d$ [private communication].

On the positive side, approximation algorithms for the problem have been developed in a series of papers by Trevisan [12], Hast [7], Charikar, Makarychev and Makarychev [3], and Guruswami and Raghavendra [6]. The best currently known algorithm for k-CSP$_d$ by Charikar et al [3] has approximation factor of $\Omega(k \log d/d^k)$. Note that a trivial algorithm for MAX k-CSP$_d$ that just picks a random assignment satisfies each constraint with probability at least $1/d^k$, and therefore its approximation ratio is $1/d^k$.

The problem is essentially settled in the boolean case. We know that the optimal approximation factor is $\Theta(k/2^k)$ assuming UGC. However, best known lower and upper bounds for the non-boolean case do not match. In this paper, we present an approximation algorithm for non-boolean MAX k-CSP$_d$ with approximation factor $\Omega(kd/d^k)$ (for $k \geq \Omega(\log d)$). This algorithm is asymptotically optimal assuming UGC — it is within a constant factor of the upper bounds of Austrin and Mossel and of Håstad (for k of the form $(d^t-1)/(d-1)$ and for $k \geq d$, respectively). Our result improves the best previously known approximation factor of $\Omega(k \log d/d^k)$.

Related Work. Raghavendra studied a more general MAX CSP($\mathcal{P}$) problem [8]. He showed that the optimal approximation factor equals the integrality gap of the standard SDP relaxation for the problem (assuming UGC). His result applies in particular to MAX k-CSP$_d$. However, the SDP integrality gap of MAX k-CSP$_d$ is not known.

Overview. We use semidefinite programming (SDP) to solve the problem. In our SDP relaxation, we have an "indicator vector" u_i for every variable x_u and value i; we also have a "indicator vector" z_C for every constraint C. In the intended solution, u_i is equal to a fixed unit vector $\mathbf{e}$ if $x_u = i$, and $u_i = 0$ if $x_u \neq i$; similarly, $z_C = \mathbf{e}$ if C is satisfied, and $z_C = 0$, otherwise.

It is interesting that the best previously known algorithm for the problem [3] did not use this SDP relaxation; rather it reduced the problem to a binary k-CSP problem, which it solved in turn using semidefinite programming. The only previously known algorithm [6] that directly rounded an SDP solution for MAX k-CSP$_d$ had approximation factor $\Omega\left(\frac{k/d^7}{d^k}\right)$.

One of the challenges of rounding the SDP solution is that vectors u_i might have different lengths. Consequently, we cannot just use a rounding scheme that projects vectors on a random direction and then chooses vectors that have largest projections, since this scheme will choose longer vectors with disproportionately large probabilities. To deal with this problem, we first develop a rounding scheme that rounds *uniform* SDP solutions, solutions in which all vectors are "short". Then we construct a randomized reduction that converts any instance to an instance with a uniform SDP solution.

Our algorithm for the uniform case is very simple. First, we choose a random Gaussian vector g. Then for every u, we find u_i that has the largest projection on g (in absolute value), and let $x_u = i$. However, the analysis of this algorithm

is quite different from analyses of similar algorithms for other problems: when we estimate the probability that a constraint C is satisfied, we have to analyze the correlation of all vectors u_i with vector z_C (where $\{u_i\}$ are SDP vectors for variables x_u that appear in C, z_C is the SDP vector for C), whereas the standard approach would be to look only at pairwise correlations of vectors $\{u_i\}$; this approach does not work in our case, however, since vectors corresponding to an assignment that satisfies C may have very small pairwise correlations, but vectors corresponding to assignments that do not satisfy C may have much larger pairwise correlations.

Remark 1.1. We study the problem only in the regime when $k \geq \Omega(\log d)$. In Theorem 5.1, we prove that when $k = O(\log d)$ our algorithm has approximation factor $e^{\Omega(k)}/d^k$. However, in this regime, a better approximation factor of $\Omega(d/d^k)$ can be obtained by a simple greedy approach.

Other Results. We also apply our SDP rounding technique to the Boolean Maximum CSP Problem. We give an algorithm that has approximation guarantee $\approx 0.62\,k/2^k$ for sufficiently large k. That slightly improves the best previously known guarantee of $\approx 0.44\,k/2^k$ [3]. Due to space limitations, we present this result only in the full version of our paper.

2 Preliminaries

We apply the approximation preserving reduction of Trevisan [12] to transform a general instance of MAX k-CSP$_d$ to an instance where each predicate is a conjunction of terms of the form $x_u = i$. The reduction replaces a predicate P, which depends on variables $x_{v_1}, \ldots, x_{v_k}$, with a set of clauses

$$\{(x_{v_1} = i_1) \wedge \cdots \wedge (x_{v_k} = i_k) : P(i_1, \ldots, i_k) \text{ is true}\}.$$

Then it is sufficient to solve the obtained instance. We refer the reader to [12] for details. We assume below that each predicate is a clause of the form $(x_{v_1} = i_1) \wedge \cdots \wedge (x_{v_k} = i_k)$.

Definition 2.1 (Constraint satisfaction problem). *An instance $\mathcal{I}$ of MAX CSP$_d$ consists of*

- *a set of "indices" X,*
- *a set of variables $\{x_u\}_{u \in X}$ (there is one variable x_u for every index $u \in X$),*
- *a set of clauses $\mathcal{C}$.*

Each variable x_u takes values in the domain $[d] = \{1, \ldots, d\}$. Each clause $C \in \mathcal{C}$ is a set of pairs (u, i) where $u \in X$ and $i \in [d]$. An assignment $x_u = x_u^*$ satisfies a clause C if for every $(u, i) \in C$, we have $x_u^* = i$. We assume that no clause C in $\mathcal{C}$ contains pairs (u, i) and (u, j) with $i \neq j$ (no assignment satisfies such clause). The length of a clause C is $|C|$. The support of C is $\operatorname{supp}(C) = \{u : (u, i) \in C\}$.

The value of an assignment x^*_u is the number of constraints in $\mathcal{C}$ satisfied by x^*_u. Our goal is to find an assignment of maximum value. We denote the value of an optimal assignment by $OPT = OPT(\mathcal{I})$.

In the MAX k-CSP$_d$ problem, we additionally require that all clauses in $\mathcal{C}$ have length at most k.

We consider the following semidefinite programming (SDP) relaxation for MAX CSP$_d$. For every index $u \in X$ and $i \in [d]$, we have a vector variable u_i; for every clause C, we have a vector variable z_C.

$$\text{maximize: } \sum_{C \in \mathcal{C}} \|z_C\|^2$$

subject to

$$\begin{aligned}
&\sum_{i=1}^{d} \|u_i\|^2 \le 1 && \text{for every } u \in X \\
&\langle u_i, u_j \rangle = 0 && \text{for every } u \in X, i, j \in [d] \ (i \neq j) \\
&\langle u_i, z_C \rangle = \|z_C\|^2 && \text{for every } C \in \mathcal{C},\ (u,i) \in C \\
&\langle u_j, z_C \rangle = 0 && \text{for every } C \in \mathcal{C},\ (u,i) \in C \text{ and } j \neq i
\end{aligned}$$

Denote the optimal SDP value by $SDP = SDP(\mathcal{I})$. Consider the optimal solution x^*_u to an instance $\mathcal{I}$ and the corresponding SDP solution defined as follows,

$$u_i = \begin{cases} \mathbf{e}, & \text{if } x^*_u = i; \\ 0, & \text{otherwise;} \end{cases} \qquad z_C = \begin{cases} \mathbf{e}, & \text{if } C \text{ is satisfied;} \\ 0, & \text{otherwise;} \end{cases}$$

where $\mathbf{e}$ is a fixed unit vector. It is easy to see that this is a feasible SDP solution and its value equals $OPT(\mathcal{I})$. Therefore, $SDP(\mathcal{I}) \ge OPT(\mathcal{I})$.

Definition 2.2. *We say that an SDP solution is uniform if $\|u_i\|^2 \le 1/d$ for every $u \in X$ and $i \in [d]$.*

Definition 2.3. *Let ξ be a standard Gaussian variable with mean 0 and variance 1. We denote*

$$\begin{aligned}
&\Phi(t) = \Pr\left(|\xi| \le t\right) = \frac{1}{\sqrt{2\pi}} \int_{-t}^{t} e^{-x^2/2} dx, \text{ and} \\
&\bar{\Phi}(t) = 1 - \Phi(t) = \Pr\left(|\xi| > t\right).
\end{aligned}$$

We will use the following lemma, which we prove in Appendix.

Lemma 2.1. *For every $t > 0$ and $\beta \in (0, 1]$, we have*

$$\bar{\Phi}(\beta t) \le \bar{\Phi}(t)^{\beta^2}.$$

We will also use the following result of Šidák [11]:

Theorem 2.1 (Šidák [11]). *Let $\xi_1, \dots, \xi_r$ be Gaussian random variables with mean zero and an arbitrary covariance matrix. Then for any positive $t_1, \dots, t_r$,*

$$\Pr\left(|\xi_1| \le t_1, |\xi_2| \le t_2, \dots, |\xi_r| \le t_r\right) \ge \prod_{i=1}^{r} \Pr\left(|\xi_i| \le t_i\right).$$

3 Rounding Uniform SDP Solutions

In this section, we present a rounding scheme for uniform SDP solutions.

Lemma 3.1. *There is a randomized polynomial-time algorithm that given an instance $\mathcal{I}$ of the MAX CSP$_d$ problem (with $d \ge 57$) and a uniform SDP solution, outputs an assignment x_u such that for every clause $C \in \mathcal{C}$:*

$$\Pr\left(C \text{ is satisfied by } x_u\right) \ge \frac{\min(\|z_C\|^2 |C| d/8, e^{|C|})}{2d^{|C|}}.$$

Proof. We use the following rounding algorithm:

Rounding Scheme for Uniform SDP solutions

Input: an instance of the MAX CSP$_d$ problem and a uniform SDP solution.
Output: an assignment $\{x_u\}$.

- Choose a random Gaussian vector g so that every component of g is distributed as a Gaussian variable with mean 0 and variance 1, and all components are independent.
- For every $u \in V$, let $x'_u = \arg\max_i |\langle u_i, g\rangle|$.
- For every $u \in V$, choose x''_u uniformly at random from $[d]$ (independently for different u).
- With probability 1/2 return assignment $\{x'_u\}$; with probability 1/2 return assignment $\{x''_u\}$.

For every clause C, let us estimate the probabilities that assignments x'_u and x''_u satisfy C. It is clear that x''_u satisfies C with probability $d^{-|C|}$. We prove now that x'_u satisfies C with probability at least $d^{-3|C|/4}$ if $\|z\|_C^2 \ge 8/(|C|d)$.

Claim. Suppose $C \in \mathcal{C}$ is a clause such that $\|z\|_C^2 \ge 8/(|C|d)$ and $d \ge 57$. Then the probability that the assignment x'_u satisfies C is at least $d^{-3|C|/4}$.

Proof. Denote $s = |C|$. We assume without loss of generality that for every $u \in \mathrm{supp}(C)$, $(u, 1) \in C$. Note that for $(u, i) \in C$, we have $\|z_C\|^2 = \langle z_C, u_i\rangle \le \|z_C\| \cdot \|u_i\| \le \|z_C\|/\sqrt{d}$ (here we use that the SDP solution is uniform and therefore $\|u_i\|^2 \le 1/d$). Thus $\|z_C\|^2 \le 1/d$. In particular, $s = |C| \ge 8$ since $\|z\|_C^2 \ge 8/(|C|d)$.

For every $u \in \mathrm{supp}(C)$, let $u_1^\perp = u_1 - z_C$. Let $\gamma_{u,1} = \langle g, u_1^\perp \rangle$ and $\gamma_{u,i} = \langle g, u_i \rangle$ for $i \geq 2$. Let $\gamma_C = \langle g, z_C \rangle$. All variables $\gamma_{u,i}, \gamma_C$ are Gaussian variables. Using that for every two vectors v and w, $\mathbb{E}\left[\langle g, v\rangle \cdot \langle g, w\rangle\right] = \langle v, w\rangle$, we get

$$\begin{aligned}\mathbb{E}\left[\gamma_C \cdot \gamma_{u,1}\right] &= \langle z_C, u_1 - z_C\rangle = \langle z_C, u_1\rangle - \|z_C\|^2 = 0;\\ \mathbb{E}\left[\gamma_C \cdot \gamma_{u,i}\right] &= \langle z_C, u_i\rangle = 0 \quad \text{for } i \geq 2.\end{aligned}$$

Therefore, all variables $\gamma_{u,i}$ are independent from γ_C. (However, for $u', u'' \in \mathrm{supp}(C)$ variables $\gamma_{u',i}$ and $\gamma_{u'',j}$ are not necessarily independent.) Let $M = \bar{\Phi}^{-1}(1/d^{s/2})/\sqrt{sd/8}$. We write the probability that x'_u satisfies C,

$$\begin{aligned}&\Pr(x'_u \text{ satisfies } C) = \Pr\big(\arg\max_i |\langle g, u_i\rangle| = 1 \text{ for every } u \in \mathrm{supp}(C)\big)\\ &\quad= \Pr\left(|\langle g, u_1\rangle| > |\langle g, u_i\rangle| \text{ for every } u \in \mathrm{supp}(C), i \in \{2, \ldots, d\}\right)\\ &\quad= \Pr\left(|\gamma_{u,1} + \gamma_C| > |\gamma_{u,i}| \text{ for every } u \in \mathrm{supp}(C), i \in \{2, \ldots, d\}\right)\\ &\quad\geq \Pr(|\gamma_{u,1}| \leq M/2, \text{ and } |\gamma_{u,i}| \leq M/2\\ &\qquad\qquad \text{for every } u \in \mathrm{supp}(C), i \in \{2, \ldots, d\} \;\big|\; |\gamma_C| > M) \cdot \Pr\left(|\gamma_C| > M\right).\end{aligned}$$

Since all variables $\gamma_{u,i}$ are independent from γ_C,

$$\begin{aligned}&\Pr\left(x'_u \text{ satisfies } C\right) \geq\\ &\quad\Pr\left(|\gamma_{u,i}| \leq M/2 \text{ for every } u \in \mathrm{supp}(C), i \in \{1, \ldots, d\}\right) \cdot \Pr\left(|\gamma_C| > M\right).\end{aligned}$$

By Šidák's Theorem (Theorem 2.1), we have

$$\Pr\left(x'_u \text{ satisfies } C\right) \geq \Big(\prod_{u \in \mathrm{supp}(C)} \prod_{i=1}^{d} \Pr\left(|\gamma_{u,i}| \leq M/2\right)\Big) \cdot \Pr\left(|\gamma_C| > M\right). \quad (1)$$

We compute the variance of vectors $\gamma_{u,i}$. We use that $\mathrm{Var}[\langle g, v\rangle] = \|v\|^2$ for every vector v and that the SDP solution is uniform.

$$\begin{aligned}\mathrm{Var}[\gamma_{u,1}] &= \|u_1^\perp\|^2 = \|u_1 - z_C\|^2 = \|u_1\|^2 - 2\langle u_1, z_C\rangle + \|z_C\|^2\\ &= \|u_1\|^2 - \|z_C\|^2 \leq \|u_1\|^2 \leq 1/d;\\ \mathrm{Var}[\gamma_{u,i}] &= \|u_i\|^2 \leq 1/d \quad \text{for } i \geq 2.\end{aligned}$$

Hence since $\Phi(t)$ is an increasing function and $\bar{\Phi}(\beta t) \leq \bar{\Phi}(t)^{\beta^2}$ (by Lemma 2.1), we have

$$\begin{aligned}\Pr\left(|\gamma_{u,i}| \leq M/2\right) &= \Phi(M/(2\sqrt{\mathrm{Var}[\gamma_{u,i}]})) \geq \Phi(\sqrt{d}M/2) = 1 - \bar{\Phi}(\sqrt{d}\,M/2)\\ &\geq 1 - \bar{\Phi}(\sqrt{sd/8}\,M)^{2/s} = 1 - (d^{-s/2})^{2/s} = 1 - d^{-1}\end{aligned}$$

(recall that we defined M so that $\bar{\Phi}(\sqrt{sd/8}\,M) = d^{-s/2}$). Similarly, $\mathrm{Var}[\gamma_C] = \|z_C\|^2 \geq 8/(sd)$ (by the condition of the lemma). We get (using the fact that $\bar{\Phi}(t)$ is a decreasing function),

$$\Pr\left(|\gamma_C| > M\right) = \bar{\Phi}(M/\sqrt{\mathrm{Var}[\gamma_C]}) \geq \bar{\Phi}(M\sqrt{sd/8}) = d^{-s/2}.$$

Plugging in bounds for $\Pr(|\gamma_{u,i}| \le M/2)$ and $\Pr(|\gamma_C| > M)$ into (1), we obtain

$$\Pr(x'_u \text{ satisfies } C) \ge (1-d^{-1})^{ds} d^{-s/2} \ge d^{-3s/4}.$$

Here, we used that $(1-d^{-1})^d \ge d^{-1/4}$ for $d \ge 57$ (the inequality $(1-d^{-1})^d \ge d^{-1/4}$ holds for $d \ge 57$ since it holds for $d = 57$ and the left hand side, $(1-d^{-1})^d$, is an increasing function, the right hand side, $d^{-1/4}$, is a decreasing function). □

We conclude that if $\|z_C\|^2 \le 8/(|C|d)$ then the algorithm chooses assignment x''_u with probability $1/2$ and this assignment satisfies C with probability at least $1/d^{|C|} \ge \|z_C\|^2 |C|\, d/(8\, d^{|C|})$. So C is satisfied with probability at least, $1/d^{|C|} \ge \|z_C\|^2 |C|\, d/(16\, d^{|C|})$; if $\|z_C\|^2 \ge 8/(|C|d)$ then the algorithm chooses assignment x' with probability $1/2$ and this assignment satisfies C with probability at least $d^{-3|C|/4} \ge e^{|C|}/d^{|C|}$ (since $e \le 57^{1/4} \le d^{1/4}$). In either case,

$$\Pr(C \text{ is satisfied}) \ge \frac{\min(\|z_C\|^2 |C| d/8, e^{|C|})}{2d^{|C|}}.$$ □

Remark 3.1. We note that we did not try to optimize all constants in the statement of Lemma 3.1. By choosing all parameters in our proof appropriately, it is possible to show that for every constant $\varepsilon > 0$, there is a randomized rounding scheme, $\delta > 0$ and d_0 such that for every instance of MAX CSP_d with $d \ge d_0$ the probability that each clause C is satisfied is at least $\min((1-\varepsilon)\|z_C\|^2 \cdot |C|\, d, \delta \cdot e^{\delta|C|})/d^{|C|}$.

4 Rounding Arbitrary SDP Solutions

In this section, we show how to round an arbitrary SDP solution.

Lemma 4.1. *There is a randomized polynomial-time algorithm that given an instance $\mathcal{I}$ of the MAX CSP_d problem (with $d \ge 113$) and an SDP solution, outputs an assignment x_u such that for every clause $C \in \mathcal{C}$:*

$$\Pr(C \textit{ is satisfied by } x_u) \ge \frac{\min(\|z_C\|^2 |C| d/64, 2e^{|C|/8})}{4d^{|C|}}.$$

Proof. For every index u, we sort all vectors u_i according to their length. Let S_u be the indices of $\lceil d/2 \rceil$ shortest vectors among u_i, and $L_u = [d] \setminus S_u$ be the indices of $\lfloor d/2 \rfloor$ longest vectors among u_i (we break ties arbitrarily). For every clause C let $r(C) = |\{(u,i) \in C : i \in S_u\}|$.

Claim. For every $i \in S_u$, we have $\|u_i\|^2 \le 1/|S_u|$.

Proof. Let $i \in S_u$. Note that $\|u_i\|^2 + \sum_{j \in L_u} \|u_j\|^2 \le 1$ (this follows from SDP constraints). There are at least $\lceil d/2 \rceil$ terms in the sum, and $\|u_i\|^2$ is the smallest among them (since $i \in S_u$). Thus $\|u_i\|^2 \le 1/\lceil d/2 \rceil = 1/|S_u|$. □

We use a combination of two rounding schemes: one of them works well on clauses C with $r(C) \geq |C|/4$, the other on clauses C with $r(C) \leq |C|/4$.

Lemma 4.2. *There is a polynomial-time randomized rounding algorithm that given an MAX CSP$_d$ instance $\mathcal{I}$ with $d \geq 113$ outputs an assignment x_u such that every clause C with $r(C) \geq |C|/4$ is satisfied with probability at least*

$$\frac{\min(\|z_C\|^2\,|C|\,d/64, e^{|C|/4})}{2d^{|C|}}.$$

Proof. We will construct a sub-instance $\mathcal{I}'$ with a uniform SDP solution and then solve $\mathcal{I}'$ using Lemma 3.1. To this end, we first construct a partial assignment x_u. For every $u \in X$, with probability $|L_u|/d = \lfloor d/2 \rfloor/d$, we assign a value to x_u uniformly at random from L_u; with probability $1 - |L_u|/d = |S_u|/d$, we do not assign any value to x_u. Let $A = \{u : x_u \text{ is assigned}\}$. Let us say that a clause C *survives* the partial assignment step if for every $(u,i) \in C$ either $u \in A$ and $i = x_u$, or $u \notin A$ and $i \in S_u$.

The probability that a clause C survives is

$$\prod_{(u,i)\in C, i\in L_u} \Pr\left(x_u \text{ is assigned value } i\right) \prod_{(u,i)\in C, i\in S_u} \Pr\left(x_u \text{ is unassigned}\right) =$$

$$\left(\frac{\lfloor d/2 \rfloor}{d} \cdot \frac{1}{\lfloor d/2 \rfloor}\right)^{|C|-r(C)} \cdot \left(\frac{\lceil d/2 \rceil}{d}\right)^{r(C)} = \frac{\lceil d/2 \rceil^{r(C)}}{d^{|C|}}.$$

For every survived clause C, let $C' = \{(u,i) : u \notin A\}$. Note that for every $(u,i) \in C'$, we have $i \in S_u$. We get a sub-instance $\mathcal{I}'$ of our problem on the set of unassigned variables $\{x_u : u \notin A\}$ with the set of clauses $\{C' : C \in \mathcal{C} \text{ survives}\}$. The length of each clause C' equals $r(C)$. In sub-instance $\mathcal{I}'$, we require that each variable x_u takes values in S_u. Thus $\mathcal{I}'$ is an instance of MAX CSP$_{d'}$ problem with $d' = |S_u| = \lceil d/2 \rceil$.

Now we transform the SDP solution for $\mathcal{I}$ to an SDP solution for $\mathcal{I}'$: we let $z_{C'} = z_C$ for survived clauses C, remove vectors u_i for all $u \in A$, $i \in [d]$ and remove vectors z_C for non-survived clauses C. By Claim 4, this SDP solution is a uniform solution for $\mathcal{I}'$ (i.e. $\|u_i\| \leq 1/d'$ for every $u \notin A$ and $i \in S_i$; note that $\mathcal{I}'$ has alphabet size d'). We run the rounding algorithm from Lemma 3.1. The algorithm assigns values to unassigned variables x_u. For every *survived* clause C, we get

$$\begin{aligned}
\Pr\left(C \text{ is satisfied by } x_u\right) &= \Pr\left(C' \text{ is satisfied by } x_u\right) \\
&\geq \frac{\min(\|z_C\|^2 |C'| d'/8, e^{|C'|})}{2d'^{|C'|}} \\
&= \frac{\min(\|z_C\|^2 r(C) d'/8, e^{r(C)})}{2d'^{r(C)}} \\
&\geq \frac{\min(\|z_C\|^2 |C| d/64, e^{|C|/4})}{2d'^{r(C)}}.
\end{aligned}$$

Therefore, for every clause C,

$$\begin{aligned}\Pr(C \text{ is satisfied by } x_u) &\geq \Pr(C \text{ is satisfied by } x_u \mid C \text{ survives})\Pr(C \text{ survives})\\ &\geq \frac{\min(\|z_C\|^2|C|d/64, e^{|C|/4})}{2d'^{r(C)}} \times \frac{\lceil d/2\rceil^{r(C)}}{d^{|C|}}\\ &= \frac{\min(\|z_C\|^2|C|d/64, e^{|C|/4})}{2d^{|C|}}.\end{aligned}$$

□

Finally, we describe an algorithm for clauses C with $r(C) \leq |C|/4$.

Lemma 4.3. *There is a polynomial-time randomized rounding algorithm that given an MAX CSP$_d$ instance $\mathcal{I}$ outputs an assignment x_u such that every clause C with $r(C) \leq |C|/4$ is satisfied with probability at least $e^{|C|/8}/d^{|C|}$.*

Proof. We do the following independently for every vertex $u \in X$. With probability $3/4$, we choose x_u uniformly at random from L_u; with probability $1/4$, we choose x_u uniformly at random from S_u. The probability that a clause C with $r(C) \leq |C|/4$ is satisfied equals

$$\begin{aligned}\prod_{(u,i)\in C, i\in L_u} \frac{3}{4|L_u|} \prod_{(u,i)\in C, i\in S_u} \frac{1}{4|S_u|} &= \frac{1}{d^{|C|}} \cdot \left(\frac{3d}{4|L_u|}\right)^{|C|-r(C)} \left(\frac{d}{4|S_u|}\right)^{r(C)}\\ &\geq \frac{1}{d^{|C|}} \cdot \left(\frac{3d}{4|L_u|}\right)^{3|C|/4} \left(\frac{d}{4|S_u|}\right)^{|C|/4}\\ &\geq \frac{1}{d^{|C|}} \cdot \left(\left(\frac{3}{2}\right)^{3/4} \left(\frac{d}{2(d+1)}\right)^{1/4}\right)^{|C|}.\end{aligned}$$

Note that $\left(\frac{3}{2}\right)^{3/4}\left(\frac{d}{2(d+1)}\right)^{1/4} \geq \left(\frac{3}{2}\right)^{3/4}\left(\frac{113}{2\cdot 114}\right)^{1/4} \geq e^{1/8}$. Therefore, the probability that the clause is satisfied is at least $e^{|C|/8}/d^{|C|}$. □

We run the algorithm from Lemma 4.2 with probability $1/2$ and the algorithm from Lemma 4.3 with probability $1/2$. Consider a clause $C \in \mathcal{C}$. If $r(C) \geq |C|/4$, we satisfy C with probability at least $\frac{\min(\|z_C\|^2|C|d/64, e^{|C|/4})}{4d^{|C|}}$. If $r(C) \leq |C|/4$, we satisfy C with probability at least $e^{|C|/8}/(2d^{|C|})$. So we satisfy every clause C with probability at least $\frac{\min(\|z_C\|^2|C|d/64, 2e^{|C|/8})}{4d^{|C|}}$. □

5 Approximation Algorithm for MAX k-CSP$_d$

In this section, we present the main result of the paper.

Theorem 5.1. *There is a polynomial-time randomized approximation algorithm for MAX k-CSP$_d$ that given an instance $\mathcal{I}$ finds an assignment that satisfies at least $\Omega(\min(kd, e^{k/8})\, OPT(\mathcal{I})/d^k)$ clauses with constant probability.*

Proof. If $d \leq 113$, we run the algorithm of Charikar, Makarychev and Makarychev [3] and get $\Omega(k/d^k)$ approximation. So we assume below that $d \geq 113$. We also assume that $kd/d^k \geq 1/|\mathcal{C}|$, as otherwise we just choose one clause from $\mathcal{C}$ and find an assignment that satisfies it. Thus d^k is polynomial in the size of the input.

We solve the SDP relaxation for the problem and run the rounding scheme from Lemma 4.1 d^k times. We output the best of the obtained solutions. By Lemma 4.1, each time we run the rounding scheme we get a solution with expected value at least

$$\sum_{C\in\mathcal{C}} \frac{\min(\|z_C\|^2|C|d/64, 2e^{|C|/8})}{4d^{|C|}} \geq \sum_{C\in\mathcal{C}} \frac{\min(kd/64, 2e^{k/8})}{4d^k}\|z_C\|^2$$
$$\geq \frac{\min(kd/64, 2e^{k/8})}{4d^k} SDP(\mathcal{I}) \geq \frac{\min(kd/64, 2e^{k/8})}{4d^k} OPT(\mathcal{I}).$$

Denote $\alpha = \frac{\min(kd/64, 2e^{k/8})}{4d^k}$. Let Z be the random variable equal to the number of satisfied clauses. Then $\mathbb{E}[Z] \geq \alpha OPT(\mathcal{I})$, and $Z \leq OPT(\mathcal{I})$ (always). Let $p = \Pr(Z \leq \alpha OPT(\mathcal{I})/2)$. Then

$$p \cdot (\alpha OPT(\mathcal{I})/2) + (1-p) \cdot OPT(\mathcal{I}) \geq \mathbb{E}[Z] \geq \alpha OPT(\mathcal{I}).$$

So $p \leq \frac{1-\alpha}{1-\alpha/2} = 1 - \frac{\alpha}{2-\alpha}$. So with probability at least $1 - p \geq \frac{\alpha}{2-\alpha}$, we find a solution of value at least $\alpha OPT(\mathcal{I})/2$ in one iteration. Since we perform $d^k > 1/\alpha$ iterations, we find a solution of value at least $\alpha OPT(\mathcal{I})/2$ with constant probability. □

References

1. Austrin, P., Mossel, E.: Approximation Resistant Predicates from Pairwise Independence. Computational Complexity 18(2), 249–271 (2009)
2. Charikar, M., Makarychev, K., Makarychev, Y.: Near-Optimal Algorithms for Unique Games. In: Proceedings of the 38th ACM Symposium on Theory of Computing, pp. 205–214 (2006)
3. Charikar, M., Makarychev, K., Makarychev, Y.: Near-Optimal Algorithms for Maximum Constraint Satisfaction Problems. ACM Transactions on Algorithms 5(3) (July 2009)
4. Engebretsen, L.: The Nonapproximability of Non-Boolean Predicates. SIAM Journal on Discrete Mathematics 18(1), 114–129 (2004)
5. Engebretsen, L., Holmerin, J.: More Efficient Queries in PCPs for NP and Improved Approximation Hardness of Maximum CSP. In: Diekert, V., Durand, B. (eds.) STACS 2005. LNCS, vol. 3404, pp. 194–205. Springer, Heidelberg (2005)
6. Guruswami, V., Raghavendra, P.: Constraint Satisfaction over a Non-Boolean Domain: Approximation Algorithms and Unique-Games Hardness. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 77–90. Springer, Heidelberg (2008)

7. Hast, G.: Approximating Max kCSP — Outperforming a Random Assignment with Almost a Linear Factor. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 956–968. Springer, Heidelberg (2005)
8. Raghavendra, P.: Optimal Algorithms and Inapproximability Results For Every CSP? In: Proceeding of the ACM Symposium on Theory of Computing, STOC (2008)
9. Samorodnitsky, A., Trevisan, L.: A PCP characterization of NP with optimal amortized query complexity. In: Proceedings of the ACM Symposium on Theory of Computing (STOC), pp. 191–199 (2000)
10. Samorodnitsky, A., Trevisan, L.: Gowers Uniformity, Influence of Variables, and PCPs. In: Proceedings of the 38th ACM Symposium on Theory of Computing, pp. 11–20 (2006)
11. Šidák, Z.: Rectangular Confidence Regions for the Means of Multivariate Normal Distributions. Journal of the American Statistical Association 62(318), 626–633 (1967)
12. Trevisan, L.: Parallel Approximation Algorithms by Positive Linear Programming. Algorithmica 21(1), 72–88 (1998)

A Proof of Lemma 2.1

In this section, we prove Lemma 2.1. We will use the following fact.

Lemma A.1 (see e.g. [2]). *For every $t > 0$,*

$$\frac{2t}{\sqrt{2\pi}\,(t^2+1)}e^{-\frac{t^2}{2}} < \bar{\Phi}(t) < \frac{2}{\sqrt{2\pi}\,t}e^{-\frac{t^2}{2}}.$$

Lemma 2.1. *For every $t > 0$ and $\beta \in (0, 1]$, we have*

$$\bar{\Phi}(\beta t) \le \bar{\Phi}(t)^{\beta^2}.$$

Proof. Rewrite the inequality we need to prove as follows: $(\bar{\Phi}(\beta t))^{1/\beta^2} \le \bar{\Phi}(t)$. Denote the left hand side by $f(\beta, t)$:

$$f(\beta, t) = \bar{\Phi}(\beta t)^{1/\beta^2}.$$

We show that for every $t > 0$, $f(\beta, t)$ is strictly increasing function as a function of $\beta \in (0, 1]$. Then,

$$(\bar{\Phi}(\beta t))^{1/\beta^2} = f(\beta) < f(1) = \bar{\Phi}(t).$$

We first prove that $\frac{\partial f(1,t)}{\partial \beta} > 0$. Write,

$$\frac{\partial f(1,t)}{\partial \beta} = -2\log(\bar{\Phi}(t))\,\bar{\Phi}(t) + t\bar{\Phi}'(t) = -2\log(\bar{\Phi}(t))\,\bar{\Phi}(t) - \frac{2t\,e^{-t^2/2}}{\sqrt{2\pi}}.$$

Consider three cases. If $t \geq \sqrt{\frac{2e}{\pi}}$, then, by Lemma A.1,

$$\bar{\Phi}(t) < \frac{2}{\sqrt{2\pi}t} e^{-t^2/2} \leq e^{-1/2} e^{-t^2/2} = e^{-(t^2+1)/2}.$$

Hence, $-2\log(\bar{\Phi}(t)) > (t^2+1)$, and by Lemma A.1,

$$-2\log(\bar{\Phi}(t))\,\bar{\Phi}(t) > (t^2+1)\,\bar{\Phi}(t) > \frac{2t\,e^{-t^2/2}}{\sqrt{2\pi}}.$$

If $t < \sqrt{\frac{2e}{\pi}}$, then let $\rho(x) = -\log x/(1-x)$ for $x \in (0,1)$ and write,

$$-\log\bar{\Phi}(t) = \rho(\bar{\Phi}(t)) \cdot (1-\bar{\Phi}(t)) = \frac{\rho(\bar{\Phi}(t))}{\sqrt{2\pi}} \int_{-t}^{t} e^{-x^2/2} dx \geq \frac{2\rho(\bar{\Phi}(t))te^{-t^2/2}}{\sqrt{2\pi}}.$$

Hence,

$$\frac{\partial f(1,t)}{\partial\beta} = -2\log(\bar{\Phi}(t))\,\bar{\Phi}(t) - \frac{2t\,e^{-t^2/2}}{\sqrt{2\pi}} \geq \frac{2te^{-t^2/2}}{\sqrt{2\pi}} \times (2\rho(\bar{\Phi}(t))\bar{\Phi}(t) - 1).$$

For $x \in [1/3, 1]$, $2\rho(x)x > 1$, since the function $\rho(x)x$ is increasing and $\rho(1/3) > 3/2$. Hence $2\rho(\bar{\Phi}(t))\bar{\Phi}(t) > 1$, if $\bar{\Phi}(t) \geq 1/3$.

The remaining case is $t < \sqrt{\frac{2e}{\pi}}$ and $\bar{\Phi}(t) < 1/3$. Then, $\bar{\Phi}(t) \geq \bar{\Phi}(\sqrt{\frac{2e}{\pi}}) > 1/6$ and hence $\bar{\Phi}(t) \in (1/6, 1/3)$. Since the function $-x\log x$ is increasing on the interval $(0, e^{-1})$,

$$-2\log(\bar{\Phi}(t))\,\bar{\Phi}(t) > -2\log(1/6) \cdot \frac{1}{6} > \frac{1}{2}.$$

The function $te^{-t^2/2}$ attains its maximum at $t = 1$, thus

$$\frac{2t\,e^{-t^2/2}}{\sqrt{2\pi}} \leq \frac{2e^{-1/2}}{\sqrt{2\pi}} < \frac{1}{2}.$$

We get

$$\frac{\partial f(1,t)}{\partial\beta} = -2\log(\bar{\Phi}(t))\,\bar{\Phi}(t) - \frac{2t\,e^{-t^2/2}}{\sqrt{2\pi}} > 0.$$

Since $\frac{\partial f(1,t)}{\partial\beta} > 0$, for every $t > 0$, there exists $\varepsilon_0 > 0$, such that for all $\varepsilon \in (0, \varepsilon_0)$, $f(1-\varepsilon, t) < f(1,t)$. Particularly, for $t' = \beta t$,

$$f(\beta, t) = f(1, t')^{1/\beta^2} \geq f(1-\varepsilon, t')^{1/\beta^2} = f((1-\varepsilon)\beta, t). \qquad \square$$

Planarizing an Unknown Surface

Yury Makarychev* and Anastasios Sidiropoulos

Toyota Technological Institute at Chicago
{yury,tasos}@ttic.edu

Abstract. It has been recently shown that any graph of genus $g > 0$ can be stochastically embedded into a distribution over planar graphs, with distortion $O(\log(g+1))$ [Sidiropoulos, FOCS 2010]. This embedding can be computed in polynomial time, provided that a drawing of the input graph into a genus-g surface is given.

We show how to compute the above embedding without having such a drawing. This implies a general reduction for solving problems on graphs of small genus, even when the drawing into a small genus surface is unknown. To the best of our knowledge, this is the first result of this type.

1 Introduction

The genus of a graph is a parameter that quantifies how far it is from being planar. Informally, a graph has genus g, for some $g \geq 0$, if it can be drawn without any crossings on the surface of a sphere with g additional handles (see Section 1.4). For example, a planar graph has genus 0, and a graph that can be drawn on a torus has genus at most 1.

Planar graphs exhibit properties that give rise to improved algorithmic solutions for numerous problems (see, for example [Bak94]). Because of their similarities to planar graphs, graphs of small genus enjoy similar algorithmic characteristic. More precisely, algorithms for planar graphs can usually be extended to graphs of bounded genus, with a small loss in efficiency or quality of the solution (e.g. [CEN09]).

Unfortunately, such extensions typically suffer from two main difficulties. First, for different problems, one typically needs to develop complicated, and ad-hoc techniques. Second, a perhaps more challenging issue is that essentially all known algorithms for graphs of small genus require that a drawing of the input graph into a small genus surface is given. In general, computing a drawing of a graph into a surface of minimum genus is NP-hard [Tho89, Tho93]. Moreover, the currently best-known approximation algorithm for this problem is only a trivial $O(n)$-approximation that follows by bounds on the Euler characteristic. This has been improved to $O(\sqrt{n})$-approximation for graphs of bounded degree [CKK97].

The first of the above two obstacles has been recently addressed for some problems by Sidiropoulos [Sid10], who showed that any graph of genus $g > 0$ can

* Yury Makarychev is supported in part by the NSF Career Award CCF-1150062.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 266–275, 2012.
© Springer-Verlag Berlin Heidelberg 2012

be embedded into a distribution over planar graphs, with distortion $O(\log(g+1))$ (see Section 1.4 for definitions). This result implies a general reduction for a large class of geometric optimization problems from instances on genus-g graphs, to corresponding ones on planar graphs, with a $O(\log(g+1))$ loss factor in the approximation guarantee.

Unfortunately, the algorithm from [Sid10] can compute the above embedding in polynomial time, only if a drawing of the input graph into a small genus surface is given. We show how to compute this embedding even when the drawing of the input graph is unknown. In particular, this implies that the above reduction for solving problems on graphs of small genus, can be performed even on graphs for which we don't have a drawing into a small genus surface. The statement of our main embedding result follows.

Theorem 1.1 (Main result). *There exists a polynomial time algorithm which given a graph G of genus $g > 0$, computes a stochastic embedding of G into planar graphs, with distortion $O(\log(g+1))$. In particular, the algorithm does not require a drawing of G as part of the input.*

1.1 Applications

The main application of our result is a general reduction from a class of optimization problems on genus-g graphs, to their restriction on planar graphs. This is the same reduction obtained in [Sid10], only here we don't require a drawing of the input graph. For completeness, we state precisely the reduction, as given in [Sid10] (see also [Bar96]). Let V be a set, $\mathcal{I} \subset \mathbb{R}_+^{V \times V}$ a set of non-negative vectors corresponding to all feasible solutions for a minimization problem, and $c \in \mathbb{R}_+^{V \times V}$. Then, we define the *linear minimization problem* $(\mathcal{I}, c)$ to be the computational problem where we are given a graph $G = (V, E)$, and we are asked to find $s \in \mathcal{I}$, minimizing

$$\sum_{\{u,v\} \in V \times V} c_{u,v} \cdot s_{u,v} \cdot d(u,v)$$

Observe that this definition captures a very general class of problems. For example, MST can be encoded by letting $\mathcal{I}$ be the set of indicator vectors of the edges of all spanning trees on V, and c the all-ones vector. Similarly, one can easily encode problems such as TSP, Facility-Location, k-Server, Bi-Chromatic Matching, etc.

The main Corollary of our embedding result can now be stated as follows.

Corollary 1.1. *Let $\Pi = (\mathcal{I}, c)$ be a linear minimization problem. If there exists a polynomial-time α-approximation algorithm for Π on planar graphs, then there exists a randomized polynomial-time $O(\alpha \cdot \log(g+1))$-approximation algorithm for Π on graphs of genus $g > 0$, even when the drawing of the input graph is unknown.*

1.2 Overview of the Algorithm

We now give a high-level overview of our algorithm. Consider a graph $G = (V, E)$. Let us say that a collection $\mathcal{P}$ of shortest paths in G is a *planarizing set of paths*, if the graph $G \setminus \bigcup_{P \in \mathcal{P}} V(P)$ is planar. It was shown by Sidiropoulos [Sid10] that any graph having a planarizing set of paths of size k, admits a stochastic embedding into planar graphs, with distortion $O(\log k)$. Moreover, given such a set of planarizing paths, the embedding can be computed in polynomial time. It follows by the work of Eppstein [Epp03], and Erickson and Whittlesey [EW05], that for any graph G of genus g, that there exists a planarizing set of paths, of size $O(g)$. However, all known algorithms for computing this planarizing set require a drawing of the graph into a surface of genus g. Since we don't know how to compute a drawing of a graph into a minimum-genus surface in polynomial time, all known algorithms are not applicable in our case.

Our main technical contribution is showing how to compute in polynomial time a planarizing set of paths of approximately optimal size (up to a $\operatorname{polylog} n$ factor) in an arbitrary graph. For a graph G, we say that a collection $\mathcal{Q}$ of shortest paths having a common endpoint is a *balanced set of paths* if $\bigcup_{Q \in \mathcal{Q}} V(Q)$ is a balanced vertex-separator of G. That is, removing all paths in $\mathcal{Q}$ from G, leaves a graph where every connected component is at most half the size of G. Our high-level approach is as follows. We find and remove a "small" balanced set of paths in G. Then we compute connected components in the obtained graph. In each non-planar connected component, we again find and remove a balanced set of paths. We repeat this procedure until all components are planar. Finally, we output the planarizing set of paths that consists of all paths that we removed from the graph.

In order for this approach to work, we first prove that in a (possibly vertex-weighted) graph G of genus g, there exists a balanced set Q of paths of size $O(g)$. Next, we show how to compute in polynomial time a balanced set of paths of approximately optimal size in an arbitrary graph G. As outlined above, we then recursively use this as a subroutine to find a set $\mathcal{P}$ of planarizing paths. We begin with a graph G of genus g (for which we don't have a drawing into a genus-g surface), and inductively build $\mathcal{P}$ in steps. At the first step, we compute a balanced set $\mathcal{Q}_1$ of paths in G. We add these paths to $\mathcal{P}$. At every subsequent step $i > 1$, let G_i be the graph obtained from G after removing all the paths we have computed so far, i.e. $G_i = G \setminus \bigcup_{P \in \mathcal{P}} V(P)$. Since G has genus g, graph G_i has at most $O(g)$ non-planar connected components. For every such non-planar component, we compute a balanced set of paths and add it to $\mathcal{P}$. We show that after every step, the size of the largest non-planar component reduces by at least a constant factor. Therefore, after $O(\log n)$ steps, we obtain the desired planarizing set of paths.

1.3 Related Work

Inspired by Bartal's stochastic embedding of general metrics into trees [Bar96], Indyk and Sidiropoulos [IS07] showed that every metric on a graph of genus g

can be stochastically embedded into a planar graph with distortion $2^{O(g)}$ (see Section 1.4 for a formal definition of stochastic embeddings). The above bound was later improved by Borradaile, Lee, and Sidiropoulos [BLS09], who obtained an embedding with distortion $g^{O(1)}$. Subsequently, Sidiropoulos [Sid10] gave an embedding with distortion $O(\log g)$, matching the $\Omega(\log g)$ lower bound from [BLS09]. The embeddings from [IS07], and [Sid10] can be computed in polynomial time, provided that the drawing of the graph into a small genus surface is given. Computing the embedding from [BLS09] requires solving an NP-hard problem, even when the drawing is given.

1.4 Preliminaries

Throughout the paper, we consider graphs with non-negative edge lengths. For a tree T with root $r \in V(T)$, and for $v \in V(T)$ we denote by $T(v)$ the unique path in T between v and r.

Graphs on surfaces. Let us recall some notions from topological graph theory (an in-depth exposition can be found in [MT01]). A *surface* is a compact connected 2-dimensional manifold, without boundary. For a graph G we can define a one-dimensional simplicial complex C associated with G as follows: The 0-cells of C are the vertices of G, and for each edge $\{u, v\}$ of G, there is a 1-cell in C connecting u and v. A *drawing* of G on a surface S is a continuous injection $f : C \to S$. The *genus* of a surface $\mathcal{S}$ is the maximum cardinality of a collection of simple closed non-intersecting curves $C_1, \ldots, C_k$ in $\mathcal{S}$, such that $\mathcal{S} \setminus (C_1 \cup \cdots \cup C_k)$ is connected. The genus of a graph G is the minimum k, such that G can be drawn into a surface of genus k. Note that a graph of genus 0 is a planar graph. We remark that we make no distinction between orientable, and non-orientable genus, since all of our results hold in both settings.

Metric embeddings. A mapping $f : X \to Y$ between two metric spaces (X, d) and (Y, d') is *non-contracting* if $d'(f(x), f(y)) \geq d(x, y)$ for all $x, y \in X$. If (X, d) is any finite metric space, and $\mathcal{Y}$ is a family of finite metric spaces, we say that (X, d) *admits a stochastic D-embedding into* $\mathcal{Y}$ if there exists a random metric space $(Y, d') \in \mathcal{Y}$ and a random non-contracting mapping $f : X \to Y$ such that for every $x, y \in X$,

$$\mathbb{E}\left[d'(f(x), f(y))\right] \leq D \cdot d(x, y). \tag{1}$$

The infimal D such that (1) holds is the *distortion of the stochastic embedding.* A detailed exposition of results on metric embeddings can be found in [Ind01] and [Mat02].

2 Path Separators in Embedded Graphs

For a graph G, a real $\alpha \in (0, 1/2]$, and a set $X \subseteq V(G)$ we say that X is an *α-balanced vertex separator* for G if every connected component of $G \setminus X$ contains at most $\alpha \cdot |V(G)|$ vertices. It is also called simply balanced vertex separator, when $\alpha = 1/2$.

For a vertex-weighted graph G with weight function $w : V(G) \to \mathbb{R}_{\geq 0}$, for every $Y \subseteq V(G)$ we use the notation $w(Y) = \sum_{v \in V(G)} w(v)$. Similarly to the unweighted case, we say that a set $X \subseteq V(G)$ is a balanced vertex separator for a weighted graph (G, w) if for every connected component C of $G \setminus X$ we have $w(V(C)) \leq w(V(G))/2$.

Theorem 2.1 (Lipton & Tarjan [LT79], Thorup [Tho04]). *Let G be a planar graph, let $r \in V(G)$, and let T be a spanning tree of G with root r. Then, there exist $v_1, v_2, v_3 \in V(G)$, such that $V(T(v_1) \cup T(v_2) \cup T(v_3))$ is a balanced vertex separator for G. Moreover, the vertices v_1, v_2 and v_3 can be computed in polynomial time.*

We will use a slight modification of Theorem 2.1, for the case of weighted graphs. The proof is a straightforward extension to the one due to Thorup [Tho04], which is based on the argument of Lipton and Tarjan [LT79].

Lemma 2.1. *Let G be a planar graph, let $r \in V(G)$, and let T be a spanning tree of G with root r. Let $w : V(G) \to \mathbb{R}_{\geq 0}$. Then, there exist $v_1, v_2, v_3 \in V(G)$, such that $V(T(v_1) \cup T(v_2) \cup T(v_3))$ is a balanced vertex separator for (G, w). Moreover, the vertices v_1, v_2 and v_3 can be computed in polynomial time.*

The next Theorem follows by the work of Eppstein [Epp03], and Erickson & Whittlesey [EW05].

Theorem 2.2 (Erickson & Whittlesey [EW05], Eppstein [Epp03]). *Let G be a graph of genus $g > 0$, and let φ be an embedding of G into a surface $\mathcal{S}$ of genus g. Let $r \in V(G)$, and let T be a spanning tree of G with root r. Then, there exist edges $\{x_1, y_1\}, \ldots, \{x_{2g}, y_{2g}\} \in E(G)$, such that $G \setminus \bigcup_{i=1}^{2g} V(T(x_i) \cup T(y_i))$ is planar. Moreover, the topological space $\mathcal{S} \setminus \bigcup_{i=1}^{2g} \varphi(T(x_i) \cup T(y_i) \cup \{x_i, y_i\})$ is homeomorphic to an open disk.*

We are now ready to prove the main result of this section.

Lemma 2.2 (Existence of path separators in embedded graphs). *Let G be a weighted graph of genus g, with weight function $w : V(G) \to \mathbb{R}_{\geq 0}$. Let $r \in V(G)$, and let T be a spanning tree of G with root r. Then, there exists $X \subseteq V(G)$, with $|X| \leq 4g + 3$, such that $\bigcup_{u \in X} V(T(u))$ is a balanced vertex separator for (G, w).*

Proof. The case $g = 0$ follows by Lemma 2.1, so we may assume that $g > 0$. Fix an embedding φ of G into a surface $\mathcal{S}$ of genus g. By Theorem 2.2 there exist $\{x_1, y_1\}, \ldots, \{x_{2g}, y_{2g}\} \in E(G)$, such that the topological space $\mathcal{S} \setminus \bigcup_{i=1}^{2g} \varphi(T(x_i) \cup T(y_i) \cup \{x_i, y_i\})$ is homeomorphic to an open disk. Let

$$H = \bigcup_{i=1}^{2g} T(x_i) \cup T(y_i) \cup \{x_i, y_i\}.$$

Note that $r \in V(H)$. Let G' be the graph obtained from G by contracting H into a single vertex r'. Since $\mathcal{S} \setminus \varphi(H)$ is an open disk, it follows that G' is planar.

Let T' be the subgraph of G' induced by T after contracting H. Since T is a spanning subgraph of G, it follows that T' is a spanning subgraph of G'. Indeed, the set of vertices $V(H)$ spans is a connected subtree of T. Therefore, after contracting H, the subgraph T' induced by T is still a tree. Thus, T' is a spanning subtree of G'. We consider T' being rooted at r'.

Define a weight function $w' : V(G') \to \mathbb{R}_{\geq 0}$ such that for every $v \in V(G')$,

$$w'(v) = \begin{cases} w(v), & \text{if } v \neq r' \\ 0, & \text{if } v = r' \end{cases}$$

By Lemma 2.1 it follows that there exist $v_1, v_2, v_3 \in V(G')$ such that $V(T'(v_1) \cup T'(v_2) \cup T'(v_3))$ is a balanced vertex separator for (G', w').

Let $J = G \setminus V(H)$. Observe that $J = G \setminus V(H) = G' \setminus \{r'\}$. Moreover, for any $v \in V(J)$, we have $T(v) \cap J = T'(v) \cap J$. Thus, the set of connected components of $(G \setminus V(H)) \setminus V(T(v_1) \cup T(v_2) \cup T(v_3))$ is the same as the set of connected components of $(G' \setminus \{r'\}) \setminus V(T'(v_1) \cup T'(v_2) \cup T'(v_3))$. Let C be a connected component of $(G \setminus V(H)) \setminus V(T(v_1) \cup T(v_2) \cup T(v_3))$. We have

$$w(C) = w'(C) \leq \frac{1}{2} w(V(G')) = \frac{1}{2}(w(V(G)) - w(V(H))) \leq \frac{1}{2} w(V(G)).$$

Thus, $V(T(v_1) \cup T(v_2) \cup T(v_3)) \cup \bigcup_{i=1}^{2g} V(T(x_i) \cup T(y_i))$ is a balanced vertex separator for (G, w), as required. □

3 Computing Path Separators in Arbitrary Graphs

Recall the definition of a caterpillar decomposition of a tree.

Definition 3.1 (Caterpillar decomposition [Mat99, CS02]). *A* caterpillar decomposition *of a rooted tree T is a family of paths $\mathcal{P} = \{P_i\}$, satisfying the following conditions:*

(i) Every $P_i \in \mathcal{P}$ is a subpath of a root–leaf path.
(ii) For every $P_i \neq P_j \in \mathcal{P}$, we have $V(P_i) \cap V(P_j) = \emptyset$.
(iii) $V(T) = \bigcup_{P_i \in \mathcal{P}} V(P_i)$.

The proof of the following lemma about caterpillar decompositions can be found in [Mat99, CS02].

Lemma 3.1 (See [Mat99, CS02]). *For every rooted tree T, there exists a caterpillar decomposition $\mathcal{P}$, such that every root–leaf path $T(u)$ crosses at most $O(\log n)$ paths from $\mathcal{P}$. Moreover, this decomposition can be found in polynomial time.*

We are now ready to prove that main result of this section.

Lemma 3.2 (Computing approximate path separators). *Let G be a graph, and $w : V(G) \to \mathbb{R}_{\geq 0}$. Let $r \in V(G)$, and let T be a spanning tree of G with*

root r. Suppose that there exists $X \subseteq V(G)$, such that $\bigcup_{u\in X} V(T(u))$ is a balanced vertex separator for (G, w). Then we can compute in polynomial time a set $Y \subseteq V(G)$ with $|Y| \leq O(\log^{3/2} n) \cdot |X|$, such that $\bigcup_{u\in Y} V(T(u))$ is a 3/4-balanced vertex separator for (G, w).

Proof. We reduce the problem to the problem of finding a vertex separator in an auxiliary graph. Using Lemma 3.1 we construct a caterpillar decomposition $\mathcal{P}$ of T such that every root–leaf path $T(u)$ crosses at most $O(\log n)$ paths from $\mathcal{P}$. We define an auxiliary graph $\mathcal{G}$ on the set $\mathcal{P}$ as follows: $P_i \in \mathcal{P}$ and $P_j \in \mathcal{P}$ are connected with an edge in $\mathcal{G}$ if there is an edge between sets $V(P_i)$ and $V(P_j)$ in G. We assign each P_i weight equal to the total weight of all vertices of P_i. Note that then the total weight of all vertices in $\mathcal{G}$ equals $w(V(G))$.

Observe that for every $\mathcal{A} \subset \mathcal{P}$ the induced graph $\mathcal{G}[\mathcal{A}]$ is connected if and only if the induced graph $G[A]$, where $A = \bigcup_{P_i\in\mathcal{A}} V(P_i)$, is connected. Consequently, if $\mathcal{C}_1, \ldots, \mathcal{C}_t$ are connected components of $\mathcal{G} \setminus \mathcal{B}$ (for some $\mathcal{B} \subset \mathcal{P}$) then sets $C_j = \bigcup_{P_i\in\mathcal{C}_j} V(P_i)$ (for $j = 1, \ldots, t$) are connected components of $G \setminus B$ where $B = \bigcup_{P_i\in\mathcal{B}} V(P_i)$; moreover, the weight of each $\mathcal{C}_i$ equals the weight of C_i. Therefore, $\mathcal{B}$ is a balanced vertex separator in $\mathcal{G}$ if and only if $B = \bigcup_{P_i\in\mathcal{B}} V(P_i)$ is a balanced vertex separator in G.

We now prove that there is a balanced vertex separator in $\mathcal{G}$ of size $O(\log n) \cdot |V(G)|$. Let $\mathcal{X} = \bigcup_{u\in X}\{P_i \in \mathcal{P} : P_i \text{ intersects } T(u)\}$. First, we show that $\mathcal{X}$ is a balanced vertex separator in $\mathcal{G}$. Denote $X' = \bigcup_{P_i\in\mathcal{X}} V(P_i)$. Observe that $X' \supset \bigcup_{u\in X} V(T(u))$. Indeed, consider $v \in \bigcup_{u\in X} V(T(u))$. Then $v \in T(u)$ for some $u \in X$. Let P_i be the path in $\mathcal{P}$ that contains v. Then P_i intersects $T(u)$ at vertex v and therefore $P_i \in \mathcal{X}$. Hence $v \in V(P_i) \subset X'$. We conclude that $X' \supset \bigcup_{u\in X} V(T(u))$. Since $\bigcup_{u\in X} V(T(u))$ is a balanced vertex separator in G, set X' is also a balanced vertex separator in G. Hence $\mathcal{X}$ is a balanced vertex separator in $\mathcal{G}$. Now we upper bound the size of $\mathcal{X}$. Note that for every u, we have $|\{P_i \in \mathcal{P} : P_i \text{ intersects } T(u)\}| = O(\log n)$ (by Lemma 3.1). Thus we have, $|\mathcal{X}| = O(\log n) \cdot |X|$. We proved that there is a balanced vertex separator in $\mathcal{G}$ of size $O(\log n) \cdot |X|$.

We use the algorithm of Feige, Hajiaghayi and Lee [FHL08] to find a $O(\sqrt{\log n})$ approximation for the optimal balanced vertex separator in $\mathcal{G}$. We get a 3/4-balanced vertex separator $\mathcal{Y} \subset \mathcal{P}$ in $\mathcal{G}$ of size at most $O(\sqrt{\log n}) \cdot |\mathcal{X}| = O(\log^{3/2} n) \cdot |X|$.

Finally, we define the set Y. For every path $P_i \in \mathcal{P}$, let p_i be a leaf of T such that P_i is a subset of $T(p_i)$. Let $Y = \{p_i : P_i \in \mathcal{P}\}$. Note that $|Y| \leq |\mathcal{Y}| = O(\log^{3/2} n) \cdot |X|$. Since $\mathcal{Y}$ is a 3/4-balanced separator in $\mathcal{G}$, the set $Y' = \bigcup_{P_i\in\mathcal{Y}} V(P_i)$ is a 3/4-balanced separator in G, and therefore $\bigcup_{u\in Y} V(T(u)) \supset Y'$ is a 3/4-balanced separator in G. □

4 Computing Planarizing Sets of Paths

Lemma 4.1 (Computing a planarizing set of paths). *Let G be an n-vertex graph of genus $g > 0$. Let $r \in V(G)$, and let T be a spanning subtree of G*

with root r. Then, we can compute in polynomial time a set $X \subseteq V(G)$, with $|X| = O(g^2 \cdot \log^{5/2} n)$, such that the graph $G \setminus \bigcup_{v \in X} V(T(v))$ is planar.

Proof. We inductively construct a sequence $\{X_i\}_{i=0}^{k}$, for some $k = O(\log n)$, where for every $i \in \{0, \ldots, k\}$, we have $X_i \subseteq V(G)$. The resulting desired set will be $X = \bigcup_{i=0}^{k} X_i$.

For the basis of the induction, we set $X_0 = \emptyset$.

Let $i > 0$, and suppose that X_{i-1} has already been constructed. We show how to construct X_i. Let $\mathcal{C}_i$ be the set of connected components of $G \setminus \bigcup_{j=0}^{i-1} \bigcup_{u \in X_j} V(T(u))$. Let also $\mathcal{C}'_i$ be the set of non-planar components in $\mathcal{C}_i$. Note that G is the only component in $\mathcal{C}_1$. For every component $C \in \mathcal{C}_i$ we define a function $w_C : V(G) \to \mathbb{R}_{\geq 0}$ such that for every $v \in V(G)$,

$$w_C(v) = \begin{cases} 1, & \text{if } v \in V(C) \\ 0, & \text{if } v \notin V(C) \end{cases}$$

By Lemma 2.2 it follows that there exists $Y_C \subseteq V(G)$, with $|Y_C| \leq 4g + 3$, such that $\bigcup_{v \in Y_C} V(T(v))$ is a balanced vertex separator for (G, w_C). Therefore, by Lemma 3.2 we can compute in polynomial time a set $Z_C \subseteq V(G)$, with

$$|Z_C| \leq O(\log^{3/2} n) \cdot |Y_C| \leq O(\log^{3/2} n) \cdot (4g + 3),$$

and such that $\bigcup_{v \in Z_C} V(T(v))$ is an 3/4-balanced vertex separator for (G, w_C). We set

$$X_i = \bigcup_{C \in \mathcal{C}'_i} Z_C.$$

This concludes the inductive construction of the sequence $\{X_i\}_{i=0}^{k}$.

We next show that for some $k = O(\log n)$, the set $X = \bigcup_{i=0}^{k} X_i$ is as required. Consider some $i \geq 1$, and let $C \in \mathcal{C}'_i$ be a non-planar connected component of $G \setminus \bigcup_{j=0}^{i-1} \bigcup_{u \in X_j} V(T(u))$. Observe that there exists a connected component $C' \in \mathcal{C}_{i-1}$ such that $C \subseteq C'$. Since C is non-planar, it follows that C' is also non-planar, and thus $C' \in \mathcal{C}'_{i-1}$. By the construction, the set X_i contains the set $Z_{C'}$, where $\bigcup_{v \in Z_{C'}} V(T(v))$ is a 3/4-balanced vertex separator for $(G, w_{C'})$. It follows that $|V(C)| \leq 3|V(C')|/4$. Thus, the size of every non-planar connected component in $\mathcal{C}_i$ is at most $(3/4)^{i-1}|V(G)|$. This implies in particular that for $k = \lceil \log_{4/3} n \rceil - 1$, the set $\mathcal{C}_k$ does not contain any non-planar connected components, and therefore the graph $G \setminus \bigcup_{v \in X} V(T(v))$ is planar.

It remains to upper bound $|X|$. Since G has genus g, we have that for every $i \in \{0, \ldots, k\}$, the set $\mathcal{C}_i$ contains at most g non-planar connected components, i.e. $|\mathcal{C}'_i| \leq g$. Therefore,

$$|X| \leq \sum_{i=0}^{k} |X_i| \leq \sum_{i=0}^{k} \sum_{C \in \mathcal{C}'_i} |Z_C| \leq \sum_{i=0}^{k} \sum_{C \in \mathcal{C}'_i} O(\log^{3/2} n) \cdot (4g+3) \leq O(g^2 \cdot \log^{5/2} n),$$

as required. □

5 Putting Everything Together

The next lemma follows by the work of Sidiropoulos [Sid10].

Lemma 5.1 (Sidiropoulos [Sid10]). *Let G be a graph, and $r \in V(G)$. Let $P_1, \ldots, P_k$ be a collection of shortest paths in G, having r as a common end-point. Suppose that $G \setminus \bigcup_{i=1}^{k} V(P_i)$ is planar. Then, G admits a stochastic embedding into planar graphs, with distortion $O(\log k)$. Moreover, if the paths $P_1, \ldots, P_k$ are given, then we can sample from the stochastic embedding in polynomial time.*

Theorem 5.1 (Kawarabayashi, Mohar & Reed [iKMR08]). *There exists an algorithm which given a graph G of genus g, computes a drawing of G into a surface of genus g, in time $O\left(2^{O(g)} \cdot n\right)$.*

Theorem 5.2 (Main result). *There exists a polynomial time algorithm which given a graph G of genus $g > 0$, computes a stochastic embedding of G into planar graphs, with distortion $O(\log(g+1))$. In particular, the algorithm does not require a drawing of G as part of the input.*

Proof. We can use the algorithm of Kawarabayashi, Mohar & Reed from Theorem 5.1 to test whether $g \leq \log n$ in polynomial time. If $g \leq \log n$, then the algorithm from Theorem 5.1 returns a drawing of G into a surface of genus g. Since we have a drawing of G into a surface of genus g, we can use the algorithm of Sidiropoulos [Sid10], to compute the required embedding.

Otherwise, if $g > \log n$, we proceed as follows. Let r be an arbitrary vertex in G, and let T be a shortest-path tree in G, with root r. By Lemma 4.1 we can compute a set $X \subseteq V(G)$, with $|X| = O(g^2 \cdot \log^{5/2} n) = O(g^{9/2})$, such that the graph $G \setminus \bigcup_{v \in X} V(T(v))$ is planar. Since for every $v \in X$, the path $T(v)$ has r as an endpoint, it follows that we can use Lemma 5.1 with the collection of paths $\{T(v)\}_{v \in X}$, to compute in polynomial time a stochastic embedding into planar graphs, with distortion $O(\log |X|) = O(\log(g+1))$, as required. □

References

[Bak94] Baker, B.S.: Approximation algorithms for np-complete problems on planar graphs. J. ACM 41(1), 153–180 (1994)

[Bar96] Bartal, Y.: Probabilistic approximation of metric spaces and its algorithmic applications. In: 37th Annual Symposium on Foundations of Computer Science (Burlington, VT), pp. 184–193. IEEE Comput. Soc. Press, Los Alamitos (1996)

[BLS09] Borradaile, G., Lee, J.R., Sidiropoulos, A.: Randomly removing g handles at once. In: Proc. 25th Annual ACM Symposium on Computational Geometry (2009)

[CEN09] Chambers, E.W., Erickson, J., Nayyeri, A.: Homology flows, cohomology cuts. In: Proc. 41st Annual ACM Symposium on Theory of Computing (2009)

[CKK97] Chen, J., Kanchi, S.P., Kanevsky, A.: A note on approximating graph genus. Inf. Process. Lett. 61(6), 317–322 (1997)

[CS02] Charikar, M., Sahai, A.: Dimension reduction in the ℓ_1 norm. In: Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science, pp. 551–560. IEEE (2002)

[Epp03] Eppstein, D.: Dynamic generators of topologically embedded graphs. In: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 599–608. Society for Industrial and Applied Mathematics (2003)

[EW05] Erickson, J., Whittlesey, K.: Greedy optimal homotopy and homology generators. In: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1038–1046. Society for Industrial and Applied Mathematics (2005)

[FHL08] Feige, U., Hajiaghayi, M.T., Lee, J.R.: Improved approximation algorithms for minimum weight vertex separators. SIAM J. Comput. 38(2), 629–657 (2008)

[iKMR08] Kawarabayashi, K.I., Mohar, B., Reed, B.A.: A simpler linear time algorithm for embedding graphs into an arbitrary surface and the genus of graphs of bounded tree-width. In: FOCS, pp. 771–780 (2008)

[Ind01] Indyk, P.: Tutorial: Algorithmic applications of low-distortion geometric embeddings. In: Symposium on Foundations of Computer Science (2001)

[IS07] Indyk, P., Sidiropoulos, A.: Probabilistic embeddings of bounded genus graphs into planar graphs. In: Proc. 23rd Annual ACM Symposium on Computational Geometry (2007)

[LT79] Lipton, R.J., Tarjan, R.E.: A separator theorem for planar graphs. SIAM Journal on Applied Mathematics 36(2), 177–189 (1979)

[Mat99] Matoušek, J.: On embedding trees into uniformly convex Banach spaces. Isr. J. Math. 114, 221–237 (1999)

[Mat02] Matousek, J.: Lectures on Discrete Geometry. Springer (2002)

[MT01] Mohar, B., Thomassen, C.: Graphs on Surfaces. John Hopkins (2001)

[Sid10] Sidiropoulos, A.: Optimal stochastic planarization. In: 2010 IEEE 51st Annual Symposium on Foundations of Computer Science, pp. 163–170. IEEE (2010)

[Tho89] Thomassen, C.: The graph genus problem is np-complete. J. Algorithms 10(4), 568–576 (1989)

[Tho93] Thomassen, C.: Triangulating a surface with a prescribed graph. J. Comb. Theory, Ser. B 57(2), 196–206 (1993)

[Tho04] Thorup, M.: Compact oracles for reachability and approximate distances in planar digraphs. Journal of the ACM (JACM) 51(6), 993–1024 (2004)

The Projection Games Conjecture and the NP-Hardness of $\ln n$-Approximating Set-Cover

Dana Moshkovitz

MIT

Abstract. We suggest the research agenda of establishing new hardness of approximation results based on the "projection games conjecture", i.e., an instantiation of the Sliding Scale Conjecture of Bellare, Goldwasser, Lund and Russell to projection games.

We pursue this line of research by establishing a tight $\mathcal{NP}$-hardness result for the Set-Cover problem. Specifically, we show that under the projection games conjecture (in fact, under a quantitative version of the conjecture that is only slightly beyond the reach of current techniques), it is $\mathcal{NP}$-hard to approximate Set-Cover on instances of size N to within $(1-\alpha)\ln N$ for arbitrarily small $\alpha > 0$. Our reduction establishes a tight trade-off between the approximation accuracy α and the time required for the approximation $2^{N^{\Omega(\alpha)}}$, assuming Sat requires exponential time.

The reduction is obtained by modifying Feige's reduction. The latter only provides a lower bound of $2^{N^{\Omega(\alpha/\log\log N)}}$ on the time required for $(1-\alpha)\ln N$-approximating Set-Cover assuming Sat requires exponential time (note that $N^{1/\log\log N} = N^{o(1)}$). The modification uses a combinatorial construction of a bipartite graph in which any coloring of the first side that does not use a color for more than a small fraction of the vertices, makes most vertices on the other side have their neighbors all colored in different colors.

1 Introduction

1.1 Projection Games and The Projection Games Conjecture

Most of the $\mathcal{NP}$-hardness of approximation results known today (e.g., all of the results in Håstad's paper [Hås01]) are based on a PCP Theorem for *projection games* (also known as Label-Cover) [AS98, ALM+98, Raz98, MR10]. The input to a projection game consists of: (i) a bipartite graph $G = (A, B, E)$; (ii) finite alphabets Σ_A, Σ_B; (iii) constraints (also called *projections*) $\pi_e : \Sigma_A \to \Sigma_B$ for every edge $e \in E$. The goal is to find assignments to the vertices $\varphi_A : A \to \Sigma_A$, $\varphi_B : B \to \Sigma_B$ that *satisfy* as many of the edges as possible. We say that an edge $e = (a, b) \in E$ is satisfied, if the projection constraint holds, i.e., $\pi_e(\varphi_A(a)) = \varphi_B(b)$. We denote the size of a projection game by $n = |A| + |B| + |E|$. A PCP Theorem for projection games with soundness error ε and alphabet size k (where ε and k may depend on n) states the following:

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 276–287, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Given a projection game of size n with alphabets of size k, it is $\mathcal{NP}$-hard to distinguish between the case where all edges can be satisfied and the case where at most ε fraction of the edges can be satisfied.

We can refine this statement by saying that there is a reduction from (exact) SAT to projection games, and the reduction maps instances of SAT of size n to projection games of size $N = n^{1+o(1)}poly(1/\varepsilon)$. Such PCPs are referred to as "almost-linear size PCP" because of the exponent of n, although for small ε the blow-up may be super-linear.

The state of the art today for PCP Theorems for projection games is the following:

Theorem 1 ([MR10]). *There exists $c > 0$, such that for every $\varepsilon \geq 1/N^c$, SAT on input of size n can be reduced to a projection game of size $N = n^{1+o(1)}poly(1/\varepsilon)$ over alphabet of size $exp(1/\varepsilon)$ that has soundness error ε. The reduction is computed in polynomial time in N.*

Note that one cannot hope for ε that is lower than $1/N$ (polynomially small). The $exp(1/\varepsilon)$ in the statement is not tight. It can be shown that $|\Sigma| \geq 1/\varepsilon$, and we conjecture that an alphabet size of $poly(1/\varepsilon)$ could be achieved:

Conjecture 1 (Projection games conjecture, PGC). *There exists $c > 0$, such that for every $\varepsilon \geq 1/N^c$, SAT on input of size n can be efficiently reduced to a projection game of size $N = n^{1+o(1)}poly(1/\varepsilon)$ over alphabet of size $poly(1/\varepsilon)$ that has soundness error ε.*

In almost all applications, one wishes the alphabet size to be at most polynomial in n. This happens in Theorem 1 only when $\varepsilon \geq 1/(\log N)^b$ for some constant $b > 0$. The PGC, on the other hand, gives polynomial alphabet for any $\varepsilon \geq 1/N^c$.

The projection games conjecture is in fact the Sliding Scale Conjecture of Bellare, Goldwasser, Lund and Russell [BGLR93] instantiated for projection games. By "sliding scale" we refer to the idea that the error can be decreased as we increase the alphabet size. Bellare et al. conjectured that polynomially small error could be achieved simultaneously with polynomial alphabet, even for two queries. They did not formulate their conjecture for projection games – the importance of projection games was not fully recognized when they published their work in 1993.

1.2 Previous Work

Approximation algorithms for projection games were researched, and the conjecture is consistent with the state of the art algorithm, giving $1/\varepsilon = O(\sqrt[3]{Nk})$ [CHK09] (Note that the formulation in [CHK09] is slightly different than ours – they have a vertex per pair (vertex, assignment) in our formulation).

The existing hardness results for projection games include two results: the one mentioned in Theorem 1 and another result that is based on parallel repetition [Raz98]:

Theorem 2 ([Raz98]). *There exists $c > 0$, such that for every $\varepsilon \geq 1/N^{c/\log n}$,* Sat *on input of size n can be efficiently reduced to a projection game of size N over alphabet of size $O(1/\varepsilon)$ that has soundness error ε.*

Note that when the reduction is polynomial, i.e., $N = n^{O(1)}$, the soundness error is constant. Better soundness error ε can be obtained for larger N. For instance, for $N = n^{O(\log n)}$, one obtains $\varepsilon = 1/n$. Unfortunately, polynomially small error $1/N^c$ cannot be obtained from Theorem 2.

For PCPs with more than two queries, soundness error approaching polynomial, $\varepsilon = 2^{-(\log N)^{1-\epsilon}}$ for every $\epsilon > 0$, is known [DFK+11]. Alas, these PCPs are not projection games, and the number of queries depends on $1/\epsilon$.

The projection games conjecture has a similar flavor to the unique games conjecture (UGC) of Khot [Kho02]: both assert that low soundness error[1] for a special kind of 2-prover games can be obtained for sufficiently large alphabets. Unique games are the special case of projection games in which the projections π_e are 1-1. Unique games appear to be much easier than general projection games. In particular, while there are constructions of projection games with low soundness error for Sat, we do not know of any constructions of unique games with almost-perfect completeness[2] and bounded soundness error. The two conjectures, UGC and PGC, seem unrelated: neither would imply the other.

1.3 The Potential Influence of The PGC

We believe that the projection games conjecture provides a stable foundation on which many new hardness of approximation results can be based. In particular, for several central approximation problems, achieving tight hardness results seems to require projection games with low soundness error; a few examples follow.

In a work in progress with Gopal we research the approximability of Max-3Sat and Max-3Lin just above their approximation thresholds, which are 7/8 and 1/2, respectively. For context, Håstad discusses hardness beyond any *constant* larger than the thresholds [Hås01], and Moshkovitz-Raz improve this to $1/(\log\log n)^{O(1)}$ beyond the threshold [MR10], which is still quite large for reasonable n's. Researching the range of $1/n^{O(1)}$ beyond the threshold is possible assuming projection games with polynomially small error.

Other results we hope could be achieved (but would require further ideas) are:

- Tight lower bound for $n^{1-o(1)}$-approximation of Clique [Hås99, Kho01].
- Tight lower bound for $n^{\Omega(1)}$-approximation of the Shortest-Vector-Problem (SVP) in lattices [Kho05].

[1] The unique games conjecture only asks for arbitrarily small constant soundness error ε, while the PGC asks for polynomially small error.

[2] For unique games, if all the edges can be satisfied simultaneously, then one can find a satisfying assignment in polynomial time. Hence, we consider the case where *almost* all edges can be satisfied simultaneously ("almost perfect completeness").

In this paper, we show a tight lower bound on $(1-\alpha)\ln n$-approximation of Set-Cover assuming the projection games conjecture. (Of course, all the lower bounds are conditioned on a lower bound for Sat.)

There are several types of gains that can obtained from the PGC:

- **Better lower bounds.** For some problems (e.g., Set-Cover) the soundness error obtained from parallel repetition (Theorem 2) is sufficient, but the blow-up in the reduction translates into weak lower bounds. For Set-Cover, this lower bound is $2^{n^{\Omega(\alpha/\log\log n)}}$ which is much lower than the exponential lower bound one could a-priori hope for (note that $n^{1/\log\log n} = n^{o(1)}$).
- **Minimal assumptions.** The parallel-repetition based hardness result for Set-Cover can equivalently stated: Assuming $\mathcal{NP} \not\subseteq DTIME(n^{O(\log\log n)})$, polynomial-time $(1-\alpha)\ln n$-approximation for Set-Cover is ruled out [Fei98]. The PGC lets one see what results can potentially be obtained relying only on the minimal assumption $\mathcal{P} \neq \mathcal{NP}$.
- **Improved inapproximability factors.** For many problems (such as the other problems mentioned above: SVP, Clique, etc), one seems to need polynomially small soundness error to obtain the best inapproximability factor.

In all the aforementioned examples, the existing reductions have super-polynomial blow-up, not only in order to achieve low error for a projection game, but also to facilitate the reduction. For instance, Håstad's reductions use the long code on top of a projection game. For low error ε, the long code incurs a large blow-up $2^{(1/\varepsilon)^{O(1)}}$ [Hås01]. Basing hardness results on the PGC, would require reductions that do not resort to large blow-ups.

1.4 Set-Cover

We demonstrate the application of the PGC to the $\mathcal{NP}$-hardness of approximating Set-Cover. In Set-Cover, given a collection of sets over the same base set, such that the sets cover all of the base set, the goal is to find as few sets as possible that cover the entire base set:

Definition 3 (Set-Cover). *The input to* Set-Cover *consists of a base set* U, $|U| = n$ *and subsets* $S_1, \ldots S_m \subseteq U$, $\bigcup_{j=1}^{m} S_j = U$, $m \leq poly(n)$. *The goal is to find as few sets* $S_{i_1}, \ldots, S_{i_k}$ *as possible that cover* U, *i.e.,* $\bigcup_{j=1}^{k} S_{i_j} = U$.

Set-Cover is a classic $\mathcal{NP}$-hard optimization problem. It is equivalent to the Hitting-Set, Hypergraph-Vertex-Cover and Dominating-Set problems, and is a special case of many other problems, e.g., Group-Steiner-Tree and Group-Traveling-Salesman-Problem.

The greedy algorithm was shown to give a $(\ln n + 1)$-approximation for Set-Cover [Chv79]. Slavík analyzed the low order terms of the greedy algorithm, and showed that it in fact obtains an approximation to within $\ln n - \ln\ln n + O(1)$ [Sla96]. Set-Cover also has a linear programming based algorithm that gives approximation to within similar factors [Sri99].

Lund and Yannakakis proved that Set-Cover cannot be approximated in polynomial time to within any factor better than $(\log_2 n)/4$, assuming $NP \not\subseteq DTIME(n^{poly \log n})$ [LY93]. By adapting their construction, Feige changed the leading constant to the right constant, and showed that Set-Cover cannot be approximated in polynomial time to within $(1-\alpha)\ln n$ for any $\alpha > 0$, assuming $\mathcal{NP} \not\subseteq DTIME(n^{O(\lg \lg n)})$ [Fei98] (the improvement in the assumption is due to the proof of the parallel repetition theorem [Raz98] in the time between the two results). Under $\mathcal{P} \neq \mathcal{NP}$, the best hardness factor known is about $0.2 \ln n$ [AMS06], based on the PCP of [RS97, AS03].

The assumption $\mathcal{NP} \not\subseteq DTIME(n^{O(\lg \lg n)})$ in Feige's work comes from the use of the parallel repetition theorem. Parallel repetition is used by Feige not only to ensure very low error $1/(\log n)^{O(1)}$, but also for its unique structure. It was assumed by some that the blow-up incurred by parallel repetition was inherent to the problem. We show that this is not the case, assuming the PGC. Moreover, the blow-up in our reduction is essentially optimal.

Theorem 4. *For every $0 < \alpha < 1$, there is $c = c(\alpha)$, such that if the projection games conjecture holds with error $\varepsilon = \frac{c}{\lg^4 n}$, then (exact)* Sat *on inputs of size n can be reduced in polynomial time to approximating* Set-Cover *on inputs of size $N = n^{O(1/\alpha)}$ better than $(1-\alpha)\ln N$.*

The theorem proves that approximating Set-Cover on inputs of size N better than $(1-\alpha)\ln N$ is NP-hard, assuming the PGC. Interestingly, the blow-up of the reduction $N = n^{O(1/\alpha)}$ is optimal (up to the constant in the $O(\cdot)$), assuming that Sat requires exponential time $2^{\Omega(n)}$ and the PGC. This follows from a sub-exponential $2^{O(n^\alpha \log n)}$-time approximation algorithm for $(1-\alpha)\ln N$ approximating Set-Cover [CKW09].

Another interesting point about the theorem is that the quantitative version of the PGC that we need, namely, $\varepsilon = \frac{c}{\lg^4 n}$ for sufficiently small constant $c > 0$, is much weaker than the full conjecture, and it is just outside the reach of current techniques.

1.5 Preliminaries

For a set S and a natural number ℓ we denote by $\binom{S}{\ell}$ the family of all sets of ℓ elements from S.

We assume without loss of generality that the projection game in Conjecture 1 is bi-regular, i.e., all the A vertices have the same degree, which we call the A-degree, and all the B vertices have the same degree, which we call the B-degree. We note that any projection game can be converted to bi-regular using a technique developed in [MR10] ("right degree reduction – switching sides – right degree reduction"), and the cost in the soundness error and graph size does not change the parameters as stated in Conjecture 1.

2 Set-Cover Hardness

2.1 The New Component

Feige uses the structure obtained from parallel repetition to achieve a projection game in which the soundness guarantee is that very few B vertices have any two of their neighbors agree on a value for them:

Definition 5 (Total disagreement). *Assume a projection game*

$$(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi).$$

Let $\varphi_A : A \to \Sigma_A$ be an assignment to the A vertices. We say that the A vertices totally disagree *on a vertex $b \in B$ if there are no two neighbors $a_1, a_2 \in A$ of b, $e_1 = (a_1, b), e_2 = (a_2, b) \in E$, for which*

$$\pi_{e_1}(\varphi_A(a_1)) = \pi_{e_2}(\varphi_A(a_2)).$$

Definition 6 (Agreement soundness). *Assume a projection game*

$$(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$$

for deciding whether a boolean formula ϕ is satisfiable. We say that G has agreement soundness error *ε, if for unsatisfiable ϕ, for any assignment $\varphi_A : A \to \Sigma_A$, the A vertices are in total disagreement on at least $1 - \varepsilon$ fraction of the $b \in B$.*

Feige used parallel repetition together with a coding theoretic "trick" to achieve agreement soundness. We show a different way to achieve agreement soundness. Our construction centers around the following combinatorial construction:

Lemma 21 (Combinatorial construction). *For $0 < \varepsilon < 1$, for infinitely many n, D, there is an explicit construction of a regular graph $H = (U, V, E)$ with $|U| = n$, V-degree D, and $|V| \leq n^{O(1)}$ that satisfies the following. For every partition $U_1, \ldots, U_l$ of U into sets, such that $|U_i| \leq \varepsilon |U|$ for $i = 1, \ldots, l$, the fraction of vertices $v \in V$ with more than one neighbor in any single set U_i, is at most εD^2.*

Note that the combinatorial property could be achieved by a randomized construction, or by a construction that has a V vertex per every possible set of D neighbors in U. However, the first construction is randomized and the second – too wasteful with a size of $\approx |U|^D$. The lemma can therefore be thought of as a *derandomization* of the randomized/full constructions.

Proof. (of Lemma 21) Associate U with a space $\mathbb{F}^m$ where $\mathbb{F}$ is a finite field of size $|\mathbb{F}| = D$, and m is a natural number. Let V be the set of all lines in $\mathbb{F}^m$. Hence, $|V| = \binom{|U|}{2} / \binom{|\mathbb{F}|}{2}$. We connect a line $v \in V$ with a point $u \in U$ if u lies in v.

Let us show this construction satisfies the desired property. Fix a partition $U_1, \ldots, U_l$ of U into tiny sets, $|U_i| \leq \varepsilon |U|$ for $i = 1, \ldots, l$. For every $1 \leq i \leq l$,

the number of V lines that have at least two neighbors in U_i is at most $\binom{|U_i|}{2}$. Thus the total number of V vertices with more than one neighbor in a single U_i is at most

$$\begin{aligned}\sum_{i=1}^{l}\binom{|U_i|}{2} &\leq \sum_{i=1}^{l}\frac{|U_i|^2}{2}\\ &\leq \max\{\,|U_i|\;|\;1\leq i\leq l\}\cdot\sum_{i=1}^{l}\frac{|U_i|}{2}\\ &\leq \varepsilon\,|U|\cdot\frac{|U|}{2}\\ &\leq \varepsilon\,|\mathbb{F}|^2\,|V|\,.\end{aligned}$$

We show how to take a projection game with standard soundness and convert it to a projection game with total disagreement soundness, by combining it with the graph from Lemma 21.

Lemma 22. *Let $D\geq 2$, $\varepsilon>0$. From a projection game with soundness error ε^2D^2, we can construct a projection game with agreement soundness error $2\varepsilon D^2$ and B-degree D. The transformation preserves the alphabets of the game. The size is raised to a constant factor.*

Proof. Let $\mathcal{G}=(G=(A,B,E),\Sigma_A,\Sigma_B,\Phi)$ be the original projection game. Assume that the B-degree is $|U|$, and we use U to enumerate the neighbors of a B vertex, i.e., there is a function $E^{\leftarrow}:B\times U\to A$ that given a vertex $b\in B$ and $u\in U$, gives us the A vertex which is the u neighbor of b.

Let $H=(U,V,E_H)$ be the graph from Lemma 21. We create a new projection game $(G=(A,B\times V,E'),\Sigma_A,\Sigma_B,\Phi')$. The intended assignment to every vertex $a\in A$ is the same as its assignment in the original game. The intended assignment to a vertex $\langle b,v\rangle\in B\times V$ is the same as the assignment to b in the original game. We put an edge $e'=(a,\langle b,v\rangle)$ if $E^{\leftarrow}(b,u)=a$ and $(u,v)\in E_H$. We define $\pi_{e'}\equiv\pi_{(a,b)}$.

If there is an assignment to the original game that satisfies c fraction of its edges, then the corresponding assignment to the new game satisfies c fraction of its edges.

Suppose there is an assignment for the new game $\varphi_A:A\to\Sigma_A$ in which more than $2\varepsilon D^2$ fraction of the vertices in $B\times V$ do not have total disagreement.

Let us say that $b\in B$ is "good" if for more than εD^2 of the vertices in $\{b\}\times V$ the A vertices do not totally disagree. Note that the fraction of good $b\in B$ is at least εD^2.

Focus on a good $b\in B$. Consider the partition of U into $|\Sigma_B|$ sets, where the set corresponding to $\sigma\in\Sigma_B$ is:

$$U_\sigma=\{\,u\in U\;|\;a=E^{\leftarrow}(b,u)\wedge e=(a,b)\wedge\pi_e(\varphi_A(a))=\sigma\}\,.$$

By the property of H, there must be $\sigma\in\Sigma_A$ such that $|U_\sigma|>\varepsilon\,|U|$. We call σ the "champion" for b.

We define an assignment $\varphi_B : B \to \Sigma_B$ that assigns good b's their champions, and other b's arbitrary values. The fraction of edges that φ_A, φ_B satisfy in the original game is at least $\varepsilon^2 D^2$.

Next we consider a variant of projection games that is relevant for the reduction to Set-Cover. In this variant the prover is allowed to assign each vertex ℓ values, and an agreement is interpreted as agreement on *one* of the assignments in the list:

Definition 7 (List agreement). *Assume a projection game*

$$(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi).$$

Let $\ell \geq 1$. Let $\hat{\varphi}_A : A \to \binom{\Sigma_A}{\ell}$ be an assignment that assigns each A vertex l alphabet symbols. We say that the A vertices totally disagree *on a vertex $b \in B$ if there are no two neighbors $a_1, a_2 \in A$ of b, $e_1 = (a_1, b), e_2 = (a_2, b) \in E$, for which there exist $\sigma_1 \in \hat{\varphi}_A(a_1)$, $\sigma_2 \in \hat{\varphi}_A(a_2)$, such that*

$$\pi_{e_1}(\sigma_1) = \pi_{e_2}(\sigma_2).$$

Definition 8 (List agreement soundness). *Assume a projection game*

$$(G = (A, B, E), \Sigma_A, \Sigma_B, \Phi)$$

for deciding membership whether a boolean formula ϕ is satisfiable. We say that G has agreement soundness error (ℓ, ε), *if for unsatisfiable ϕ, for any assignment $\hat{\varphi}_A : A \to \binom{\Sigma_A}{\ell}$, the A vertices are in total disagreement on at least $1-\varepsilon$ fraction of the $b \in B$.*

If a projection game has low error ε, then even when the prover is allowed to assign each A vertex ℓ values, the game is still sound. This is argued in the next corollary:

Lemma 23 (Projection game with list agreement soundness). *Let $\ell \geq 1$, $0 < \varepsilon' < 1$. A projection game with agreement soundness error ε' has agreement soundness error $(\ell, \varepsilon' \ell^2)$.*

Proof. Assume on way of contradiction that the projection game has an assignment $\hat{\varphi}_A : A \to \binom{\Sigma_A}{\ell}$ such that on more than $\varepsilon' \ell^2$ fraction of the B vertices, the A vertices do not totally disagree. Define an assignment $\varphi_A : A \to \Sigma_A$ by assigning every vertex $a \in A$ a symbol picked uniformly at random from the ℓ symbols in $\hat{\varphi}_A(a)$. If a vertex $b \in B$ has two neighbors $a_1, a_2 \in A$ that agree on b under the list assignment $\hat{\varphi}_A$, then the probability that they agree on b under the assignment φ_A is at least $1/\ell^2$. Thus, under φ_A, the expected fraction of the B vertices that have at least two neighbors that agree on them, is more than ε'. In particular, there exists an assignment to the A vertices, such that more than ε' fraction of the B vertices have two neighbors that agree on them. This contradicts the agreement soundness of the game.

By applying Lemma 22 and then Lemma 23 on the game from Conjecture 1, we get:

Corollary 24. *Assuming Conjecture 1, for any $\ell \geq 1$, for infinitely many D, for any $\varepsilon \geq 1/n^c$, given a projection game with alphabet size $poly(1/\varepsilon)$ and B-degree D, it is $\mathcal{NP}$-hard to distinguish between the case where all edges can be satisfied, and the case where the agreement soundness error is $(\ell, 2D\ell^2\sqrt{\varepsilon})$.*

2.2 Following Feige's Reduction

In the remainder, we will show how to use Corollary 24 to obtain the desired hardness result for Set-Cover. The reduction is along the lines of Feige's original reduction.

For the reduction we rely on a combinatorial construction of a universe together with partitions of it. Each partition covers the universe, but any cover that takes at most one set out of each partition, is necessarily large:

Lemma 25 (Partition system, [NSS95]). *For natural numbers m, D, for $\alpha \leq 2/D$, there is an explicit construction of a universe U, $|U| \leq poly(D^{\log D}, \log m)$ and partitions $\mathcal{P}_1, \ldots, \mathcal{P}_m$ of U into D sets that satisfy the following: there is no cover of U with $\ell = D \ln |U| (1-\alpha)$ sets $S_{i_1}, \ldots, S_{i_\ell}$, $1 \leq i_1 < \cdots < i_\ell \leq m$, such that set S_{i_j} belongs to partition $\mathcal{P}_{i_j}$.*

To see why $\ell = D \ln |U| (1-\alpha)$ is to be expected (this later determines the hardness factor we get), think of the following randomized construction: each element in U corresponds to a vector in $[D]^m$, specifying for each of the m partitions, to which of its D sets it belongs. Consider a uniformly random choice of such a vector. Fix any $S_{i_1}, \ldots, S_{i_\ell}$. The probability that a random element is not covered by $S_{i_1}, \ldots, S_{i_\ell}$ is $(1-1/D)^\ell \approx e^{-\ell/D}$. When $\ell = D \ln |U| (1-\alpha)$, we have $e^{-\ell/D} \geq 1/|U|$, and we expect one of the $|U|$ elements in U not to be covered by $S_{i_1}, \ldots, S_{i_\ell}$. The construction in [NSS95] de-randomizes this randomized construction.

We now describe the reduction from a projection game $\mathcal{G}$ as in Corollary 24, to a Set-Cover instance $\mathcal{SC}_\mathcal{G}$.

Apply Lemma 25 for $m = |\Sigma_B|$ and D which is the B-degree of the projection game. Let U be the universe, and $\mathcal{P}_{\sigma_1}, \ldots, \mathcal{P}_{\sigma_m}$ be the partitions of U. We index the partitions by Σ_B symbols $\sigma_1, \ldots, \sigma_m$. The elements of the Set-Cover instance are $B \times U$.

For every vertex $a \in A$ and an assignment $\sigma \in \Sigma_A$ to a we have a set $S_{a,\sigma}$ in the Set-Cover instance. The intuition is that whether we take $S_{a,\sigma}$ to the cover would correspond to assigning σ to a. The set $S_{a,\sigma}$ is a union of subsets, one for every edge $e = (a, b)$ touching a. Suppose e is the i'th edge coming into b ($1 \leq i \leq D$), then the subset associated with e is the i'th subset of the partition $\mathcal{P}_{\varphi_e(\sigma)}$. Note that if we have an assignment to the A vertices, such that all of b's neighbors agree on one value for b, then the D subsets corresponding to those neighbors and their assignments form a partition that covers b's universe. On the other hand, if one uses only sets that correspond to totally disagreeing

assignments to the neighbors, then by the definition of the partitions, covering U requires $\approx \ln|U|$ times more sets.

Claim 26 *The following hold:*

- *Completeness: If all the edges in $\mathcal{G}$ can be satisfied, then $\mathcal{SC}_{\mathcal{G}}$ has a set cover of size $|A|$.*
- *Soundness: Let $\ell \doteq D\ln|U|\,(1-\alpha)$ be as in Lemma 25. If $\mathcal{G}$ has agreement soundness (ℓ,α), then every set cover of $\mathcal{SC}_{\mathcal{G}}$ is of size more than $|A|\ln|U|\,(1-2\alpha)$.*

Proof. Completeness follows from taking the set cover corresponding to each of the A vertices and its satisfying assignment.

Let us prove soundness. Assume on way of contradiction that there is a set cover C of $\mathcal{SC}_{\mathcal{G}}$ of size at most $|A|\ln|U|\,(1-2\alpha)$. For every $a \in A$ let s_a be the number of sets in C of the form $S_{a,\cdot}$. Hence, $\sum_{a\in A} s_a = |C|$. For every $b \in B$ let s_b be the number of sets in C that participate in covering $\{b\}\times U$. Then, denoting the A-degree of G by D_A,

$$\sum_{b\in B} s_b = \sum_{a\in A} s_a D_A \le D_A\,|A|\ln|U|\,(1-2\alpha) = D\,|B|\ln|U|\,(1-2\alpha).$$

In other words, on average over the $b \in B$, the universe $\{b\}\times U$ is covered by at most $D\ln|U|\,(1-2\alpha)$ sets. Therefore, by Markov's inequality, the fraction of $b \in B$ whose universe $\{b\}\times U$ is covered by at most $D\ln|U|\,(1-\alpha) = \ell$ sets is at least α. By Lemma 25 and our construction, for such $b \in B$, there are two edges $e_1 = (a_1, b), e_2 = (a_2, b) \in E$ with $S_{a_1,\sigma_1}, S_{a_2,\sigma_2} \in C$ where $\pi_{e_1}(\sigma_1) = \pi_{e_2}(\sigma_2)$.

We define an assignment $\hat{\varphi}_A : A \to \binom{\Sigma_A}{\ell}$ to the A vertices as follows. For every $a \in A$ pick ℓ different symbols $\sigma \in \Sigma_A$ from those with $S_{a,\sigma} \in C$ (add arbitrary symbols if there are not enough). As we showed, for at least α fraction of the $b \in B$, the A vertices will not totally disagree.

Fix a constant $0 < \alpha < 1$. The inapproximability ratio we get for Set-Cover from Claim 26 is $(1-2\alpha)\ln|U|$, assuming agreement soundness (ℓ,α). The latter is obtained from Corollary 24 for $\varepsilon = c/\log^4 n$ for a certain constant $c = c(\alpha)$. Let $N = |U|\,|B|$ be the number of elements in $\mathcal{SC}_{\mathcal{G}}$. We take $|U| = \Theta(|B|^{1/\alpha})$ (we might need to duplicate elements for that), so $\ln N = (1+\alpha)\ln|U|$, and the inapproximability ratio is at least $(1-3\alpha)\ln N$. Note that the reduction is polynomial in n. This proves Theorem 4.

3 Open Problems

The main open problem is to prove the projection games conjecture. We believe that many more hardness of approximation results could be proved based on the PGC. Two concrete open problems are to prove results for Clique and SVP. It will be interesting to show equivalence between certain strong hardness results and the PGC. Another very interesting open problem is to find better approximation algorithms for projection games.

Acknowledgments. The motivation to prove the Set-Cover result came from discussions with Ran Raz. The author would also like to thank Scott Aaronson, Zach Friggstad, Ryan O'Donnell, Muli Safra and the anonymous reviewers for useful comments.

References

[ALM+98] 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)

[AMS06] Alon, N., Moshkovitz, D., Safra, S.: Algorithmic construction of sets for k-restrictions. ACM Trans. Algorithms 2, 153–177 (2006)

[AS98] Arora, S., Safra, S.: Probabilistic checking of proofs: a new characterization of NP. Journal of the ACM 45(1), 70–122 (1998)

[AS03] Arora, S., Sudan, M.: Improved low-degree testing and its applications. Combinatorica 23(3), 365–426 (2003)

[BGLR93] Bellare, M., Goldwasser, S., Lund, C., Russell, A.: Efficient probabilistically checkable proofs and applications to approximations. In: Proc. 25th ACM Symp. on Theory of Computing, pp. 294–304 (1993)

[CHK09] Charikar, M., Hajiaghayi, M., Karloff, H.: Improved Approximation Algorithms for Label Cover Problems. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 23–34. Springer, Heidelberg (2009)

[Chv79] Chvatal, V.: A greedy heuristic for the set-covering problem. Mathematics of Operations Research 4(3), 233–235 (1979)

[CKW09] Cygan, M., Kowalik, L., Wykurz, M.: Exponential-time approximation of weighted set cover. Inf. Process. Lett. 109(16), 957–961 (2009)

[DFK+11] Dinur, I., Fischer, E., Kindler, G., Raz, R., Safra, S.: PCP characterizations of NP: Toward a polynomially-small error-probability. Computational Complexity 20(3), 413–504 (2011)

[Fei98] Feige, U.: A threshold of $\ln n$ for approximating set cover. Journal of the ACM 45(4), 634–652 (1998)

[Hås99] Håstad, J.: Clique is hard to approximate within $n^{1-\epsilon}$. Acta Mathematica 182, 105–142 (1999)

[Hås01] Håstad, J.: Some optimal inapproximability results. Journal of the ACM 48(4), 798–859 (2001)

[Kho01] Khot, S.: Improved inapproximability results for maxclique, chromatic number and approximate graph coloring. In: Proc. 42nd IEEE Symp. on Foundations of Computer Science, pp. 600–609 (2001)

[Kho02] Khot, S.: On the power of unique 2-prover 1-round games. In: Proc. 34th ACM Symp. on Theory of Computing, pp. 767–775 (2002)

[Kho05] Khot, S.: Hardness of approximating the shortest vector problem in lattices. Journal of the ACM 52(5), 789–808 (2005)

[LY93] Lund, C., Yannakakis, M.: On the hardness of approximating minimization problems. In: Proc. 25th ACM Symp. on Theory of Computing (1993)

[MR10] Moshkovitz, D., Raz, R.: Two query PCP with sub-constant error. Journal of the ACM 57(5) (2010)

[NSS95] Naor, M., Schulman, L.J., Srinivasan, A.: Splitters and near-optimal derandomization. In: Proc. 36th IEEE Symp. on Foundations of Computer Science, pp. 182–191 (1995)

[Raz98] Raz, R.: A parallel repetition theorem. SIAM Journal on Computing 27, 763–803 (1998)

[RS97] Raz, R., Safra, S.: A sub-constant error-probability low-degree test and a sub-constant error-probability PCP characterization of NP. In: Proc. 29th ACM Symp. on Theory of Computing, pp. 475–484 (1997)

[Sla96] Slavík, P.: A tight analysis of the greedy algorithm for set cover. In: Proc. 28th ACM Symp. on Theory of Computing, pp. 435–441 (1996)

[Sri99] Srinivasan, A.: Improved approximations guarantees for packing and covering integer programs. SIAM Journal on Computing 29(2), 648–670 (1999)

New and Improved Bounds for the Minimum Set Cover Problem

Rishi Saket[1] and Maxim Sviridenko[2]

[1] IBM T.J. Watson Research Center, NY, USA
[2] Department of Computer Science, University of Warwick, UK

Abstract. We study the relationship between the approximation factor for the Set-Cover problem and the parameters Δ : the maximum cardinality of any subset, and k : the maximum number of subsets containing any element of the ground set. We show an LP rounding based approximation of $(k-1)(1-e^{-\frac{\ln \Delta}{k-1}})+1$, which is substantially better than the classical algorithms in the range $k \approx \ln \Delta$, and also improves on related previous works [19,22]. For the interesting case when $k = \theta(\log \Delta)$ we also exhibit an integrality gap which essentially matches our approximation algorithm. We also prove a hardness of approximation factor of $\Omega\left(\frac{\log \Delta}{(\log\log \Delta)^2}\right)$ when $k = \theta(\log \Delta)$. This is the first study of the hardness factor specifically for this range of k and Δ, and improves on the only other such result implicitly proved in [18].

1 Introduction

We consider the classical minimum set cover problem. We are given the ground set $\{1,\ldots,n\} = [n]$ and m subsets $S_j \subseteq [n]$ for $j = 1,\ldots,m$. Each set S_j has an associated non-negative weight w_j. The goal is to choose a collection of sets indexed by $\mathcal{C} \subseteq \{1,\ldots,m\} = [m]$ such that $[n] = \cup_{j\in\mathcal{C}} S_j$ and minimize $\sum_{j\in\mathcal{C}} w_j$.

There are two additional parameters associated with the problem. Let $\Delta = \max_{j\in[m]} |S_j|$ be the maximal cardinality of a set in the instance. For each element $i \in [n]$, let $k_i = |\{S_j : i \in S_j, j \in [m]\}|$ be the number of sets in the instance containing the element $i \in [n]$ and let $k = \max_{i\in[n]} k_i$.

There are two types of classical approximation algorithms for the minimum set cover problem. The natural greedy algorithm has performance guarantee $\ln \Delta + 1$ [20,12,5]. Another well-known type of algorithms has performance guarantee k [4,10]. Both performance guarantees are asymptotically the best possible under natural complexity assumptions [7,6,17] specifically for the regime where Δ is not bounded by a constant, although for constant Δ a performance guarantee strictly better than k can be obtained [9]. Nevertheless, assuming that Δ is not bounded, if one defines the performance ratio $\rho(k)$ as a function of parameter k the classical approximation algorithms provide us with performance guarantee $\rho(k) = \min\{k, \ln \Delta + 1\}$ (see Figure 1). The function $\rho(k)$ is not smooth at the point $k = \ln \Delta + 1$, which indicates that performance guarantee of classical algorithms is not best possible, at least in regime when $k \approx \ln \Delta + 1$.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 288–300, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Our Results. In this paper we study the relationship between the approximation factor for Set-Cover in terms of k and Δ. We prove the following results.

Approximation Algorithm. In this paper we design a simple LP rounding based approximation algorithm with performance guarantee $(k-1)(1-e^{-\frac{\ln \Delta}{k-1}})+1$ which asymptotically matches the performance guarantee of known (and best possible) approximation algorithms when $k \ll \ln \Delta$ or $k \gg \ln \Delta$ in the regime where Δ is unbounded. In particular, when $k = \ln \Delta + 1$, our algorithm has performance guarantee $(1 - e^{-1}) \ln \Delta + 1$. For a comparison of the performance of our algorithm with $\rho(k)$, refer to Figure 2. Our approximation algorithm and its analysis are presented in Section 2.

Previous results in this direction are due to Krivelevich [19] and Okun [22]. Using our notations Krivelevich [19] designed an approximation algorithm with performance guarantee $\max\{k-1, (k-1)(1-e^{-\frac{\ln \Delta}{k-1}})+1\}$ for the case when all subsets have cardinality Δ and all elements of the ground set belong to exactly k sets. Okun [22] designed an approximation algorithm that works in the regime when $(1 - e^{-1})k \leq \ln \Delta$ with performance guarantee smaller than k but strictly worse than ours.

Integrality Gap. For the interesting regime where $k = \theta(\log \Delta)$ we show an LP integrality gap of $k(1 - e^{-\frac{\ln \Delta}{k}} - \delta)$ for any constant $\delta > 0$, essentially matching our LP rounding upper bound. Our construction is probabilistic and is given in Section 3.

Hardness of Approximation. In this work we obtain a lower bound of $\Omega\left(\frac{\log \Delta}{(\log \log \Delta)^2}\right)$ when $k = \theta(\log \Delta)$, where Δ is a polynomial in n. In previous work, Feige [7] had shown that in the regime where $k = \Omega(\Delta^\gamma)$ for some constant $\gamma > 0$, it is hard to approximate Set-Cover to within a factor of $(1 - \varepsilon) \ln \Delta$. As mentioned before, this essentially matches the $\ln \Delta + 1$ greedy algorithm. A slightly weaker lower bound of $\Omega(\log \Delta)$ was obtained by Lund and Yannakakis [21] for $k = \Omega((\log \Delta)^c)$, where $c > 1$ is a large constant, and for $k = 2^{\log^{1-\varepsilon} \Delta}$ by Raz and Safra [23] and by Alon, Moshkovitz and Safra [1]. On the other hand, for small values of k the known hardness factors are linear in k. For constant k, assuming the Unique Games Conjecture [14] it is NP-hard to approximate within a factor of $k - \varepsilon$ [17,3]. In [6] it was shown that for superconstant $k = O((\log \log \Delta)^{1/c})$ the hardness factor is $k - 1 - \varepsilon$, and for $k = O((\log \Delta)^{1/c})$ it is $k/2 - \varepsilon$. In all these hardness reductions (except for that of [17,3]) Δ is a polynomial in the size of the ground set n. It should be noted that these hardness results did not explicitly state the dependence between k, Δ and n, and these relations can be inferred from the respective hardness reductions.

However, the interesting case when $k = \theta(\log \Delta)$ remained unexplored till the work of Khot and Saket [18] who studied the problem of minimizing the size of DNF expression of a boolean function given its truth table. In their work [18], they implicitly obtain a hardness factor (for $k = \theta(\log \Delta)$, Δ polynomial in n) of $\Omega(\log^{1-\varepsilon} \Delta)$ although [18] do not explicitly mention this in their work.

Our stronger lower bound of $\Omega\left(\frac{\log \Delta}{(\log \log \Delta)^2}\right)$ is obtained by revisiting the Probabilistically Checkable Proof (PCP) construction of [18] using different parameters while avoiding some of the complications of their reduction, and is presented in Section 4. This still leaves open the possibility that when $k = \ln \Delta + 1$, our approximation of $(1 - e^{-1}) \ln \Delta + 1$ may not be optimal. Conversely, it may be possible to improve the hardness factor to match the algorithmic bound. We include, in Section 4.4, a brief discussion on some of the limitations of current PCP techniques to improving the hardness factor. The hardness result of this paper along with previous ones for various regimes of k are summarized in Figure 3.

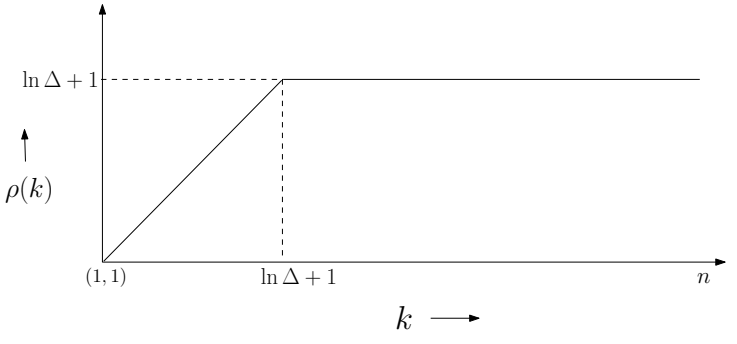

Fig. 1. Approximation Factor by Classical Algorithms

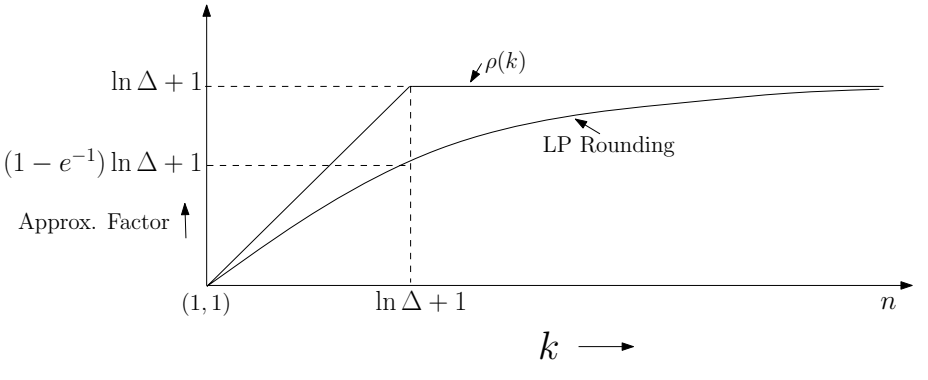

Fig. 2. Comparison of $\rho(k)$ with the LP Rounding Approximation for growing parameter Δ

Range of k	Hardness Factor	Complexity Assumption	Reference
k: arbitrarily large const.	$k - \varepsilon$	Unique Games Conj. [14]	[17,3]
$k \leq O((\log\log \Delta)^{1/c})$	$k - 1 - \varepsilon$	NP$\not\subseteq$ DTIME($n^{O(\log\log n)}$)	[6]
$k \leq O((\log \Delta)^{1/c})$	$k/2 - \varepsilon$	NP$\not\subseteq$ DTIME($n^{O(\log\log n)}$)	[6]
$k = \theta(\log \Delta)$	$\Omega(\log^{1-\varepsilon} \Delta)$	NP$\not\subseteq$ DTIME($n^{\mathrm{poly}(\log n)}$)	Implicit in [18]
$k = \theta(\log \Delta)$	$\Omega\left(\frac{\log \Delta}{(\log\log \Delta)^2}\right)$	NP$\not\subseteq$ DTIME($n^{\mathrm{poly}(\log n)}$)	This work.
$k = \Omega((\log \Delta)^c)$	$\Omega(\log \Delta)$	NP$\not\subseteq$ DTIME($n^{O(\log\log n)}$)	[21,15]
$k = \Omega(2^{\log^{1-\varepsilon} \Delta})$	$\Omega(\log \Delta)$	P $\neq$ NP	[23,1]
$k = \Omega(\Delta^\gamma)$	$(1-\varepsilon)\ln \Delta$	NP$\not\subseteq$ DTIME($n^{O(\log\log n)}$)	[7]

Fig. 3. Summary of known NP-hardness factors for Set-Cover with different ranges of k

2 Approximation Algorithm

Consider the following linear programming relaxation of the minimum set cover problem:

$$\min \sum_{j\in[m]} w_j x_j, \tag{1}$$

$$\sum_{j:i\in S_j} x_j \geq 1, \ \forall i \in [n], \tag{2}$$

$$x_j \geq 0, \ \forall j \in [m]. \tag{3}$$

Our approximation algorithm solves linear programming relaxation on the first step. Let LP^* be the optimal value of the linear programming relaxation and $x_j^*, j \in [m]$ be the optimal fractional solution found by the LP solver. We define $p_j = \min\{1, \alpha k \cdot x_j^*\}$ where $\alpha = 1 - e^{-\frac{\ln \Delta}{k-1}}$. Our approximation algorithm defines a partial cover by choosing to add the set S_j to the cover with probability p_j and not choosing it with probability $1 - p_j$ independently at random. Let R_1 be the indices of sets chosen by our random procedure. Let I^r be the set of the elements of the ground set that do not belong to any of the sets chosen by the random procedure, i.e. the set of uncovered elements. Each element in I^r chooses the cheapest set in our instance that covers it. Let R_2 be the set of indices of such sets covering I^r. Our algorithm outputs $R_1 \cup R_2$ as the final solution.

Theorem 1. *The expected value of the approximate solution output by our algorithm is at most* $((k-1)(1-e^{-\frac{\ln \Delta}{k-1}})+1)LP^*$.

Proof. By linearity of expectation, the expected value of the sets indexed by R_1 is $\sum_{j\in[m]} w_j p_j \leq k(1-e^{-\frac{\ln \Delta}{k-1}})LP^*$.

Assume that each element $i \in [n]$ of the ground set chooses the cheapest set that covers that element. Let j_i be the index of such a set and $W = \sum_{i\in[n]} w_{j_i}$ be

the upper bound on the weight of chosen sets. Then by utilizing the constraints (2) we obtain

$$W = \sum_{i\in[n]} w_{j_i} \leq \sum_{i\in[n]} w_{j_i} \sum_{j:i\in S_j} x^*_j \leq \sum_{i\in[n]} \sum_{j:i\in S_j} w_j x^*_j \leq \Delta \cdot LP^*.$$

Now, we estimate $Pr[i \in I^r]$ above. If $p_j = 1$ for at least one set such that $i \in S_j$ then $Pr[i \in I^r] = 0$. Otherwise, $p_j = \alpha k \cdot x^*_j$ for all sets S_j such that $i \in S_j$ and

$$Pr[i \in I^r] = \prod_{j|i\in S_j} (1-p_j) \leq \left(1 - \frac{\sum_{j|i\in S_j} p_j}{k_i}\right)^{k_i} \leq \left(1 - \frac{\sum_{j|i\in S_j} p_j}{k}\right)^{k}$$
$$= \left(1 - \frac{\sum_{j|i\in S_j} \alpha k \cdot x^*_j}{k}\right)^{k} \leq (1-\alpha)^k = \frac{1}{\Delta^{k/(k-1)}}.$$

Therefore, by linearity of expectation, the expected weight of the sets in R_2 can be estimated above by $W/\Delta^{k/(k-1)} \leq LP^*/\Delta^{1/(k-1)}$. Overall, the expected cost of the approximate solution is upper bounded above by

$$\left(k(1-e^{-\frac{\ln\Delta}{k-1}}) + \frac{1}{\Delta^{1/(k-1)}}\right) LP^* = \left((k-1)(1-e^{-\frac{\ln\Delta}{k-1}}) + 1\right) LP^*$$

3 Integrality Gap

The integrality gap of a linear programming relaxation for the specific instance of a minimization problem is the ratio between the minimum value integral solution of the relaxation (in the numerator) and the minimum value of the fractional solution (in the denominator).

Consider the following instance of the minimum set cover problem. We are given a ground set of n elements and $m = n^\varepsilon$ sets. We fix an arbitrary constant $c > 0$ and consider the regime when $k = c \cdot \ln n$. Each element $i \in [n]$ independently at random chooses k sets out of possible m sets, i.e. this element chooses one combination of k sets out of possible $\binom{m}{k}$ variants uniformly at random. Each set S_j for $j \in [m]$ consists of elements that chose that set. Let $\mathcal{I}_\varepsilon$ be the resulting random instance of the minimum set cover problem. Note that the parameter $\Delta \leq n$. We prove the following theorem showing that for all values of $c > 0$ the instance $\mathcal{I}_\varepsilon$ is, with high probability, the desired integrality gap example.

Theorem 2. *For any constants $c > 0$ and $\delta > 0$, there exists a constant $\varepsilon > 0$ such that the integrality gap of the linear programming relaxation (1)-(3) for the instance $\mathcal{I}_\varepsilon$ is at least $k(1 - e^{-1/c} - \delta)$ with high probability for large enough n.*

Proof. First, we note the fractional solution $x'_j = 1/k$ for all $j \in [m]$ is feasible. Indeed, each element is covered by exactly k sets in the instance $\mathcal{I}_\varepsilon$. Therefore, $\sum_{j:i\in S_j} x'_j = 1$ for each element $i \in [n]$. We obtain $LP^* \leq m/k$.

We will assume that the constants $c, \delta > 0$ and m are such that the number $(e^{-1/c} + \delta)m$ is an integer. We now fix an arbitrary collection of sets indexed by $\mathcal{C} \subseteq [m]$ such that $|\mathcal{C}| = (1 - e^{-1/c} - \delta)m$. We will estimate the probability that this integral solution is infeasible, i.e. there exist an element $i \in [n]$ which is left uncovered by the sets in this collection in the instance $\mathcal{I}_\varepsilon$.

The probability that a fixed element $i \in [n]$ is not covered by the sets indexed by $\mathcal{C}$ is exactly

$$\frac{\binom{(e^{-1/c}+\delta)m}{k}}{\binom{m}{k}} = \prod_{i=0}^{k-1} \frac{(e^{-1/c}+\delta)m - i}{m-i} \geq (e^{-1/c} + \delta/2)^k = \frac{(1 + e^{1/c}\delta/2)^{c \ln n}}{n} = n^{-(1-F_{c,\delta})},$$

where the inequality holds for large enough m since $k << m$ and $F_{c,\delta} = c \ln(1 + \delta e^{1/c}/2)$ is a constant depending on c and δ. We assume that δ is small enough that $F_{c,\delta} \in (0, 1)$.

Since each element chooses its sets independently, the probability that all n elements are covered by the sets indexed by $\mathcal{C}$ is at most

$$\left(1 - n^{-(1-F_{c,\delta})}\right)^n \leq e^{-n^{F_{c,\delta}}}.$$

The total number of choices for the index set $\mathcal{C}$ is at most $2^m = 2^{n^\varepsilon}$. Therefore, by the union bound, the probability that there exists a feasible index set $\mathcal{C}$ is at most

$$e^{-n^{F_{c,\delta}}} 2^{n^\varepsilon} \leq e^{n^\varepsilon - n^{F_{c,\delta}}}.$$

If we choose $\varepsilon = F_{c,\delta}/2$ then probability that there exists a feasible solution becomes negligibly small for large values of n. Therefore, with probability at least $1 - e^{n^\varepsilon - n^{F_{c,\delta}}}$ one needs to choose at least $(1 - e^{-1/c} - \delta)m$ sets into any feasible integral solution. This implies the claimed bound on the integrality gap.

4 Hardness of Approximation

In this section we shall derive an inapproximability result for the minimum set cover problem when $k = \theta(\log \Delta)$. Our reduction utilizes a PCP verifier constructed by Khot and Saket [18] who used it to prove a nearly optimal hardness result for minimizing the size of DNF expressions for a boolean function given its truth table, which is itself a special case of minimum set cover. We slightly modify the parameters of the verifier constructed in [18] to construct an instance of maximum constraint satisfaction problem (CSP) with some specific properties. This is then combined with a reduction – similar to that of Holmerin [11] for vertex cover – to obtain an instance of Set-Cover. In Section 4.1 we define the constraint satisfaction problem and state a hardness result for it, a proof of which is given in Section 4.3. The hardness reduction to Set-Cover is given in Section 4.2.

In the rest of this section, for convenience, we shall use notations (such as k, n) in contexts different to the previous sections.

4.1 A Hardness Result for Constraint Satisfaction

In this section we shall describe a result on the hardness of a variant of maximum constraints satisfaction problem (as defined below), which shall be useful in our reduction for the Set-Cover problem.

Definition 1. *An instance of Max-CSP-Reg(t, k) with N constraints, with parameters t, k as functions of N consists of a set of variables V, a label set $[k]$ and a set of t-variable constraints E, with $|E| = N$. The constraints are non-trivial, i.e. there is at least one satisfying labeling for every constraint. Additionally, each variable occurs in the same number of constraints. The goal is to assign labels to each variable in V to satisfy as many constraints in E as possible.*

The following hardness result for Max-CSP-Reg follows from the results in [16] and [18], and a formal proof is presented in Section 4.3.

Theorem 3. *Given an instance $\mathcal{A}$ of Max-CSP-Reg(t, k) with variable set V and set of constraints E, where $|E| = N$, $tk = \omega(\log N)$ and $t = \theta((\log k)^2)$, there is no polynomial time algorithm to distinguish between the following two cases:*

YES CASE: There is a set $V' \subset V$ of variables of size at most $|V|/(k^3)$ and a labeling $\sigma^* : V \setminus V' \mapsto [k]$ such that,

1. *(Strong Completeness) σ^* satisfies all constraints in E induced by $V \setminus V'$.*
2. *(Extendability) For any constraint $e \in E$ (possibly containing variables from V'), there is a labeling σ'_e to variables in $e \cap V'$ such that σ^* extended by σ'_e satisfies constraint e.*

NO CASE: Any labeling σ to the variables of V satisfies at most $k^{-t+O(\sqrt{t})}$ (soundness) fraction of the constraints,

unless NP $\subseteq$ DTIME$(n^{\text{poly}(\log n)})$.

In the next subsection we shall give a reduction from Max-CSP-Reg to an instance of Set Cover to prove our hardness result.

4.2 Reduction to Set-Cover

Now we give a reduction from the instance $\mathcal{A}$ of Max-CSP-Reg(t, k) given in Theorem 3 to an instance $\mathcal{I}$ of Set-Cover. As before we have V as the variable set of $\mathcal{A}$ and E as the set of t-variable constraints, where $E = |N|$, $kt = \omega(\log N)$ and $t = \theta((\log k)^2)$. Before describing the instance $\mathcal{I}$ of Set-Cover, we need to construct the following objects.

For every variable v, define a set $L(v) := \{(v, i) \mid i \in [k]\}$, which is just the set of all labels for that variable. Let $e \in E$ be any constraint over variables $v_1, \ldots, v_t$. Define $\tilde{L}(e) := \cup_{i=1}^{t} L(v_i)$. Clearly, $|\tilde{L}(e)| = tk$ for all $e \in E$.

Let $T(e)$ be set of all labelings τ to $v_1, \ldots, v_t$ that satisfy e. Since the constraints are non-trivial, $T(e) \neq \emptyset$ for all $e \in E$. We say that a subset $S \subseteq \tilde{L}(e)$ is "good" if for all $\tau \in T(e)$, there is an $i \in \{1, \ldots, t\}$ such that $(v_i, \tau(v_i)) \in S$.

In other words, every assignment to the variables $v_1, \ldots, v_t$, that satisfies e, has a variable-label pair from S. As an illustration, suppose e is a constraint over vertices $v_1, \ldots, v_t$ such that assigning the label $1 \in [k]$ to each of $v_1, \ldots v_t$ satisfies e. Then any good subset $S \subseteq \tilde{L}(e)$ must contain at least one pair $(v_i, 1)$ for some $i \in \{1, \ldots, t\}$, and this should similarly hold for any satisfying assignment to $v_1, \ldots, v_t$ which satisfies e.Let $G(e)$ to be the set of all such "good" subsets S for the constraint $e \in E$. With these definitions we now describe the ground set G and the set of subsets $\mathcal{C}$ for our instance $\mathcal{I}$ of Set-Cover.

Ground Set G. The ground set is defined as $G := \cup_e G(e)$, where the union is over all constraints $e \in E$.

Set of Subsets $\mathcal{C}$. Every possible variable-label pair (v, i), there is a subset $C(v, i)$ which contains all elements from G (i.e. "good" subsets of $\tilde{L}(e)$ for all constraints e) that contain (v, i). This finishes the construction of our Set-Cover instance.

Note that every element of the ground set G can be covered by at most tk subsets from $\mathcal{C}$ and that for every constraint e, $|G(e)| \leq 2^{tk}$ and therefore $|G| \leq 2^{kt}N$. Also, since $kt = \omega(\log N)$, we obtain that $\log |G| = O(kt)$. We now analyze the YES and NO cases of $\mathcal{A}$.

YES Case. In the YES case there is a subset V' of the variables V and a labeling τ^* to $V \setminus V'$ as given in Theorem 3. We construct a cover $\mathcal{H}^*$ for the instance $\mathcal{I}$ as follows. For all variables v in $V \setminus V'$ we choose the subset $C(v, \tau^*(v))$. Additionally, for all variables v' in V' we choose *all* subsets $C(v', i)$ for all $i \in [k]$.

Let us first confirm that $\mathcal{H}^*$ indeed covers all elements of the ground set G. Consider a constraint e over variables $v_1, \ldots, v_t$. Let us first consider the case when e does not contain any variable from V'. By construction of $G(e)$, we have that for every $S \in G(e)$, there is an $i \in \{1, \ldots, t\}$ such that $(v_i, \tau^*(v_i)) \in S$. Thus $G(e)$ is covered by $\mathcal{H}^*$. Now consider the case that e contains some variables from V'. In this case, by the Extendability property of Theorem 3, τ^* can be extended by choosing labels to variables in $e \cap V'$ so that the constraint e is satisfied. Since $\mathcal{H}^*$ contains all subsets $C(v', i), i \in [k]$, for all $v' \in e \cap V'$, this implies that it covers all elements in $G(e)$. Thus, $\mathcal{H}^*$ is a valid set cover.

Now, $\mathcal{H}^*$ chooses one subset for each variable in $V \setminus V'$, and k subsets for all variables in V'. Therefore we have, $|\mathcal{H}^*| = |V \setminus V'| + k|V'| \leq |V|(1 + k^{-2})$, using the bound in Theorem 3 that $|V'|/|V| = O(k^{-3})$.

NO Case. In the NO case let $\mathcal{H} \subseteq \mathcal{C}$ be any cover. We shall prove that it cannot be small. For any variable v, let $H(v)$ be the set of variable-label pairs (v, i) where $i \in [k]$ such that $C(v, i)$ is in $\mathcal{H}$. Consider a constraint e over variables $v_1, \ldots, v_t$. Let $\tilde{H}(e) := \cup_{i=1}^{t} H(v_i)$. It can be seen that there must be a choice of variable-label pairs $(v_i, j_i) \in H(v_i)$ for each $1 \leq i \leq t$ which constitutes a satisfying assignment to e. In other words $\tilde{H}(e)$ must *contain* a satisfying assignment to e. If not, then $\tilde{L}(e) \setminus \tilde{H}(e) \in G(e)$ is "good", and is not covered by $\mathcal{H}$. Note that this also implies that $H(v)$ is non-empty for every variable v.

The above analysis suggests a randomized way to assign labels to each variable based on the cover $\mathcal{H}$. For every variable choose a label uniformly at random from the labels corresponding to the set $H(v)$. For any constraint e over variables $v_1, \ldots, v_t$, let p_e be the probability that it is satisfied. Then, $p_e \geq \frac{1}{\prod_{i=1}^{t} |H(v_i)|}$. In expectation, the number of constraints satisfied is $\sum_{e \in E} p_e$. This quantity has to be at most the soundness of the instance $\mathcal{A}$ in the NO case, i.e. $\sum_{e \in E} p_e \leq |E| k^{-t+O(\sqrt{t})}$. This implies by Markov's Inequality, that for at least half of the constraints $e \in E$ over variables $v_1, \ldots, v_t$, we have $\frac{1}{\prod_{i=1}^{t} |H(v_i)|} \leq p_e \leq 2k^{-t+O(\sqrt{t})}$, and thus,

$$\frac{\sum_{i=1}^{t} |H(v_i)|}{t} \geq \left(\prod_{i=1}^{t} |H(v_i)| \right)^{\frac{1}{t}} \geq \frac{1}{(2k^{-t+O(\sqrt{t})})^{\frac{1}{t}}}. \tag{4}$$

Since each variable in V occurs in the same number of constraints, we have the following, $(|\mathcal{H}|/|V|) = (1/|V|) \sum_{v \in V} |H(v)| = \mathrm{E}_{e \in E} \left[\frac{\sum_{i=1}^{t} |H(v_i)|}{t} \right]$, where the inner summation in the final expression is over the variables $v_1, \ldots v_t$ of the constraint e in the outer expectation. Combining the above with the fact that Equation (4) is satisfied for at least half of the coonstraints we obtain,

$$|\mathcal{H}| \geq |V| \left(\frac{1}{2} \right) \left(\frac{1}{(2k^{-t+O(\sqrt{t})})^{\frac{1}{t}}} \right) \geq |V| \Omega \left(k^{1-O\left(\frac{1}{\sqrt{t}}\right)} \right).$$

Therefore we obtain a hardness factor of $\Omega \left(k^{1-O\left(\frac{1}{\sqrt{t}}\right)} \right)$. Since $t = \theta((\log k)^2)$, the hardness factor is $\Omega(k)$. Let d be the upper bound on the number of subsets that can cover any element in G. From our construction and previous calculations we have that $d = O(kt)$, and $\log |G| = O(kt)$, where $t = \theta((\log k)^2)$. Therefore, we obtain a hardness of approximation factor of $\Omega \left(\frac{\log |G|}{(\log \log |G|)^2} \right)$. By adding a dummy subset of $|G|$ new dummy elements to the ground set we can ensure that $\Delta = \Omega(|G|)$, and by adding another dummy element and $\log |G|$ additional dummy singleton sets containing that element, we can ensure that $d = \theta(\log |G|)$. This implies a hardness factor of $\Omega \left(\frac{\log \Delta}{(\log \log \Delta)^2} \right)$, which holds under the assumption that NP$\not\subseteq$ DTIME$(n^{\mathrm{poly}(\log n)})$.

4.3 Proof of Theorem 3

We begin by stating the following theorem proved by Khot and Ponnuswami [16] on the hardness of approximating Max-3LIN : the problem of satisfying as many of a system of three variable linear equations over $\mathbb{F}_2$.

Theorem 4. *[16] Given a 7-regular instance $\mathcal{A}$ of Max-3LIN over $\mathbb{F}_2$ on n variables, unless* NP $\subseteq$ DTIME$(2^{O(\log^2 N)})$*, there is no polynomial time algorithm to distinguish between the following two cases,*

YES CASE. There is an assignment to the variables of $\mathcal{A}$ that satisfies $1 - 2^{-\Omega(\sqrt{\log n})}$ fraction of the equations (completeness).

NO CASE. No assignment to the variables of $\mathcal{A}$ satisfies more than $1-\Omega(\log^{-3} n)$ fraction of the equations (soundness).

We shall combine the above result of [16] with the following "inner verifier" constructed by Khot and Saket [18]. A similar combination was done in [16] itself, however our construction is more optimized and we also use slightly different parameters.

Theorem 5. *Given an instance $\mathcal{A}$ of Max-3LIN over n variables with completeness $1 - c(n)$ and soundness $1 - s(n)$, for parameters m, r, k, ℓ and t there is a verifier V_{lin} which expects a proof Π where each position in the proof is expected to be labeled from $[k] = [2^r]$ such that,*

1. *V_{lin} uses $m \log n + O(\ell m r)$ random bits.*
2. *V_{lin} queries $t := \ell^2 + 2\ell$ positions from the proof.*
3. *If the instance $\mathcal{A}$ is a YES instance then there is a set Γ consisting of at most $mc(n)$ fraction (by the probability that V_{lin} queries any of them) of the positions in the proof, and an assignment τ^* to all the positions of the proof except those in Γ such that,*
   a. *(Strong Completeness) The verifier accepts on τ^* whenever none of the positions in Γ are queried.*
   b. *(Extendability) For any constraint q of the verifier which (possibly) queries positions from Γ, there is an assignment τ_q to the positions in Γ queried in q, such that τ^* extended by τ_q satisfies the constraint q.*
4. *If the instance $\mathcal{A}$ is a NO instance then the probability that the verifier accepts is at most $k^{-\ell^2} + \delta$, for $\delta^2 = (1 - s(n)^\kappa)^{(m/(\kappa r))}(k-1)^{\ell^2}$, for some universal constant κ.*

We first *regularize* the above inner verifier as follows. Let p be any position in the proof Π, and let R_p be the set of all random strings on which the verifier V_{lin} queries p. Replicate p with $|R_p|$ copies one for each string in R_p, for each position p. The new verifier simply chooses an element in R_p at random for each position p in the original query. Clearly, the new verifier queries each position with equal probability. It can also be seen that this does not change the completeness or the soundness of the verifier, and the strong completeness and extendability properties hold as well. The following lemma formalizes the modification to Theorem 5 that we can make.

Lemma 1. *The properties of the verifier V_{lin} in Theorem 4 hold with the following modifications : (i) The verifier V_{lin} queries each position with equal probability, and (ii) The number of random bits used by the verifier is $t(m \log n + O(\ell m r))$.*

In the combination of the above verifier with the Max-3Lin instance of [16] with n variables we have the completeness $c(n) = 2^{-\Omega(\sqrt{\log n})}$, and soundness $s(n) = \Omega(\log^3 n)$. We set the rest of the parameters as follows: take $m = \theta(\log^{3\kappa+3} n)$ and $r = \theta(\log\log n)$ such that $k = \theta(\log^{6\kappa+10} n)$. Additionally, we set $\ell = \theta(\log\log n)$. Now let Q be the set of all queries that the verifier makes. Clearly $\log|Q| = t(m\log n + O(\ell mr)) = O(\log^{3\kappa+5} n) = o(k)$. Furthermore, the fraction of positions in the subset Γ (as defined in Theorem 5) is $mc(n) \leq \theta(\log^{3\kappa+3} n) \cdot 2^{-\Omega(\sqrt{\log n})} = o(k^{-3})$. Since $s(n) = \Omega(\log^{-3} n)$, we have $\delta^2 = (1-s(n)^\kappa)^{(m/(\kappa r))}(k-1)^{\ell^2} = 2^{-\Omega(\log^2 n)}(k-1)^{\ell^2}$. Therefore, the soundness $(k^{-\ell^2} + \delta) = k^{-t+O(\sqrt{t})}$.

It is easy to see that with this setting of the parameters, the PCP verifier obtained in the above combination is an instance of Max-CSP-Reg(t,k) with variable set V identical to the set of positions Π, the set of constraints E identical to the set of queries Q, and the subset V' same as Γ such that the properties of Theorem 3 hold. This completes the proof of Theorem 3.

4.4 Limitations to Improving the Hardness Factor

In Section 4 we have shown a hardness factor of $\Omega\left(\frac{\log\Delta}{(\log\log\Delta)^2}\right)$ for Set-Cover where every subset has at most Δ elements, and each element of the ground set is in at most $\theta(\log\Delta)$ subsets. The two parts of this result are : a reduction to the Set-Cover problem from the Max-CSP problem; and the construction of a hard instance of Max-CSP with appropriate alphabet size, arity and hardness factor.

The second step is accomplished by running the t-query PCP test of Samorodnitsky and Trevisan [24] on a Hadamard Code based encoding of 3SAT introduced by Khot [13], which reduces the blowup of the PCP compared to the alphabet size. The t-query PCP test of [24] on an alphabet $[q]$ has a soundness of $q^{-t+O(\sqrt{t})}$. Notably, a better soundness of $q^t/O(qt)$ is achieved by more efficient PCP tests given in [8,2]. However these tests do not combine with the Hadamard Code encoding of [13] and instead are used along with the Long Code encoding of Unique Games, which leads to a large blowup of the PCP compared to the alphabet size. Another issue with the efficient tests of [8,2] is that t needs to be at least q^2, which will lead to weaker bounds for the canonical reduction to Set-Cover.

Thus, improving the hardness factor proved in Section 4 is connected to the question of designing efficient PCPs for Max-CSP problems over large alphabet, which in itself is a significant line of research. The current PCP techniques seem to fall short of yielding a tight bound for Set-Cover when $k = \theta(\log\Delta)$ and resolving the gap between the hardness result and the algorithmic upper bound remains an interesting open question.

Acknowledgements. We would like to thank Uriel Feige for helpful and insightful discussions on the topic.

References

1. Alon, N., Moshkovitz, D., Safra, M.: Algorithmic construction of sets for k-restrictions. The ACM Transactions on Algorithms 2(2), 153–177 (2006)
2. Austrin, P., Mossel, E.: Approximation Resistant Predicates from Pairwise Independence. Computational Complexity 18(2), 249–271 (2009)
3. 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)
4. Bar-Yehuda, R., Even, S.: A linear-time approximation algorithm for the weighted vertex cover problem. J. Algorithms 2(2), 198–203 (1981)
5. Chvatal, V.: A greedy heuristic for the set-covering problem. Math. Oper. Res. 4(3), 233–235 (1979)
6. Dinur, I., Guruswami, V., Khot, S., Regev, O.: A new multilayered PCP and the hardness of hypergraph vertex cover. SIAM J. Comput. 34(5), 1129–1146 (2005)
7. Feige, U.: A threshold of $\ln n$ for approximating set cover. J. ACM 45(4), 634–652 (1998)
8. Guruswami, V., Raghavendra, P.: Constraint Satisfaction over a Non-Boolean Domain: Approximation Algorithms and Unique-Games Hardness. In: Goel, A., Jansen, K., Rolim, J.D.P., Rubinfeld, R. (eds.) APPROX and RANDOM 2008. LNCS, vol. 5171, pp. 77–90. Springer, Heidelberg (2008)
9. Halperin, E.: Improved Approximation Algorithms for the Vertex Cover Problem in Graphs and Hypergraphs. SIAM J. Comput. 31(5), 1608–1623 (2002)
10. Hochbaum, D.: Approximation algorithms for the set covering and vertex cover problems. SIAM J. Comput. 11(3), 555–556 (1982)
11. 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, pp. 1005–1016. Springer, Heidelberg (2002)
12. Johnson, D.: Approximation algorithms for combinatorial problems. J. Comput. Syst. Sci. 9, 256–278 (1974)
13. Khot, S.: Improved inaproximability results for maxclique, chromatic number and approximate graph coloring. In: Proc. FOCS, pp. 600–609 (2001)
14. Khot, S.: On the power of unique 2-prover 1-round games. In: Proc. STOC, pp. 767–775 (2002)
15. Khot, S.: Online lecture notes for Probabilistically Checkable Proofs and Hardness of Approximation, Lecture 3 (scribed by Deeparnab Chakrabarty), www.cs.nyu.edu/~khot/pcp-lecnotes/lec3.ps
16. Khot, S., Ponnuswami, A.K.: Better Inapproximability Results for MaxClique, Chromatic Number and Min-3Lin-Deletion. In: Bugliesi, M., Preneel, B., Sassone, V., Wegener, I. (eds.) ICALP 2006. LNCS, vol. 4051, pp. 226–237. Springer, Heidelberg (2006)
17. Khot, S., Regev, O.: Vertex cover might be hard to approximate to within $2-\varepsilon$. J. Comput. System Sci. 74(3), 335–349 (2008)
18. Khot, S., Saket, R.: Hardness of Minimizing and Learning DNF Expressions. In: Proc. FOCS, pp. 231–240 (2008)
19. Krivelevich, M.: Approximate set covering in uniform hypergraphs. J. Algorithms 25(1), 118–143 (1997)

20. Lovasz, L.: On the ratio of the optimal integral and fractional covers. Disc. Math. 13, 383–390 (1975)
21. Lund, C., Yannakakis, M.: On the hardness of approximating minimization problems. J. ACM 31(5), 960–981 (1994)
22. Okun, M.: On the approximation of the vertex cover problem in hypergraphs. Discrete Optimization 2(1), 101–111 (2005)
23. Raz, R., Safra, M.: A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In: Proc. STOC, pp. 475–484 (2007)
24. Samorodnitsky, A., Trevisan, L.: A PCP characterization of NP with optimal amortized query complexity. In: Proc. STOC, pp. 191–199 (2000)

Hardness of Vertex Deletion and Project Scheduling*

Ola Svensson

EPFL, Switzerland
ola.svensson@epfl.ch

Abstract. Assuming the Unique Games Conjecture, we show strong inapproximability results for two natural vertex deletion problems on directed graphs: for any integer $k \geq 2$ and arbitrary small $\epsilon > 0$, the Feedback Vertex Set problem and the DAG Vertex Deletion problem are inapproximable within a factor $k-\epsilon$ even on graphs where the vertices can be almost partitioned into k solutions. This gives a more structured and therefore stronger UGC-based hardness result for the Feedback Vertex Set problem that is also simpler (albeit using the "It Ain't Over Till It's Over" theorem) than the previous hardness result.

In comparison to the classical Feedback Vertex Set problem, the DAG Vertex Deletion problem has received little attention and, although we think it is a natural and interesting problem, the main motivation for our inapproximability result stems from its relationship with the classical Discrete Time-Cost Tradeoff Problem. More specifically, our results imply that the deadline version is NP-hard to approximate within any constant assuming the Unique Games Conjecture. This explains the difficulty in obtaining good approximation algorithms for that problem and further motivates previous alternative approaches such as bicriteria approximations.

1 Introduction

Many interesting problems can be formulated as that of finding a large induced subgraph satisfying a desired property of a given (directed) graph. One of the most well studied such problems is the *Feedback Vertex Set (FVS)* problem where the property is acyclicity, i.e., given a directed graph $G = (V, E)$ we wish to delete the minimum number of vertices so that the resulting graph is acyclic. Another example is the *DAG Vertex Deletion (DVD)* problem, where we are given an integer k and a directed acyclic graph and we wish to delete the minimum number of vertices so that the resulting graph has no path of length[1] k.

* This research was supported by Grant 228021-ECCSciEng of the European Research Council.

[1] For notational convenience, we shall measure the length of a path in terms of the number of vertices it contains instead of the number of edges.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 301–312, 2012.
© Springer-Verlag Berlin Heidelberg 2012

The FVS problem and the related Feedback Arc Set problem was shown to be NP-complete already in Karp's seminal paper [9] and there is a long history of approximation algorithms for these problems. Leighton and Rao [13] first gave a $O(\log^2 |V|)$-approximation algorithm. Seymour [16] improved the approximation guarantee by showing that a certain linear program approximates the value within a factor $O(\log |V| \log \log |V|)$. Seymour's arguments were then generalized by Even et al. [5] to obtain the best known approximation algorithms achieving a factor $O(\log |V| \log \log |V|)$ even in weighted graphs.

Motivated by certain VLSI design and communication problems, Paik et al. [15] considered the DVD problem and showed it to be NP-complete on general graphs and polynomial time solvable on series-parallel graphs. One can also see that DVD for a fixed k is a special case of the Vertex Cover problem on k-uniform hypergraphs and has a fairly straightforward k-approximation algorithm.

In comparison to FVS, the DVD problem has received little attention and, although we think it is a natural problem, our main motivation for studying its approximability comes from its relationship (that we prove in Section 4) with the classical deadline version of the project scheduling problem known as the *Discrete Time-Cost Tradeoff problem*. Informally (see Section 4 for a formal definition of the Deadline problem), this is the problem where we are given a deadline and a project consisting of tasks related by precedence constraints, and the time it takes to execute each task depends, by a given cost function, on how much we pay for it. The objective is to minimize the cost of executing all the tasks in compliance with the precedence constraints so that they all finish within the given deadline. Due to its obvious practical relevance, the problem has been studied in various contexts over the last 50 years (see the paper [11] by Kelly and Walker for an early reference). Fulkerson [6] and Kelley [10] obtained polynomial time algorithms if all cost functions are linear. In contrast, the problem becomes NP-hard for arbitrary cost functions [3] and there is even no known constant factor approximation algorithm in the general case. However, better (approximation) algorithms have been obtained for special cases. For example, Grigoriev and Woeginger [7] gave polynomial time algorithms for special classes of precedence constraints and one of several algorithms by Skutella [17] is a bicriteria approximation that, for any $\mu \in (0, 1)$, approximates the Deadline problem within a factor $1/(1 - \mu)$ if the deadline is allowed to be violated by a factor $1/\mu$.

In summary, there are no known constant approximation algorithms for FVS, DVD, and the Deadline problem although few strong inapproximability results are known. The best known NP-hardness of approximation results follow from the fact that they are all as hard to approximate as Vertex Cover which is NP-hard to approximate within a factor 1.3606 [4]. It is indeed easy to see that Vertex Cover is a special case of FVS and DVD, and Grigoriev and Woeginger [7] gave an approximation-preserving reduction from Vertex Cover to the Deadline problem. If we assume the Unique Games Conjecture (UGC) [12], our understanding of the approximability of FVS becomes significantly better: the hardness of approximation result for Maximum Acyclic Subgraph by Guruswami

et al. [8] implies that it is NP-hard to approximate FVS within any constant factor assuming the UGC. However, the results in [8] use very sophisticated techniques that are not known to imply a similar hardness for DVD and the Deadline problem.

Even though the starting motivation of this work was to better understand the approximability of the Deadline problem (and DVD), the techniques that we develop also lead to a stronger UGC-based hardness result for FVS: similar to the recent results for Vertex Cover on k-uniform hypergraphs by Bansal and Khot [1,2], we show that, for any integer $k \geq 2$ and arbitrarily small $\epsilon > 0$, there is no $k - \epsilon$-approximation algorithm for FVS even on graphs where the vertices can be almost partitioned into k feedback vertex sets. Our reduction is also much simpler than the one in [8] (albeit using the "It Ain't Over Till It's Over" theorem) but is tailored for FVS and does not yield any inapproximability result for the Maximum Acyclic Subgraph problem. More importantly, our techniques also lead to an analogous result for the DVD problem (and thereby the Deadline problem). Formally, our results for the considered vertex deletion problems can be stated as follows.

Theorem 1. *Assuming the Unique Games Conjecture, for any integer $k \geq 2$ and arbitrary constant $\epsilon > 0$, the following problems are* NP*-hard:*

FVS: *Given a graph $G(V, E)$, distinguish between the following cases:*
- (Completeness): *there exist disjoint subsets $V_1, \ldots, V_k \subset V$ satisfying $|V_i| \geq \frac{1-\epsilon}{k}|V|$ and such that a subgraph induced by all but one of these subsets is acyclic.*
- (Soundness): *every feedback vertex set has size at least $(1 - \epsilon)|V|$.*

DVD: *Given a DAG $G(V, E)$, distinguish between the following cases:*
- (Completeness): *there exist disjoint subsets $V_1, \ldots, V_k \subset V$ satisfying $|V_i| \geq \frac{1-\epsilon}{k}|V|$ and such that a subgraph induced by all but one of these subsets has no path of length k.*
- (Soundness): *every induced subgraph of $\epsilon|V|$ vertices has a path of length $|V|^{1-\epsilon}$.*

Note that in the completeness cases, letting $V' = V \setminus (V_1 \cup \cdots \cup V_k)$, the sets $V' \cup V_i$ for $i = 1, \ldots, k$ are almost disjoint solutions of size at most $(\frac{1}{k} + \epsilon)|V|$ each. In contrast, any solution basically needs to delete all vertices in the soundness case (even to avoid paths of length $|V|^{1-\epsilon}$ for DVD).

When proving UGC-based inapproximability results, the main task is usually to design "gadgets" of the considered problems that simulate a so-called dictatorship test. Once we have such "dictatorship gadgets", the process of obtaining UGC-based hardness results often follows from (by now) fairly standard arguments. In particular, the main ideas needed for our reductions leading to Theorem 1 are already present in the design of the gadgets. We have therefore chosen to present those gadget constructions with less cumbersome notation in the conference version (Section 3) and the reductions from Unique Games can be found in the full version of the paper.

As alluded to above, our main interest in DVD stems from its relationship with the Deadline problem. More specifically, in Section 4, we give an approximation-preserving reduction from DVD to the Deadline problem that combined with Theorem 1 yields:

Theorem 2. *Conditioned on the Unique Games Conjecture, for every $C > 0$, it is* NP*-hard to find a C-approximation to the Deadline problem.*

This explains the difficulty in obtaining good approximation algorithms for the Deadline problem and also further motivates alternative approaches such as the bicriteria approach by Skutella [17] that approximates the Deadline problem within a constant if the deadline is allowed to be violated by a constant factor.

2 Preliminaries

2.1 Low Degree Influence and "It Ain't over Till It's over" Theorem

Let $[k] = \{0, 1, \ldots, k-1\}$. When analyzing our hardness reductions, we shall use known properties regarding the behavior of functions of the form $f : [k]^R \mapsto \{0,1\}$ depending on whether they have influential co-ordinates. Similar to [14, Section 3], we define the influence of the i-th co-ordinate by

$$\mathrm{Infl}_i(f) = \mathbb{E}_x[\mathrm{Var}(f)|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_R].$$

We note that if $f : \{-1,1\}^R \mapsto \{-1,1\}$ then this definition coincides with the intuitive expression $\Pr_x[f(x_1, \ldots, x_i, \ldots, x_R) \neq f(x_1, \ldots, -x_i, \ldots, x_R)]$.

It is well known that if we let $f = \sum_{\phi} \hat{f}(\phi) X_\phi$ be the multi-linear representation of f (where, analogous of the standard Fourier representation, the characters $(X_\phi)_{\phi \in [k]^R}$ define an orthonormal basis of the vector space of all functions $[k]^n \mapsto \mathbb{R}$) then the influence can also be expressed as

$$\mathrm{Infl}_i(f) = \sum_{\phi : \phi_i \neq 0} \hat{f}^2(\phi),$$

which motivates the following definition of the degree d-influence of the i-th co-ordinate:

$$\mathrm{Infl}_i^d(f) = \sum_{\phi : \phi_i \neq 0, |\phi| \leq d} \hat{f}^2(\phi).$$

As we shall not work directly with these definitions or with the multi-linear representation, we refer the reader to [14] for the precise definitions and cut the discussion short by mentioning the property of low degree influence that shall be crucial to us (which follows from that $\sum_\phi \hat{f}^2(\phi) = \mathbb{E}_x[f(x)^2] \leq 1$).

Observation 3. *For a boolean function $f : \{0,1\}^R \mapsto \{0,1\}$, the sum of all degree d-influences is at most d.*

We shall now introduce a simplified version of the "It Ain't Over Till It's Over" theorem that is sufficient for the applications in this paper. The first proof was given by Mossel et al. [14] and a more combinatorial proof of a simplified version (very similar to the one used here) was given by Bansal and Khot [1] who used it to prove tight inapproximability results for Vertex Cover and a classical single machine scheduling problem. In fact many of our ideas are inspired from [1]. For $x \in [k]^R$ and a subsequence $S_\epsilon = (i_1, \ldots, i_{\epsilon R})$ of ϵR not necessarily distinct indexes in $[R]$, let

$$C_{x,S_\epsilon} = \{z \in [k]^R : z_j = x_j \ \forall j \notin S_\epsilon\}$$

denote the sub-cube defined by fixing the co-ordinates not in S_ϵ according to x. Let also $f(C_{x,S_\epsilon}) \equiv 0$ denote the expression that f is identical to 0 on the sub-cube C_{x,S_ϵ}.

Theorem 4. *For every $\epsilon, \delta > 0$ and integer k, there exists $\eta > 0$ and integer d such that any $f : [k]^R \mapsto \{0,1\}$ that satisfies*

$$\mathbb{E}[f] \geq \delta \qquad \text{and} \qquad \forall i \in [R], \mathrm{Infl}_i^d(f) \leq \eta,$$

has

$$\Pr_{x,S_\epsilon}[f(C_{x,S_\epsilon}) \equiv 0] \leq \delta.$$

Here and throughout the paper, the probability over x, S_ϵ is such that x and S_ϵ are taken independently and uniformly at random. When ϵ is clear from the context we often also abbreviate S_ϵ by S. Note that the theorem says that a reasonably balanced function with no low degree influential co-ordinates has very low probability to be identical to 0 over the random choice of sub-cubes. In contrast, it is easy to see that a dictatorship function (on the boolean domain) $f(x) = x_s$, for some s, has $\Pr_{x,S_\epsilon}[f(C_{x,S_\epsilon}) \equiv 0] = \Pr_{x,S_\epsilon}[f(C_{x,S_\epsilon}) \equiv 1] \geq 1/2 - \epsilon$. It is this drastic difference that we will exploit in our hardness reductions.

2.2 Unique Games Conjecture

An instance of Unique Games $\mathcal{L} = (G(V, W, E), [R], \{\pi_{v,w}\}_{(v,w)})$ consists of a regular bipartite graph $G(V, W, E)$ and a set $[R]$ of labels. For each edge $(v, w) \in E$ there is a constraint specified by a permutation $\pi_{v,w} : [R] \mapsto [R]$. The goal is to find a labeling $\rho : (V \cup W) \mapsto [R]$ so as to maximize $val(\rho) := \Pr_{e \in E}[\rho \text{ satisfies } e]$, where a labeling ρ is said to satisfy an edge $e = (v, w)$ if $\rho(v) = \pi_{v,w}(\rho(w))$. For a Unique Games instance $\mathcal{L}$, we let $OPT(\mathcal{L}) = \max_{\rho : V \cup W \mapsto [R]} val(\rho)$. The now famous Unique Games Conjecture that has been extensively used to prove strong hardness of approximation results can be stated as follows.

Conjecture 1 ([12]). For any constants $\zeta, \gamma > 0$, there is a sufficiently large integer $R = R(\zeta, \gamma)$ such that, for Unique Games instances $\mathcal{L}$ with label set $[R]$ it is NP-hard to distinguish between:

- *(Completeness):* $OPT(\mathcal{L}) \geq 1 - \zeta$.
- *(Soundness):* $OPT(\mathcal{L}) \leq \gamma$.

3 Dictatorship Gadgets for Vertex Deletion Problems

We give fairly simple gadgets of the considered vertex deletion problems that informally corresponds to a dictatorship test in the following sense: (Completeness:) any dictatorship function $f : [k]^R \mapsto [k]$ (defined by $f(x) = x_s$ for some $s \in [R]$) corresponds to a good solution whereas (Soundness:) any non-trivial solution corresponds to a function $f : [k]^R \mapsto \{0, 1\}$ with a high influence co-ordinate. By fairly standard arguments, these gadgets are then used to obtain analogous hardness results assuming the Unique Games Conjecture (see the full version of the paper for details).

Throughout this section, we fix k to be an integer, $\epsilon, \delta > 0$ to be arbitrarily small constants, and let η and d be as in Theorem 4 (depending on k, ϵ and δ).

3.1 Feedback Vertex Set

We shall here describe a graph $G = (V, E)$ that naturally corresponds to a dictatorship test in the following sense:

- *(Completeness:)* A dictatorship function partitions the vertex set into subsets $V', V_0, \ldots, V_{k-1}$ satisfying $V_j \geq \frac{1-\epsilon}{k}|V|, |V'| \leq \epsilon|V|$, and for $j \in [k]$ the graph obtained by deleting $V' \cup V_j$ is acyclic.
- *(Soundness:)* Any feedback vertex set that deletes less than $(1 - 2\delta)|V|$ vertices corresponds to a function $f : [k]^R \mapsto \{0, 1\}$ with a co-ordinate i so that $\mathrm{Infl}_i^d(f) > \eta$.

Dictatorship Gadget. To make the analysis more intuitive, it will be convenient to first present a gadget that consists of two types of vertices that we refer to as *bit-vertices* and *test-vertices* and all arcs are between bit- and test-vertices:

- There is a bit-vertex b_x of weight ∞ for every $x \in [k]^R$.
- There is a test-vertex $t_{x,S}$ of weight 1 for every $x \in [k]^R$ and every sequence $S = (i_1, \ldots, i_{\epsilon R}) \in [R]^{\epsilon R}$ of ϵR not necessarily distinct indices.
- The arc incident to a test-vertex $t_{x,S}$ are the following. There is an arc $(b_z, t_{x,S})$ if $z \in C_{x,S}$ and an arc $(t_{x,S}, b_z)$ if $z \in C^{\oplus}_{x,S}$, where

$$C^{\oplus}_{x,S} = \{z \oplus 1 : z \in C_{x,S}\}$$

  (here $\oplus$ denotes addition mod k).

As the bit-vertices have weight ∞, they will never be deleted in an optimal solution. We can therefore obtain an unweighted graph G of same optimal value by omitting the bit-vertices and having an arc $(t_{x,S}, t_{x',S'})$ between two test vertices if there exists a bit-vertex b_z so that $(t_{x,S}, b_z)$ and $(b_z, t_{x',S'})$. The vertex set of G will therefore correspond to the set T of test-vertices. The analysis of G therefore follows from proving that (completeness:) any dictatorship function partitions the test-vertices as required and (soundness:) that any solution that deletes less than a fraction $1 - 2\delta$ of the test-vertices corresponds to a function with a co-ordinate of high influence.

Completeness. We show that a dictatorship function $f : [k]^R \mapsto [k]$ of index s naturally partitions the test-vertices into subsets $T', T_0, \ldots, T_{k-1}$ satisfying $T_j \geq \frac{1-\epsilon}{k}|T|, |T'| \leq \epsilon|T|$, and such that the sets $T' \cup T_j$ for $j \in [k]$ are almost disjoint feedback vertex sets of size at most $(\frac{1}{k} + \epsilon)|T|$ each.

As $f(x) = x_s$, it partitions the bit-vertices in k equal sized sets

$$B_j = \{b_x : f(x) = j\} \qquad \text{for} \qquad j \in [k].$$

We say that a test-vertex $t_{x,S}$ is good if $s \notin S$ and partition the good test-vertices into k equal sized sets

$$T_j = \{t_{x,S} : s \notin S \text{ and } f(x) = j\} \qquad \text{for} \qquad j \in [k].$$

The sets are of equal size since they are partitioned according to x and whether a test-vertex is good only depends on S. Furthermore, as at least a fraction $1 - \epsilon$ of the test-vertices are good we have that $|T_j| \geq \frac{1-\epsilon}{k}|T|$ for $j \in [k]$ and therefore the remaining test-vertices in T' are at most $\epsilon|T|$ many.

It remains to show that $T_j \cup T'$ defines a feedback vertex set for any $j \in [k]$. The key observation is that T_j only have incoming edges from bit-vertices in B_j and outgoing edges to bit-vertices in $B_{j\oplus 1}$. Indeed, consider a test-vertex $t_{x,S} \in T_j$ and an arc $(b_z, t_{x,S})$. By definition we have that $z \in C_{x,S}$ and as S is good we have that $f(z) = f(x) = j$, which implies that $z \in B_j$. The exact same argument implies that $t_{x,S}$ only has outgoing edges to $B_{j\oplus 1}$.

The graph obtained by deleting all bad test-vertices and one of the sets $T_0, T_1, \ldots, T_{Q-1}$ is therefore acyclic as required.

Soundness. Let A be the last $1/2$ fraction of the bit-vertices according to a topological sort of the graph. Let f_A be the indicator function of A. Note that a test-vertex $t_{x,S}$ has incoming arcs from all bit-vertices in $C_{x,S}$ and outgoing arcs to all bit-vertices in $C^{\oplus}_{x,S}$. Therefore, if a test-vertex $t_{x,S}$ is not deleted then we must have that either f_A is identical to 0 on $C_{x,S}$ (if $t_{x,S}$ is placed before the last bit-vertex for which f_A evaluates to 0) or identical to 1 on $C^{\oplus}_{x,S}$ (if $t_{x,S}$ is placed after the last bit-vertex for which f_A evaluates to 0) depending on where $t_{x,S}$ is placed according to the topological sort.

As $\mathbb{E}[f_A] = 1/2$, we have by Theorem 4 that if $\mathrm{Infl}^d_i(f_A) \leq \eta$ for all $i \in [R]$ then

$$\Pr_{x,S}[f(C_{x,S}) \equiv 0] \leq \delta$$

and

$$\Pr_{x,S}[f(C^{\oplus}_{x,S}) \equiv 1] = \Pr_{x,S}[f(C_{x,S}) \equiv 1] = \Pr_{x,S}[(1-f)(C_{x,S}) \equiv 0] \leq \delta.$$

Therefore, if the solution does not correspond to a function with a co-ordinate of high low-degree influence it must have deleted at least a fraction $1 - 2\delta$ of the test-vertices.

3.2 Dag Vertex Deletion Problem

We shall describe a directed acyclic graph (DAG) $G = (V, E)$ that naturally corresponds to dictatorship test in the following sense:

- *(Completeness:)* A dictatorship function partitions the vertex set into subsets $V', V_0, \ldots, V_{k-1}$ satisfying $V_j \geq \frac{1-\epsilon}{k}|V|, |V'| \leq \epsilon|V|$, and such that for $j \in [k]$ the graph obtained by deleting $V' \cup V_j$ has no path of length k.
- *(Soundness:)* Any graph obtained by deleting less than $(1 - 6\delta)|V|$ vertices either has a path of length $|V|^{1-\delta}$ or corresponds to a function $f : [k]^R \mapsto \{0, 1\}$ with a co-ordinate i such that $\mathrm{Infl}_i^d(f) > \eta$.

Dictatorship Gadget. As in Section 3.1, it will be convenient to first present a gadget that consists of two types of vertices that we refer to as *bit-vertices* and *test-vertices*, and all edges will be between bit- and test-vertices:

- The bit-vertices are partitioned into $L + 1$ bit-layers (L is selected below). Each bit-layer $\ell = 0, \ldots, L$ contains a bit-vertex b_x^ℓ of weight ∞ for every $x \in [k]^R$.
- Similarly, the test-vertices are partitioned into L test-layers. Each test-layer $\ell = 0, \ldots, L - 1$ has a test-vertex $t_{x,S}^\ell$ of weight 1 for every $x \in [k]^R$ and every sequence of indices $S = (i_1, \ldots, i_{\epsilon R}) \in [R]^{\epsilon R}$.
- The arcs are the following: there is an arc $(b_z^\ell, t_{x,S}^{\ell'})$ if $\ell \leq \ell'$ and $z \in C_{x,S}$, and there is an arc $(t_{x,S}^{\ell'}, b_z^\ell)$ if $\ell > \ell'$ and $z \in C_{x,S}^{\oplus}$.
- Finally, L is selected so as $\delta L \geq |T|^{1-\delta}$, where T is the set of test-vertices.

Note that, as there are only arcs from a bit-layer ℓ to a test-layer ℓ' if $\ell' \geq \ell$ and only arcs from a test-layer ℓ' to a bit-layer ℓ if $\ell > \ell'$, the constructed graph is acyclic. Similar to the gadget for FVS, the bit-vertices can be omitted to obtain an unweighted graph G (with the set T of test-vertices as vertices) with the same optimal value by having an arc between two test-vertices if there was a path between them through one bit-vertex. Note that a path in G of length k is a path in the gadget that consists of k test-vertices. When arguing about the gadget, we will therefore say that a *path has length k if it consists of k test-vertices*.

Similarly to Section 3.1, the analysis of G follows from proving that (completeness:) any dictatorship function partitions the test-vertices as required and (soundness:) that any solution that deletes less than a fraction $1 - 6\delta$ of the test-vertices either has a path of length $|T|^{1-\delta}$ or corresponds to a function with a co-ordinate of high influence.

Completeness. We show that a dictatorship function $f : [k]^R \mapsto [k]$ of index s naturally partitions the test-vertices into subsets $T', T_0, \ldots, T_{k-1}$ satisfying $T_j \geq \frac{1-\epsilon}{k}|T|, |T'| \leq \epsilon|T|$, and such that for $j \in [k]$ the graph obtained by deleting $T' \cup T_j$ has no path of length k.

This can be seen by the same arguments as in Section 3.1. Indeed if we "collapse" the different layers by identifying the different copies of bit- and test-vertices then the gadget constructed here is identical to the gagdet in that section. We can therefore (by the arguments of Section 3.1), partition the bit-vertices into k equal sized sets $B_0, B_1, \ldots, B_{k-1}$ and all but an ϵ fraction of the test-vertices into k equal sized sets $T_0, T_1, \ldots, T_{k-1}$ so that any test-vertex in T_j has only incoming arcs from bit-vertices in B_j and outgoing arcs to bit-vertices in $B_{j\oplus 1}$.

Any $j \in [k]$ therefore corresponds to a solution by removing an ϵ fraction of the test-vertices (i.e., the set T') and those test-vertices in T_j.

Soundness. Before proceeding to the analysis it will be convenient to consider a different but equivalent formulation of the problem.

First, note that in any solution to DVD, i.e., a subgraph so that each path contains less than k test-vertices, we can find a coloring χ (using for example depth-first search) that assigns a color in $\{1, 2, \ldots, k\}$ to the bit-vertices with the property that, for each remaining test-vertex, the maximum color assigned to its predecessors is strictly less than the minimum color assigned to its successors. Similarly, any such coloring χ can be turned into a solution to DVD by deleting those test-vertices, for which not all predecessors are assigned lower colors than all its successors. Furthermore, from the construction of the arcs, we can assume w.l.o.g that the coloring satisfies $\chi(b_x^\ell) \leq \chi(b_x^{\ell'})$ if $\ell \leq \ell'$.

From the above discussion, an equivalent formulation of DVD on the constructed instances is as follows: find a coloring χ that assigns a color in $\{1, 2, \ldots, k\}$ to each bit-vertex satisfying $\chi(b_x^\ell) \leq \chi(b_x^{\ell'})$ if $\ell \leq \ell'$ so as to minimize the number of unsatisfied test-vertices where a test-vertex $t_{x,S}^\ell$ is said to be satisfied if

$$\max_{z \in C_{x,S}} \chi(b_z^\ell) < \min_{z \in C_{x,S}^{\oplus}} \chi(b_z^{\ell+1}),$$

that is all its predecessors are assigned lower colors than its successors.

It will also be convenient to consider the following lower bound on the colors assigned to most bit-vertices in each layer: define the color $\chi(\ell)$ of a bit-layer $\ell = 0, 1, \ldots, L$ as the maximum color that satisfies $\Pr_x[\chi(b_x^\ell) \geq \chi(\ell)] \geq 1 - \delta$.

Now, with each test-layer $\ell = 0, 1, \ldots, L-1$ we associate the indicator function $f^\ell : [k]^R \mapsto \{0, 1\}$ defined as follows

$$f^\ell(x) = \begin{cases} 0 & \text{if} \quad \chi(b_x^{\ell+1}) > \chi(\ell), \\ 1 & \text{otherwise.} \end{cases}$$

The key observation for the soundness analysis is the following.

Claim. For $\ell = 0, \ldots, L-1$, assuming that $\mathrm{Infl}_i^d(f^\ell) \leq \eta$ for all $i \in [R]$: if a fraction 3δ of the test-vertices of test-layer ℓ are satisfied, then $\chi(\ell) < \chi(\ell+1)$.

Proof. As at least a fraction 3δ of the test-vertices of test-layer ℓ are satisfied,

$$\Pr_{x,S}\left[\max_{z \in C_{x,S}} \chi(b_z^\ell) < \min_{z \in C_{x,S}^{\oplus}} \chi(b_z^{\ell+1})\right] \geq 3\delta.$$

By the definition of $\chi(\ell)$ we have $\Pr_x[\chi(b_x^\ell) \geq \chi(\ell)] \geq 1-\delta$ and therefore

$$\Pr_{x,S}\left[\chi(\ell) < \min_{z \in C^{\oplus}_{x,S}} \chi(b_z^{\ell+1})\right] = \Pr_{x,S}\left[f^\ell(C_{x,S}) \equiv 0\right] \geq 2\delta.$$

As $\mathrm{Infl}_i^d(f^\ell) \leq \eta$ for all $i \in [R]$, Theorem 4 implies that $E[f^\ell] < \delta$ and hence $\chi(\ell+1) > \chi(\ell)$.

If a coloring satisfies more than a fraction 6δ of the test-vertices then at least a 3δ fraction of the test-layers are such that at least a fraction 3δ of the test-vertices of that layer are satisfied, which in turn by the preceding claim implies that either one of them corresponds to a function with a co-ordinate of high influence or $3\delta L$ many colors are needed (or equivalently the graph contains a path consisting of at least $3\delta L - 1 \geq \delta L \geq |T|^{1-\delta}$ test-vertices).

4 Discrete Time-Cost Tradeoff Problem

In the discrete time-cost tradeoff problem we are given a set J of activities together with a partial order $(J, <)$. Any execution of the activities must comply with the partial order, that is, if $j < k$ activity k may not be started until j is completed. The duration of an activity depends on how much resources that are spent on it. This tradeoff between time and cost for each job is described by a nonnegative cost function $c_j : \mathbb{R}_+ \mapsto \mathbb{R}_+ \cup \{\infty\}$, where $c_j(x_j)$ denotes the cost to run j with duration x_j. The project duration $t(x)$ of the realization x is the makespan (length) of the schedule which starts each activity at the earliest point in time obeying the precedence constraints and durations x_j. Given a deadline $T > 0$, the *Deadline problem* is that of finding the cheapest realization x that obeys the deadline, i.e., $t(x) \leq T$.

Theorem 5. *The Deadline problem is as hard to approximate as DVD.*

Proof. We reduce (in polynomial time) the problem of approximating DVD to that of approximating the Deadline problem. Given an instance of DVD, i.e., an integer k and a DAG $G(V, A)$ with the vertices ordered $0, 1, \ldots, n-1$ according to a topological sort, consider the instance of the Deadline problem defined as follows:

- The deadline T is set to n.
- The set J of activities contains three activities l_i, m_i, r_i for each vertex $i \in V = \{0, 1 \ldots, n-1\}$ with precedence constraints $l_i < c_i < r_i$ and cost functions

$$c_{l_i}(x) = \begin{cases} 0, & \text{if } x \geq i \\ \infty, & \text{otherwise} \end{cases} \qquad c_{m_i}(x) = \begin{cases} 0, & \text{if } x \geq 9/10 \\ 1, & \text{otherwise} \end{cases}$$

$$c_{r_i}(x) = \begin{cases} 0, & \text{if } x > n-1-i \\ \infty, & \text{otherwise .} \end{cases}$$

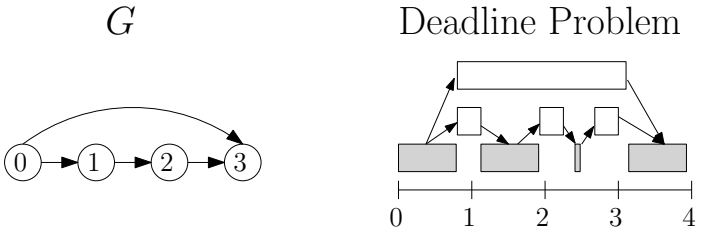

Fig. 1. For each vertex $i \in V$ the activity m_i is depicted in light gray (activities l_i and r_i are omitted). The activities corresponding to arcs are depicted in white. Finally, the depicted solution pays a cost of 1 for running activity m_2 in time 0.

In addition, there is an activity $a_{(i,j)}$ for each arc $(i,j) \in A$ with precedence constraints $m_i < a_{(i,j)} < m_j$ and cost function

$$c_{a_{(i,j)}}(x) = \begin{cases} 0, & \text{if } x \geq j - i - \frac{9}{10} + \frac{1}{10(k-1)} \\ \infty, & \text{otherwise.} \end{cases}$$

See Figure 1 for an example of the Deadline problem corresponding to a DVD instance G with $k = 3$.

Note that the cost functions of l_i, m_i, and r_i enforces that activity m_i has to be executed in the interval $[i, i+1)$ and that it will require time $9/10$ unless we pay a cost of 1 which allows us to run the activity in 0 time. Furthermore, as an activity $a_{(i,j)}$ always has duration (at least) $j - i - \frac{9}{10} + \frac{1}{10(k-1)}$, the start time s_j of activity m_j must be such that $s_j - j \geq s_i - i + \frac{1}{10(k-1)}$, where s_i is the start time of activity i. Using the fact that an activity m_i must run in the interval $[i, i+1)$ in order to obey the deadline, it follows that we have to pay a cost of 1 for at least one activity corresponding to each path of length k. By similar arguments, it also follows that this is also sufficient for having a realization respecting the deadline. Therefore, any solution to the Deadline problem naturally corresponds to a solution to DVD (and vice versa) by deleting those vertices corresponding to activities with a cost of 1.

References

1. Bansal, N., Khot, S.: Optimal long code test with one free bit. In: Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, pp. 453–462. IEEE Computer Society, Washington, DC (2009)
2. 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)

3. De, P., James Dunne, E., Ghosh, J.B., Wells, C.E.: The discrete time-cost tradeoff problem revisited. European Journal of Operational Research 81(2), 225–238 (1995)
4. Dinur, I., Safra, S.: On the hardness of approximating minimum vertex cover. Annals of Mathematics 162, 2005 (2004)
5. Even, G., (Seffi) Naor, J., Schieber, B., Sudan, M.: Approximating minimum feedback sets and multi-cuts in directed graphs. Algorithmica 20, 151–174 (1998)
6. Fulkerson, D.R.: A network flow computation for project cost curves. Management Science 7(2), 167–178 (1961)
7. Grigoriev, A., Woeginger, G.J.: Project scheduling with irregular costs: complexity, approximability, and algorithms. Acta Inf. 41(2-3), 83–97 (2004)
8. Guruswami, V., Håstad, J., Manokaran, R., Raghavendra, P., Charikar, M.: Beating the random ordering is hard: Every ordering CSP is approximation resistant. SIAM J. Comput. 40(3), 878–914 (2011)
9. Karp, R.: Reducibility among combinatorial problems. In: Miller, R., Thatcher, J. (eds.) Complexity of Computer Computations, pp. 85–103. Plenum Press (1972)
10. Kelley, J.E.: Critical-path planning and scheduling: Mathematical basis. Operations Research 9(3), 296–320 (1961)
11. Kelley Jr., J.E., Walker, M.R.: Critical-path planning and scheduling. Papers presented at the December 1-3, Eastern Joint IRE-AIEE-ACM Computer Conference, IRE-AIEE-ACM 1959 (Eastern), pp. 160–173. ACM, New York (1959)
12. Khot, S.: On the power of unique 2-prover 1-round games. In: Reif, J.H. (ed.) STOC, pp. 767–775. ACM (2002)
13. Leighton, T., Rao, S.: An approximate max-flow min-cut theorem for uniform multicommodity flow problems with applications to approximation algorithms. In: Proceedings of the 29th Annual Symposium on Foundations of Computer Science, SFCS 1988, pp. 422–431. IEEE Computer Society, Washington, DC (1988)
14. Mossel, E., O'Donnell, R., Oleszkiewicz, K.: Noise stability of functions with low influences: Invariance and optimality. Annals of Mathematics 171(1) (2010)
15. Paik, D., Reddy, S., Sahni, S.: Deleting vertices to bound path length. IEEE Trans. Comput. 43(9), 1091–1096 (1994)
16. Seymour, P.D.: Packing directed circuits fractionally. Combinatorica 15(2), 281–288 (1995)
17. Skutella, M.: Approximation algorithms for the discrete time-cost tradeoff problem. Mathematics of Operations Research 23(4), 909–929 (1998)

Approximation Guarantees for the Minimum Linear Arrangement Problem by Higher Eigenvalues

Suguru Tamaki[1] and Yuichi Yoshida[2]

[1] Graduate School of Informatics, Kyoto University, Japan
tamak@kuis.kyoto-u.ac.jp
[2] National Institute of Informatics, and Preferred Infrastructure, Inc., Japan
yyoshida@nii.ac.jp

Abstract. Given an undirected graph $G = (V, E)$ and positive edge weights $\{w_e\}_{e \in E}$, a *linear arrangement* is a permutation $\pi : V \to [n]$. The value of the arrangement is $\mathbf{val}(G, \pi) := \frac{1}{n} \sum_{e=\{u,v\}} w_e |\pi(u) - \pi(v)|$. In the *minimum linear arrangement problem* (MLA), the goal is to find a linear arrangement π^* that achieves $\mathbf{val}(G, \pi^*) = \mathrm{MLA}(G) := \min_\pi \mathbf{val}(G, \pi)$.

In this paper, we show that for any $\epsilon > 0$ and positive integer r, there is an $O(n^{r/\epsilon})$-time randomized algorithm which, given a graph G, returns a permutation π such that

$$\mathbf{val}(G, \pi) \leq \left(1 + \frac{2}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})}\right) \mathrm{MLA}(G) + O\left(\frac{\log n}{\sqrt{n}} \sum_{e \in E} w_e\right)$$

with high probability. Here $\mathcal{L}$ is the normalized Laplacian of G and $\lambda_r(\mathcal{L})$ is the r-th eigenvalue of $\mathcal{L}$. Our algorithm gives a constant factor approximation for regular graphs that are weak expanders.

Keywords: Semidefinite programming, Lasserre hierarchy, graph Laplacian, expander graph, ordering problem.

1 Introduction

Given an undirected graph $G = (V, E)$ and positive edge weights $\{w_e\}_{e \in E}$, a *linear arrangement* is a permutation $\pi : V \to [n]$. The value of the arrangement is $\mathbf{val}(G, \pi) := \frac{1}{n} \sum_{e=\{u,v\}} w_e |\pi(u) - \pi(v)|$. In the *minimum linear arrangement problem* (MLA), the goal is to find a linear arrangement π^* that achieves $\mathbf{val}(G, \pi^*) = \mathrm{MLA}(G) := \min_\pi \mathbf{val}(G, \pi)$. Since MLA is known to be NP-hard [9], we are interested in approximation algorithms for MLA.

Rao and Richa [18] presented an algorithm achieving approximation factors of $O(\log n)$. Charikar et al. [6] and Feige and Lee [8] independently improved the approximation factor to $O(\sqrt{\log n} \log \log n)$. There are also better algorithms for some families of graphs; Arora et al. [3] gave polynomial time approximation

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 313–324, 2012.
© Springer-Verlag Berlin Heidelberg 2012

scheme (PTAS) for dense graphs and Rao and Richa [18] obtained a constant factor approximation ratio for planar graphs.

On the inapproximability side, no NP-hardness of approximation results are known. The semidefinite programing (SDP) relaxation used for MLA were shown to have integrality gap $\Omega(\log \log n)$ by Devanur et al. [7]. Under the assumption that SAT has no probabilistic algorithm that runs in time $2^{n^{o(1)}}$, Ambühl et al. [1] show that MLA has no PTAS. Raghavendra et al. [17] show that there are no constant factor approximation algorithms for MLA under the Small-Set Expansion Hypothesis (SSEH) [16].

Our Contribution. In this paper, we present an algorithm with an approximation ratio which relies on the higher eigenvalues of the normalized Laplacian of an input graph. Given a graph $G = (V, E, \{w_e\})$, we denote by $\mathcal{L}$ the normalized Laplacian of G and by $\lambda_r(\mathcal{L})$ the r-th eigenvalue of $\mathcal{L}$. We show the following.

Theorem 1. *For any $\epsilon > 0$ and positive integer r, there is an $O(n^{r/\epsilon})$-time randomized algorithm which, given a graph G, returns a permutation π such that*

$$\mathbf{val}(G, \pi) \leq \left(1 + \frac{2}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})}\right) \mathrm{MLA}(G) + O\left(\frac{\log n \sum_{e \in E} w_e}{\sqrt{n}}\right)$$

with high probability.

Note that if $\mathrm{MLA}(G) = \omega(n^{-1/2} \log n \sum_{e \in E} w_e)$, then the above theorem gives an $O(1/\lambda_r)$-approximation factor. For example, we will show that $\mathrm{MLA}(G) = \omega(n^{-1/2} \log n \sum_{e \in E} w_e)$ if G is a regular graph and $\lambda_r / r^2 = \omega(n^{-1/2} \log n)$.

Our Technique. We basically follow the approach by [11] to relate the approximation ratio and eigenvalues of $\mathcal{L}$. We first formulate MLA as an integer program and consider its SDP relaxation. Our SDP is chosen from the Lasserre hierarchy [14] of r/ϵ rounds. Here, its solution has a vector $\boldsymbol{x}_{S,\alpha}$ for each set S of at most $r/\epsilon + 1$ vertices and an assignment $\alpha \in [n]^S$. Intuitively, vectors $\{\boldsymbol{x}_{S,\alpha}\}_{\alpha \in [n]^S}$ defines probability distribution of assignments over S. The vectors satisfy constraints on dot products to keep the consistency among these local probability distribution. Also, we add constraints among vectors so that each variable takes only one label and each label is chosen by one variable.

Given an optimal solution to the Lasserre SDP, we first choose an appropriate subset S^* of r/ϵ vertices. Then, for each assignment α^* to vertices in S^*, we randomly extend the assignment to all vertices by assigning, for each $u \in V \setminus S^*$ independently, a random value from u's marginal distribution based on $\boldsymbol{x}_{S \cup \{u\},\alpha}$ conditioned on the assignment α^* to S.

We note that the required round of the Lasserre hierarchy in the algorithm by [11] is proportional to the label size. This is because the error term is bounded by using eigenvalues in the Laplacian of the *constraint graph* instead of the original graph. However, since the label size of MLA is n, it will take n^n time to compute the optimal solution for the Lasserre hierarchy. To avoid this issue,

we apply the technique used to give an approximation algorithm for Unique Games in [11]. That is, we embed the set of n vectors $\{\boldsymbol{x}_{u,i}\}_{i\in[n]}$ for a vertex u into a single vector $\widetilde{\boldsymbol{x}}_u$ with some nice distance preserving properties. With the embedding, we can upper bound the error term by $\sum_{u\in V} d_u \|X_S^{\perp}\widetilde{\boldsymbol{x}}_u\|$, where d_u is the (weighted) degree of u and X_S is the projection matrix onto the span of $\{\widetilde{\boldsymbol{x}}_u\}_{u\in S}$. Then, we can apply the method by [11] to choose S so that the error term becomes at most $\frac{2}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})}$ times $\mathrm{MLA}(G)$.

A remaining issue is that the rounding above does not give a permutation. However, since the rounding is done independently for each $u \in V \setminus S$, the obtained integer solution is very close to a permutation. Thus, we can transform it a permutation at the cost of a tiny error.

Related Work. The idea of relating approximation ratios with eigenvalues of Laplacians was developed while designing approximation algorithms for Unique Games [12]. A Unique Game is a special case of maximum constraint satisfaction problems (Max CSPs), in which each constraint forms a permutation between labels over two variables. In [4], it was shown that, given a $(1-\epsilon)$-satisfiable instance, we can obtain a $(1-O(\frac{\epsilon}{\lambda_2(\mathcal{L}_G)}\log\frac{\epsilon}{\lambda_2(\mathcal{L}_G)}))$-satisfying assignment. Then, [13] gave an algorithm that uses higher eigenvalues. For a graph G, let $\mathrm{rank}_{\le\tau}(G)$ be the number of eigenvalues of the normalized Laplacian that is smaller than τ. In a breakthrough work [2], it is shown that we can obtain a $(1-\epsilon^{O(1)})$-satisfying assignment in $\exp(kn^{O(\epsilon)})$ time. The idea there is decomposing a graph into subgraphs S so that $\mathrm{rank}_{\le\epsilon^{O(1)}}(G[S]) \le kn^{O(\epsilon)}$ holds, where $G[S]$ is the graph induced by S and k is the label size.

Recently in independent works [5] and [11], it is shown that the Lasserre hierarchy is useful to obtain a good approximation for Unique Games when $\mathrm{rank}_{\le\tau}(G)$ is small. In [5], it is shown that, if each constraint has arity two, then we can approximate any instance of Max CSPs with an ϵ-fraction of error using an $O(k\cdot\mathrm{rank}_{\le 1-\Omega((\epsilon/k)^2)}(G)/\epsilon^4)$-round Lasserre hierarchy. In [11], it is shown that we can approximate Max Cut, Minimum Uncut, Minimum (Maximum) Bisection, Small Set Expansion and Unique Games within multiplicative error of $O(1/\lambda_r(G))$ using an r-round Lasserre hierarchy. The Lasserre hierarchy is also applied to obtain a similar approximation ratio for Sparsest Cut [10].

2 Preliminaries

In this paper, we consider undirected weighted graphs. An undirected weighted graph is represented as $G=(V,E,W)$, where V is a set of vertices, E is a set of undirected edges and $W=\{w_e\}_{e\in E}$ is a set of edge weights with $\sum_{e\in E} w_e = 1$. Let n denote the number of vertices in G, i.e., $n := |V|$. We define the degree of a vertex u as $d_u := \sum_{e=\{u,v\}} w_e$. Though we drop G from the notations, it must be clear from the context. Matrices L_G and $\mathcal{L}_G$ are the *Laplacian* and the *normalized Laplacian* of a graph $G=(V,E,W)$. That is,

$$L_G(u,v) := \begin{cases} -w_{\{u,v\}} \text{ if } u \neq v. \\ d_u \text{ if } u = v. \end{cases}$$
$$\mathcal{L}_G(u,v) := \begin{cases} -w_{\{u,v\}}/\sqrt{d_u d_v} \text{ if } u \neq v. \\ 1 \text{ if } u = v. \end{cases}$$

We denote by $\lambda_i(\mathcal{L}_G)$ the i-th smallest eigenvalue of $\mathcal{L}_G$.

We use define $[n] := \{1, 2, \ldots, n\}$, and S_n stands for the set of permutations from V to $[n]$.

2.1 The Minimum Linear Arrangement Problem

Given a graph $G = (V, E, W)$, we define the value of $\pi : V \to [n]$ as

$$\mathbf{val}(G, \pi) := \frac{1}{n} \sum_{\{u,v\} \in E} w_{\{u,v\}} |\pi(u) - \pi(v)|$$

and the *minimum linear arrangement* as $\mathrm{MLA}(G) := \min_{\pi \in S_n} \mathbf{val}(G, \pi)$.

In the next section, we consider an algorithm that outputs π which is approximately a permutation. Formally, $\pi : V \to [n]$ is called γ-*good* if for any i, j with $1 \leq i < j \leq n$,

$$j - i + 1 - \gamma \leq |\{k \in [n] \mid i \leq \pi^{-1}(k) \leq j\}| \leq j - i + 1 + \gamma$$

holds. Then, we have the following lemma.

Lemma 1. *If $\pi : V \to [n]$ is γ-good, then there exists $\tilde{\pi} \in S_n$ such that $\mathbf{val}(G, \tilde{\pi}) \leq \mathbf{val}(G, \pi) + \frac{2\gamma}{n} \sum_{e \in E} w_e$.*

Proof. Define $\tilde{\pi} \in S_n$ to satisfy $\tilde{\pi}(u) < \tilde{\pi}(v)$ if and only if $\pi(u) \leq \pi(v)$ for any $u, v \in V$. We can easily obtain such $\tilde{\pi}$ by ranking $u \in V$ according to $\pi(u)$ and using any tie-breaking rule. Note that $|\pi(u) - \pi(v)| \leq |\tilde{\pi}(u) - \tilde{\pi}(v)| + 2\gamma$ by γ-goodness.

2.2 Lasserre Hierarchy of Semidefinite Programs

Let $\binom{V}{\leq r+1}$ denote the set of subsets of V of size at most $r+1$. For any $S \in \binom{V}{\leq r+1}$ and $\alpha \in [n]^S$, $\boldsymbol{x}_{S,\alpha}$ is a row vector in $\mathbb{R}^{\Upsilon}$, where Υ is some positive integer. For $\alpha \in [n]^S$ and $S' \subseteq S$, we write $\alpha_{S'}$ to denote the projection of α on the coordinates indexed from S'. For $\alpha \in [n]^S$ and $\beta \in [n]^T$, we say α and β are *consistent* if $\alpha_{S \cap T} = \beta_{S \cap T}$. If α and β are consistent, we denote by $\alpha \circ \beta \in [n]^{S \cup T}$ a vector which is consistent to α and β. We consider the following r-round Lasserre SDP relaxation of MLA.

$$
\begin{aligned}
&\min \tfrac{1}{n} \sum_{e=\{u,v\}\in E} w_e \sum_{i,j\in[n]} \langle \boldsymbol{x}_{u,i}, \boldsymbol{x}_{v,j}\rangle |i-j| \\
&\text{s.t. } \langle \boldsymbol{x}_{S,\alpha}, \boldsymbol{x}_{T,\beta}\rangle = 0 && \forall |S\cup T| \le r+1, \alpha_{S\cap T} \neq \beta_{S\cap T} \\
&\quad \langle \boldsymbol{x}_{S,\alpha}, \boldsymbol{x}_{T,\beta}\rangle = \langle \boldsymbol{x}_{A,\alpha'}, \boldsymbol{x}_{B,\beta'}\rangle && \forall |S\cup T| \le r+1, S\cup T = A\cup B, \\
&\quad && \alpha\circ\beta = \alpha'\circ\beta' \\
&\quad \sum_{i\in[n]} \boldsymbol{x}_{v,i} = \boldsymbol{x}_\emptyset && \forall v \in V \\
&\quad \sum_{v\in V} \boldsymbol{x}_{v,i} = \boldsymbol{x}_\emptyset && \forall i \in [n] \\
&\quad |\boldsymbol{x}_\emptyset|^2 = 1.
\end{aligned}
$$

By Lasserre constraints, for any $S \in \binom{V}{\le r+1}, u \in S$ and $\alpha \in [n]^{S\setminus\{u\}}$, $\sum_{\beta\in[n]^u} \boldsymbol{x}_{S,\alpha\circ\beta} = \boldsymbol{x}_{S\setminus\{u\},\alpha}$. Therefore,

Fact 2 (Observation 6 in the full version of [11]). *For all* $S \in \binom{V}{r}$,

$$\operatorname{span}\left(\{\boldsymbol{x}_{S,\alpha}\}_{\alpha\in[n]^S}\right) \supseteq \operatorname{span}\left(\{\boldsymbol{x}_{u,i}\}_{u\in S, i\in[n]}\right)$$

holds.

2.3 Matrix Analysis

We present some useful facts from matrix analysis theory. For a square matrix A, $\mathsf{Tr}(A)$ denotes the trace of A. For a matrix B, B^{tr} stands for the transposed matrix of B. For a vector $\boldsymbol{a}$, we define $\overline{\boldsymbol{a}} := \boldsymbol{a}/\|\boldsymbol{a}\|$.

Fact 3. *For a matrix* $X \in \mathbb{R}^{\Upsilon\times V}$, *let* X_u *be the* u*-th column vector of* X *and* L *be the Laplacian matrix of some graph* $G = (V, E, W)$. *Then,*

$$\mathsf{Tr}(X^{\mathrm{tr}} X L) = \sum_{e=\{u,v\}\in E} w_e \|X_u - X_v\|^2.$$

We use the following notations.

$$
\begin{aligned}
\Pi_S &:= \sum_{\alpha\in[n]^S} \overline{\boldsymbol{x}_{S,\alpha}}^{\mathrm{tr}}\, \overline{\boldsymbol{x}_{S,\alpha}}, \quad \Pi_S^\perp := I - \Pi_S, \\
P_S &:= \sum_{u\in S, i\in[n]} \overline{\boldsymbol{x}_{u,i}}^{\mathrm{tr}}\, \overline{\boldsymbol{x}_{u,i}}, \\
X_S &:= \sum_{u\in S} \overline{X_u}\, \overline{X_u}^{\mathrm{tr}}, \quad X_S^\perp := I - X_S.
\end{aligned}
$$

Here, Π_S is the projection matrix onto the span of $\{\boldsymbol{x}_{S,\alpha}\}_{\alpha\in[n]^S}$. Note that $\boldsymbol{x}_{S,\alpha}$'s are row vectors. Similarly, P_S is the projection matrix onto the span of $\{\boldsymbol{x}_{u,i}\}_{u\in S, i\in[n]}$. Finally, X_S is the projection matrix onto the span of $\{X_u\}_{u\in S}$.

Proposition 1 (Lemma 30 in the full version of [11]). *Given $X \in \mathbb{R}^{\Upsilon \times V}$ and a Laplacian matrix $L \in \mathbb{R}^{V \times V}$, for any positive integer r and positive constant $\epsilon > 0$, there exists a set of r/ϵ columns $S \in \binom{V}{r/\epsilon}$ of X such that*

$$\mathsf{Tr}(X^{\mathrm{tr}} X_S^{\dagger} X D) \leq \frac{\mathsf{Tr}(X^{\mathrm{tr}} X L)}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})}$$

where $\mathcal{L}$ is the corresponding normalized Laplacian and D is the diagonal matrix of L. Furthermore such S can be found in deterministic $O(rn^4)$ time.

3 The Main Rounding Algorithm and Its Analysis

Let $\{\boldsymbol{x}_{S,\alpha}\}$ be an optimal solution to the (r/ϵ)-round Lasserre SDP. Our approach to round $\{\boldsymbol{x}_{S,\alpha}\}$ to a labeling $\pi : V \to [n]$ is similar to the propagation sampling used in [4,5,11]. Our rounding algorithm is described in Algorithm 1.

Algorithm 1. Algorithm for labeling in time $O(n^{r/\epsilon})$

Input: $\{\boldsymbol{x}_{S,\alpha}\}$ and $\mathcal{S} \subseteq V$ of size at most r/ϵ.
Output: $\pi : V \to [n]$.
Choose $\alpha \in [n]^{\mathcal{S}}$ with probability $\|\boldsymbol{x}_{\mathcal{S},\alpha}\|^2$.
For each $u \in V$, set $\pi(u) = i \in [n]$ with probability $\frac{\langle \boldsymbol{x}_{\mathcal{S},\alpha}, \boldsymbol{x}_{u,i} \rangle}{\|\boldsymbol{x}_{\mathcal{S},\alpha}\|^2}$.

When we write $\pi \sim \mathcal{D}_{\mathcal{S}}$, $\pi : V \to [n]$ is sampled by the rounding algorithm. We first observe that the algorithm above yields an integer solution close to a permutation.

Lemma 2. *For any $\mathcal{S} \in \binom{V}{r/\epsilon}$,*

$$\Pr_{\pi \sim \mathcal{D}_{\mathcal{S}}}[\pi \text{ is not } O(\sqrt{n}\log n)\text{-good}] = o(1).$$

Proof. First note that $\{\pi(u)\}_{u \in [n] \setminus \mathcal{S}}$ are independent random variables. Define $p_k := \Pr[\pi(k) \in \{i, i+1, \ldots, j\}]$, then $\sum_{k \in [n]} p_k = j - i + 1$ by Lasserre constraints.

We can show the followings by the Chernoff-Hoeffding bound.

$$\Pr[|\{k \in [n] \mid i \leq \pi^{-1}(k) \leq j\}| < j - i + 1 - \omega(\sqrt{n}\log n)] = O(1/n^3)$$

and

$$\Pr[|\{k \in [n] \mid i \leq \pi^{-1}(k) \leq j\}| > j - i + 1 + \omega(\sqrt{n}\log n)] = O(1/n^3).$$

By the union bound over all pairs (i, j), we obtain the conclusion of the lemma.

We prove the following in Section 3.1.

Theorem 4. *Define* $\mathbf{alg}(G) := \mathbf{E}_{\pi\sim\mathcal{D}_\mathcal{S}} \mathbf{val}(G,\pi)$

$$= \mathop{\mathbf{E}}_{\pi\sim\mathcal{D}_\mathcal{S}} \left[\frac{1}{n} \sum_{e=\{u,v\}\in E} w_e \sum_{i,j\in[n]} \Pr[\pi(u)=i, \pi(v)=j]|i-j| \right].$$

Then, There exists $\mathcal{S} \in \binom{V}{r/\epsilon}$ *such that*

$$\mathbf{alg}(G) \le \left(1 + \frac{2}{(1-\epsilon)\lambda_{r+1}}\right) \mathrm{MLA}(G).$$

Corollary 1. *There exists* $\mathcal{S} \in \binom{V}{r/\epsilon}$ *such that for any* $\delta > 0$,

$$\Pr_{\pi\sim\mathcal{D}_\mathcal{S}} \left[\mathbf{alg}(G) \le (1+\delta)\left(1 + \frac{2}{(1-\epsilon)\lambda_{r+1}}\right) \mathrm{MLA}(G) \right] \ge \delta.$$

Combining Lemmas 1, 2 and Corollary 1, we have Theorem 1.

3.1 Upper Bounds on the Expected Value

In this section, we see the proof of Theorem 4. Let us denote the optimum value of the (r/ϵ)-round Lasserre SDP as

$$\mathbf{sdp}(G) := \frac{1}{n} \sum_{e=\{u,v\}\in E} w_e \sum_{i,j\in[n]} \langle \boldsymbol{x}_{u,i}, \boldsymbol{x}_{v,j} \rangle |i-j|.$$

We use the following convention: $\boldsymbol{x}_{u,<i} = \sum_{j<i} \boldsymbol{x}_{u,j}$ and $\boldsymbol{x}_{u,\ge i} = \sum_{j\ge i} \boldsymbol{x}_{u,j}$. We define a vector

$$\widetilde{\boldsymbol{x}}_u := \frac{1}{\sqrt{n}} (\boldsymbol{x}_{u,<1}, \ldots, \boldsymbol{x}_{u,<n}, \boldsymbol{x}_{u,\ge 1}, \ldots, \boldsymbol{x}_{u,\ge n})$$

and for a matrix $P \in \mathbb{R}^{\Upsilon\times\Upsilon}$,

$$\widetilde{P\boldsymbol{x}}_u := \frac{1}{\sqrt{n}} (P\boldsymbol{x}_{u,<1}, \ldots, P\boldsymbol{x}_{u,<n}, P\boldsymbol{x}_{u,\ge 1}, \ldots, P\boldsymbol{x}_{u,\ge n}).$$

Then, we can see that the objective function of the Lasserre SDP is an expected distance between $\widetilde{\boldsymbol{x}}_u$ and $\widetilde{\boldsymbol{x}}_v$.

Lemma 3

$$\mathbf{sdp}(G) = \frac{1}{2} \sum_{e=\{u,v\}\in E} w_e \|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2.$$

Similarly, we can prove the following expression on the expected value of the output by the rounding algorithm.

Lemma 4. *Given a set* $\mathcal{S} \in \binom{V}{r/\epsilon}$,

$$\mathbf{alg}(G) = \frac{1}{2} \sum_{e=\{u,v\}\in E} w_e \|\widetilde{\Pi_\mathcal{S}\boldsymbol{x}}_u - \widetilde{\Pi_\mathcal{S}\boldsymbol{x}}_v\|^2 + \sum_{u\in V} d_u \|\widetilde{\Pi_\mathcal{S}^\perp \boldsymbol{x}}_u\|^2.$$

Combining the above lemmas with the following lemmas, we have Theorem 4.

Lemma 5. *For any set* $\mathcal{S} \in \binom{V}{r/\epsilon}$,

$$\|\widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_v\|^2 \leq \|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2.$$

Lemma 6. *For any* $\epsilon > 0$, *there exists a set* $\mathcal{S} \in \binom{V}{r/\epsilon}$ *such that*

$$\sum_{u \in V} d_u \|\widetilde{\Pi_{\mathcal{S}}^{\perp}\boldsymbol{x}}_u\|^2 \leq \frac{2}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})} \cdot \frac{1}{2} \sum_{e=\{u,v\} \in E} w_e \|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2.$$

In the next section, we give proofs of Lemmas 3, 4, 5 and 6.

3.2 Proofs of Lemmas

Note that some proofs of technical lemmas in this section are omitted due to page limitations.

Proof of Lemma 3. First we need the following technical lemma.

Lemma 7. *Let* $\{\boldsymbol{x}_{u,i}\}_{u \in V, i \in [n]}$ *be vectors such that* $\sum_{i \in [n]} \boldsymbol{x}_{u,i}$ *is a unit vector for each* $u \in V$.

$$n - \sum_{i \in [n]} \langle \boldsymbol{x}_{u,<i}, \boldsymbol{x}_{v,<i} \rangle - \sum_{i \in [n]} \langle \boldsymbol{x}_{u,\geq i}, \boldsymbol{x}_{u,\geq i} \rangle = \sum_{j,k \in [n]} \langle \boldsymbol{x}_{u,j}, \boldsymbol{x}_{v,k} \rangle |j-k|.$$

The following lemma immediately implies Lemma 3.

Lemma 8

$$\frac{1}{n} \sum_{i,j \in [n]} \langle \boldsymbol{x}_{u,i}, \boldsymbol{x}_{v,j} \rangle |i-j| = \frac{1}{2} \|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2.$$

Proof. First, we check

$$\|\widetilde{\boldsymbol{x}}_u\|^2 = \frac{1}{n} \sum_{j \in [n]} \|\boldsymbol{x}_{u,j}\|^2 (n-j) + \frac{1}{n} \sum_{j \in [n]} \|\boldsymbol{x}_{u,j}\|^2 j = \sum_{j \in [n]} \|\boldsymbol{x}_{u,j}\|^2 = 1.$$

Thus,

$$\begin{aligned}
\frac{1}{2}\|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2 &= \frac{1}{2}\|\widetilde{\boldsymbol{x}}_u\|^2 + \frac{1}{2}\|\widetilde{\boldsymbol{x}}_v\|^2 - \langle \widetilde{\boldsymbol{x}}_u, \widetilde{\boldsymbol{x}}_v \rangle \\
&= \frac{1}{n}\left(n - \sum_i \langle \boldsymbol{x}_{u,<i}, \boldsymbol{x}_{v,<i} \rangle - \sum_i \langle \boldsymbol{x}_{u,\geq i}, \boldsymbol{x}_{v,\geq i} \rangle \right) \\
&= \frac{1}{n} \sum_{i,j \in [n]} \langle \boldsymbol{x}_{u,i}, \boldsymbol{x}_{v,j} \rangle |i-j|, \quad \text{(from Lemma 7)}
\end{aligned}$$

which implies the lemma.

Proof of Lemma 4. First we need the following technical lemma.

Lemma 9

$$\frac{1}{n}\sum_{i,j\in[n]}\langle \Pi_{\mathcal{S}}\boldsymbol{x}_{u,i}, \Pi_{\mathcal{S}}\boldsymbol{x}_{v,j}\rangle|i-j| = \frac{1}{2}\|\widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_v\|^2 + \frac{1}{2}(\|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2 + \|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_v\|^2).$$

Note that

$$\begin{aligned}\Pr_{\pi\sim\mathcal{D}_S}[\pi(u)=i,\pi(v)=j] &= \sum_{\alpha\in[n]^S}\|\boldsymbol{x}_{\mathcal{S},\alpha}\|^2\frac{\langle \boldsymbol{x}_{\mathcal{S},\alpha},\boldsymbol{x}_{u,i}\rangle}{\|\boldsymbol{x}_{\mathcal{S},\alpha}\|^2}\frac{\langle \boldsymbol{x}_{\mathcal{S},\alpha},\boldsymbol{x}_{v,j}\rangle}{\|\boldsymbol{x}_{\mathcal{S},\alpha}\|^2}\\ &= \sum_{\alpha\in[n]^S}\langle \overline{\boldsymbol{x}}_{\mathcal{S},\alpha},\boldsymbol{x}_{u,i}\rangle\langle \overline{\boldsymbol{x}}_{\mathcal{S},\alpha},\boldsymbol{x}_{v,j}\rangle = \langle \Pi_{\mathcal{S}}\boldsymbol{x}_{u,i},\Pi_{\mathcal{S}}\boldsymbol{x}_{v,j}\rangle.\end{aligned}$$

From Lemma 9, we have (recall the definition of $\mathbf{alg}(G)$ in Theorem 4)

$$\begin{aligned}\mathbf{alg}(G) &= \frac{1}{2}\sum_{e=\{u,v\}\in E} w_e\left(\|\widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_v\|^2 + \|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2 + \|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_v\|^2\right)\\ &= \frac{1}{2}\sum_{e=\{u,v\}\in E} w_e\|\widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_v\|^2 + \sum_{u\in V} d_u\|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2.\end{aligned}$$

Proof of Lemma 5.

$$\|\widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_v\|^2 \le \|\widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}}\boldsymbol{x}_v\|^2 + \|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u - \widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_v\|^2 = \|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2.$$

Proof of Lemma 6. First note that

$$\|\widetilde{\Pi_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2 \le \|\widetilde{P_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2$$

by Fact 2.

Let X be the matrix in $\mathbb{R}^{(n\times\Upsilon)\times V}$ whose u-th column is $\widetilde{\boldsymbol{x}}_u^{\mathrm{tr}}$ and $X_{\mathcal{S}}, X_{\mathcal{S}}^{\perp}$ be defined as

$$X_{\mathcal{S}} := \sum_{u\in S}\overline{\widetilde{\boldsymbol{x}}_u}^{\mathrm{tr}}\,\overline{\widetilde{\boldsymbol{x}}_u}, \quad X_{\mathcal{S}}^{\perp} := I - X_{\mathcal{S}},$$

Then we have:

Lemma 10

$$\|\widetilde{P_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2 \le \|X_{\mathcal{S}}^{\perp}\widetilde{\boldsymbol{x}}_u\|^2.$$

Thus,

$$\sum_{u\in V} d_u\|\widetilde{P_{\mathcal{S}}^{\perp}}\boldsymbol{x}_u\|^2 \le \sum_{u\in V} d_u\|X_{\mathcal{S}}^{\perp}\widetilde{\boldsymbol{x}}_u\|^2 = \mathsf{Tr}(X^{\mathrm{tr}}X_{\mathcal{S}}^{\perp}XD)$$

and by choosing $\mathcal{S}$ as Proposition 1,

$$\le \frac{\mathsf{Tr}(X^{\mathrm{tr}}XL)}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})} = \frac{2}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})}\cdot\frac{1}{2}\sum_{e=\{u,v\}\in E} w_e\|\widetilde{\boldsymbol{x}}_u - \widetilde{\boldsymbol{x}}_v\|^2.$$

4 Lower Bounds for MLA by Eigenvalues

The goal of this section is showing that $\mathrm{MLA}(G)$ can be bounded below by $\Omega(\lambda_r(\mathcal{L}_G)/r^2)$. We only consider d-regular graphs, that is, $d_v = d$ for every $v \in V$.

Definition 1. *The (edge) expansion of S is*

$$\phi_G(S) = \frac{|E(S)|}{d|S|}$$

where $E(S)$ is the total weight of edges in G crossing from S to its complement. We define the r-way expansion of G as

$$\rho_G(r) = \min_{S_1,\ldots,S_r \subseteq V} \max_{i \in [r]} \phi_G(S_i)$$

where the minimum is over all collections of r non-empty disjoint subsets $S_1, \ldots, S_r \subseteq V$.

Theorem 5 ([15]). *For every graph G and every integer $r \geq 1$, we have*

$$\frac{\lambda_r(\mathcal{L}_G)}{2} \leq \rho_G(r) \leq O(r^2)\sqrt{\lambda_r(\mathcal{L}_G)}.$$

For a (closed line) segment $T = [a, b]$, we define $\min T = a, \max T = b$, and $|T| = b - a$. For a segment $T \subseteq [1, n]$, we define $\mathbf{val}(G, \pi, T)$ as the value involved in the segment T. That is,

$$\mathbf{val}(G, \pi, T) = \frac{1}{|T|} \mathop{\mathbf{E}}_{(u,v)\in E} |\mathrm{chop}_T(\pi(u)) - \mathrm{chop}_T(\pi(v))|$$

where

$$\mathrm{chop}_T(c) = \begin{cases} c & \text{if } \min T \leq c \leq \max T \\ \min T & \text{if } c < \min T \\ \max T & \text{if } \max T < c \end{cases}$$

Note that $\mathbf{val}(G, \pi) = \frac{1}{n} \sum_{j=1}^{t} \mathbf{val}(G, \pi, T_i)|T_i|$ holds for any partition $T_1, \ldots, T_t$ of $[1, n]$ into segments.

Theorem 6. *For a graph G and any integer $r \geq 1$, $\lambda_r(\mathcal{L}_G) = O(r^2 \mathrm{MLA}(G))$ holds.*

Proof. For simplicity, we assume n is a multiple of $2r - 1$. The proof can be easily modified for general case.

Let $\theta = \mathrm{MLA}(G)$ and π be the permutation achieving θ. Let $t = 2r - 1$ and define a partition $T_1, \ldots, T_{2r-1}$ of $[n]$ into segments so that each T_i has the same

size $\frac{n}{2r-1}$. Note that $\mathbf{val}(G, \pi, T_i) \leq (2r-1)\theta$ for each $i \in [t]$. Define $s_0 = 0$ and $s_r = n$. Then, there exists some $s_i \in \{\min T_{2i}, \ldots, \max T_{2i} - 1\}$ for each $i \in [r-1]$ satisfying the following property: If we define $S_i = \{s_{i-1} + 1, \ldots, s_i\}$ for each $i \in [r]$, then the number of edges crossing the boundary between S_i and S_{i+1} is at most $(2r-1)\theta \cdot m$ for each $i \in [r-1]$. Since each S_i has size at least $\frac{n}{2r-1}$, we have

$$\phi_G(S_i) \leq \frac{(2r-1)\theta \cdot m}{d\frac{n}{2r-1}} = (2r-1)^2\theta.$$

Thus, $\frac{\lambda_r(\mathcal{L}_G)}{2} \leq \rho_G(r) \leq (2r-1)^2\theta$ from Theorem 5.

From the above theorem, we can show that our rounding algorithm gives a $O(1/\lambda_{r+1}(\mathcal{L}_G))$-factor approximation for d-regular weak expander graphs.

Corollary 2. *For any $\epsilon > 0$ and positive integer r, there is an $O(n^{r/\epsilon})$-time randomized algorithm which, given a d-regular graph G with $\lambda_r/r^2 = \omega(n^{-1/2}\log n)$, returns a permutation π such that*

$$\mathbf{val}(G, \pi) \leq \left(1 + \frac{2}{(1-\epsilon)\lambda_{r+1}(\mathcal{L})} + o(1)\right) \mathrm{MLA}(G)$$

with high probability.

Proof. By Theorem 6, we have $\frac{\log n \sum_{e \in E} w_e}{\sqrt{n}} = o(\mathrm{MLA}(G))$ in Theorem 1.

We remark that Theorem 6 cannot be extended to weighted graphs as is. Let $G = (V, E)$ be a graph consisting of a clique of size $n-1$ with an edge (u, v) attached, where u is a vertex in the clique and v is the only vertex outside of the clique. Then, we set weights of edges in the clique to 1 and the weight of the edge (u, v) to an extremely large value, say 2^n. We can obtain $\mathrm{MLA}(G) = O(1/n)$ by putting v next to u in the permutation. Now, we consider $\rho_G(r)$ for $r \geq 3$. For any set of r non-empty disjoint subsets $S_1, \ldots, S_r \subseteq V$, we have some set S_i such that $|S_i| \leq \frac{n-2}{r-2}$ and S_i only contains vertices in $V \setminus \{u, v\}$. Then, we can show $\phi_G(S_i) = 1 - c/r$ for some constant c since S_i is a small subset in the clique of size $n-1$. From Theorem 5, it follows that $\lambda_r(\mathcal{L}_G) \geq (\frac{1}{r^2}(1 - \frac{c}{r}))^2 = \Omega(\frac{1}{r^4})$, which contradicts the statement of Theorem 6.

Acknowledgement. The authors thank anonymous reviewers for suggestions on how to improve the presentation.

References

1. Ambühl, C., Mastrolilli, M., Svensson, O.: Inapproximability results for maximum edge biclique, minimum linear arrangement, and sparsest cut. SIAM Journal on Computing 40(2), 567–596 (2011)

2. Arora, S., Barak, B., Steurer, D.: Subexponential algorithms for unique games and related problems. In: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 563–572 (2010)
3. Arora, S., Frieze, A.M., Kaplan, H.: A new rounding procedure for the assignment problem with applications to dense graph arrangement problems. Mathematical Programming 92(1), 1–36 (2002)
4. Arora, S., Khot, S., Kolla, A., Steurer, D., Tulsiani, M., Vishnoi, N.K.: Unique games on expanding constraint graphs are easy: extended abstract. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC), pp. 21–28 (2008)
5. Barak, B., Raghavendra, P., Steurer, D.: Rounding semidefinite programming hierarchies via global correlation. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 472–481 (2011); Full version: Electronic Colloquium on Computational Complexity (ECCC) TR11-65
6. Charikar, M., Hajiaghayi, M.T., Karloff, H.J., Rao, S.: ℓ_2^2 spreading metrics for vertex ordering problems. Algorithmica 56(4), 577–604 (2010)
7. Devanur, N.R., Khot, S., Saket, R., Vishnoi, N.K.: Integrality gaps for sparsest cut and minimum linear arrangement problems. In: Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC), pp. 537–546 (2006)
8. Feige, U., Lee, J.R.: An improved approximation ratio for the minimum linear arrangement problem. Information Processing Letters 101(1), 26–29 (2007)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
10. Guruswami, V., Sinop, A.K.: Certifying graph expansion and non-uniform sparsity via generalized spectra. CoRR abs/1112.4109 (2011)
11. Guruswami, V., Sinop, A.K.: Lasserre hierarchy, higher eigenvalues, and approximation schemes for quadratic integer programming with PSD objectives. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 482–491 (2011); Full version: Electronic Colloquium on Computational Complexity (ECCC) TR11-66
12. Khot, S.: On the power of unique 2-prover 1-round games. In: Proceedings of the 34th Annual ACM Symposium on Theory of Computing (STOC), pp. 767–775 (2002)
13. Kolla, A., Tulsiani, M.: Playing random and expanding unique games (unpublished manuscript)
14. Lasserre, J.: An explicit equivalent positive semidefinite program for nonlinear 0-1 programs. SIAM Journal on Optimization 12(3), 756–769 (2002)
15. Lee, J.R., Gharan, S.O., Trevisan, L.: Multi-way spectral partitioning and higher-order cheeger inequalities. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC), pp. 1117–1130 (2012); Full version: arXiv:1111.1055
16. Raghavendra, P., Steurer, D.: Graph expansion and the unique games conjecture. In: Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC), pp. 755–764 (2010)
17. Raghavendra, P., Steurer, D., Tulsiani, M.: Reductions between expansion problems. In: Proceedings of the 27th Annual IEEE Conference on Computational Complexity (CCC) (to appear, 2012); Full version: Electronic Colloquium on Computational Complexity (ECCC) TR10-172
18. Rao, S., Richa, A.W.: New approximation techniques for some linear ordering problems. SIAM Journal on Computing 34(2), 388–404 (2004)

Circumventing d-to-1 for Approximation Resistance of Satisfiable Predicates Strictly Containing Parity of Width Four
(Extended Abstract)

Cenny Wenner*

KTH – Royal Institute of Technology and Stockholm University
cenny@cwenner.net

Abstract. Håstad established that any predicate $P \subseteq \{0,1\}^m$ containing parity of width at least three is approximation resistant for almost satisfiable instances. However, in comparison to for example the approximation hardness of Max-3SAT, the result only holds for almost satisfiable instances. This limitation was mitigated by O'Donnell, Wu, and Huang under the d-to-1 Conjecture. They showed the threshold result that if a predicate *strictly* contains parity of width at least three, then it is approximation resistant also for satisfiable instances. We extend modern hardness of approximation techniques by Mossel et al. to projection games, eliminating dependencies on the degree of projections via Smooth Label Cover, and prove unconditionally the same approximation resistance result for predicates of width four.

1 Introduction

We study the limits of approximation for $\mathcal{NP}$-hard Constraint Satisfaction Problems (CSP). A canonical example of such problems is Max-3SAT which in the CSP framework can be denoted Max-CSP$^{\pm}$(3OR).[1] In Max-3SAT, we are given Boolean variables $x_1, \ldots, x_n$ and clauses of the form "$a \vee b \vee c$", where each literal a, b, and c is either a variable x_i or its negation. A solution to an instance is an assignment to the variables, the value of a solution is the number of clauses it satisfies, and the value of an instance is the maximum value over all solutions. In the CSP framework, we substitute the value 'true' for 1 and 'false' for 0. In greater generality, a Max-CSP$^{\pm}$(P) problem is defined by specifying the width-m predicate P applied to the set of literals instead of 3OR.

Seeing how 3SAT is $\mathcal{NP}$-complete to solve exactly, we turn our attention to efficient approximations. We say that a solution is a c-approximation if its value is at least c times the value of an instance. In particular, for Max-3SAT, assigning each variable a random value yields a 7/8-approximation in expectation.

* Supported by ERC Advanced Investigator Grant 226203.

[1] The definition of Max-CSP is sometimes ambiguous and we have added a plus-minus superscript to signify that constraints may involve negations of variables.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 325–337, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Unfortunately, this is essentially the best efficient approximation of the problem as Max-3SAT is known to be $\mathcal{NP}$-hard to approximate better than $7/8+\epsilon$ for every $\epsilon > 0$ [9]. In fact, even if the instance is perfectly satisfiable, i.e., positive instances have value 1, it is $\mathcal{NP}$-hard to satisfy more than a fraction $7/8+\epsilon$.

We call predicates with this property for *approximation resistant*, i.e. when a random assignment essentially achieves the best polynomial-time approximation factor assuming $\mathcal{P} \neq \mathcal{NP}$. For simplicity, our treatise hereafter works under this assumption. A convenient consequence of showing that a predicate is approximation resistant is that it establishes the optimal polynomial-time approximation factor of the predicate, up to lower-order terms. In particular, this quantity is $2^{-m}|\mathrm{P}|$ and is also called the *random assignment threshold*. The celebrated work by Håstad [9] demonstrated that a number of well-studied predicates are approximation resistant and has been a starting point of a long line of strong inapproximability results. In fact, assuming Khot's Unique Games Conjecture (UGC) [12], most predicates of sufficiently large width are known to be approximation resistant [1].

Of particular interest to us is the predicate *odd parity* defined by $(a_1, \ldots, a_m) \in P$ if the number of $a_i = 1$ is odd. The predicate *even parity* is defined analogously. Håstad showed that (either) parity is *hereditarily approximation resistant*; meaning that not only is parity approximation resistant, but so is any predicate $\mathrm{Q} \subseteq \{0,1\}^m$ containing parity. By containing, we mean in the set sense. However, in comparison to e.g. Max-3SAT, the approximation resistance result holds with respect to *almost satisfiable* instances. Formally, letting Q be an arbitrary predicate containing parity, for any $\epsilon, \epsilon' > 0$, given a Max-CSP$^{\pm}(Q)$ instance with value at least $1-\epsilon'$, it is $\mathcal{NP}$-hard to find a solution with value at least $2^{-m}|\mathrm{Q}| + \epsilon$.

For parity, the use of almost-satisfiable instances is necessary: perfectly-satisfiable instances can via Gaussian elimination be solved in polynomial time, whereas almost-satisfiable instances are hard to approximate within $1/2+\epsilon$. It is not immediately clear whether other approximation-resistant predicates containing parity should be easy or hard for satisfiable instances, and indeed 3SAT is as hard to approximate for almost-satisfiable as perfectly-satisfiable instances.

Assuming Khot's d-to-1 Conjecture [12], this question was settled by O'Donnell and Wu [19] for $m = 3$ and later generalized to $m \geq 3$ by Huang [10]. They showed the remarkable threshold that any predicate *strictly* containing parity is approximation resistant also for perfectly-satisfiable instances. More specifically, O'Donnell and Wu showed the hereditary approximation resistance, for satisfiable instances, of the predicate "Not-Two", the predicate which accepts three bits that are either of odd parity or all zeroes.

The result of O'Donnell and Wu follows from the construction of a Probabilistically Checkable Proof (PCP) reducing from an outer verifier to Max-CSP$^{\pm}$(Not-Two). The outer verifier may be taken as a black-box constraint satisfaction problem called Label Cover. In Label Cover, one is given a bipartite graph $G = (U \cup V, E)$, a "small" label set K, a "large" label set L, and for each edge $e \in E$ an associated projection $\pi_e : L \to K$. Solutions assign each

vertex $u \in U$ a label $\lambda(u)$ from K and each vertex $v \in V$ a label $\lambda(v)$ from L, and the value of a solution is the fraction of edges $\{u, v\} \in E$ for which $\lambda(u) = \pi_{\{u,v\}}(\lambda(v))$. One can show that it is $\mathcal{NP}$-hard for every $\varepsilon > 0$ to distinguish whether a Label Cover instance has value 1 (the *completeness*) or value at most ε (the *soundness*) for sufficiently large label sets K and L depending on ε.

Reductions from Label Cover are today standard in hardness of approximation. For Boolean constraints, such proofs typically involve semantically replacing $\lambda(u)$ and $\lambda(v)$ with $2^{2^{|K|}}$ and $2^{2^{|L|}}$ Boolean variables, respectively. These variables are respectively viewed as functions $f^u : \{-1,1\}^K \to \{-1,1\}$ and $g^v : \{-1,1\}^L \to \{-1,1\}$. The intention, for positive instances, is to set these functions to *dictators*. That is, setting $f^u(\mathbf{x}) = x_{\lambda(u)}$ and $g^v(\mathbf{y}) = y_{\lambda(v)}$. For negative instances, there are however no guarantees that the functions are set according to this coding scheme. Reducing to a Max-CSP$^{\pm}(P)$ instance, and viewing P as the indicator of its set, points of such functions are passed as arguments to P. The value of an edge $\{u, v\}$ in the Label Cover instance is thereby reduced to, for some integer T, to the value of the expectation

$$\mathbf{E}_{\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(T)},\mathbf{y}^{(T+1)},\ldots,\mathbf{y}^{(m)}}\left[P(f^u(\mathbf{x}^{(1)}),\ldots,f^u(\mathbf{x}^{(T)}),g^v(\mathbf{y}^{(T+1)}),\ldots,g^v(\mathbf{y}^{(m)})\right], \tag{1}$$

where the arguments are chosen according to a *test distribution*. Equation 1 is the starting point for Fourier analysis of PCP's. For approximation resistance, this involves first taking the Fourier expansion of P and proceeding to bounds terms of the forms $\mathbf{E}[\prod f^u]$, $\mathbf{E}[\prod g^v]$, and/or $\mathbf{E}[\prod f^u \prod g^v]$. For work most similar to this treatise, T is typically one, rendering the first kind of term(s) trivial to bound while terms of the third kind become $\mathbf{E}[f^u \prod g^v]$. Finally, a central parameter to this work is the *(maximum) degree of projections*, $d = d(\varepsilon) = \max_{e \in E} \max_{i \in K} |\pi_e^{-1}(i)|$. That is, the greatest number of labels from the large label set which share projection. For present $\mathcal{NP}$-hard constructions of Label Cover, $d \to \infty$ as $\varepsilon \to 0$.

The construction by O'Donnell and Wu is similar to that of Håstad for Max-3-Lin-2, i.e. parity on three bits. Working with almost satisfiable instances, Håstad could define his test distribution such that each argument to a function was somewhat "noised". O'Donnell and Wu, working with perfectly satisfiable instances, could not afford this. Instead they made use of the "unpredictability" of a predicate which strictly contains parity. Defining a test distribution close to that for Max-3-Lin-2, but with somewhat bounded correlation between the arguments to the functions, they used theorems by Mossel [14] to argue that the analysis behave roughly as though the arguments were somewhat noised. Following this, the effect of only being "close" to the uniform distribution over parity had to be bounded. For this, they extended modern techniques for analyzing PCP's. They introduced a "matrix-notation technique" to bound terms of the form $\mathbf{E}[\prod g^v]$ while for terms of the form $\mathbf{E}[f^u \prod g^v]$, they used a coordinate-wise distribution-substitution method to bound the terms by influences. Their method has subsequently found other applications [21,20].

We note that all of the steps in the above proof involves degenerative dependencies on d, the degrees of projections. This promoted the use of the d-to-1 Conjecture which states that Label Cover remains $\mathcal{NP}$-hard for arbitrarily low soundness ε, even for a fixed degree of projections d. The d-to-1 Conjecture, and its more well-known sibling, the UGC, have proven highly useful for establishing (conditional) hardness of approximation results. However, despite remarkable efforts to prove or refute these conjectures, we appear to be nowhere near settling the conjectures nor theorems serving equivalent purposes. There has however been recent progress towards circumventing the conjectures for particular problems [20,6].

2 Our Contributions and Techniques

Our main contribution is to circumvent the d-to-1 Conjecture to show that any predicate strictly containing parity of width four is approximation resistant for satisfiable instances unless $\mathcal{P} = \mathcal{NP}$. The overarching steps of our proof follow those of O'Donnell and Wu, and our main technical contribution is to extend the methods of Mossel et al. [16,14]. Subject to smoothness, we show that the analysis behaves roughly the same subject to what we call projected noise as it does subject to independent noise; more on this below. Additionally, we employ a multivariate invariance principle extended to projection games which avoids dependencies on the degree of projections d. We note that a similar elimination of the dependency on d, using different methods, was recently shown by O'Donnell and Wright [17] for a particular two-variable case.

The Smooth Label Cover problem serves an integral role in our proofs. The problem is a variant of Label Cover which roughly states that if one looks at a vertex $v \in V$ and two labels $j, j' \in L$, over the random choice e of edges incident v, the two labels are unlikely to share projection, i.e. the event '$\pi_e(j) = \pi_e(j')$' has arbitrarily low positive measure over the choice of $u\tilde{v}$. The problem was first defined by Khot to show approximation hardness of Coloring [11]. Subsequently, Feldman et al. [5] used it for the hardness of learning monomials, and Guruswami et al. [6] to establish exciting optimal inapproximability results for two geometric results where previously only optimal UG-hardness results were known. More intimately related to our work, Khot and Saket [13] used smoothness to show $20/27 + \epsilon$ approximation hardness of Max-CSP on satisfiable instances.

Subject to smoothness, we relate what we call *projected noise* to *non-projected* or *independent noise*. The former is introduced by conventional techniques from correlation bounds, while the latter is needed to decode from influences without a dependency on the degree of a projection. The issue with the former is that projected noise does not significantly affect functions which depend on a large number of coordinates with the same projection. However, under Smooth Label Cover, any function which depends on many coordinates must essentially depend in expectation on many coordinates with different projections. With the limited unpredictability of the distribution we define, we can via correlation

bounds introduce projected noise independent of d and subsequently turn it into independent noise due to smoothness.

With a test distribution which behaves roughly as though arguments were independently noised, we wish to bound expectations of the form $\mathbf{E}[\prod g^v]$ and $\mathbf{E}[f^u \prod g^v]$. For the former, we employ smoothness, partial independence of the test distribution, and hypercontractivity to argue that the expectation is roughly the same as for a distribution where all coordinates $j \in L$ are drawn independently, as in Unique Games. Since our test distribution is arbitrarily close to being independent over the arguments $\{\mathbf{y}^{(t)}\}_t$ in this setting, the expectation $\mathbf{E}[\prod g^v]$ is close to 0.

Finally, we extend the coordinate-wise distribution-substitution method of O'Donnell and Wu, to show a multivariate invariance theorem similar to Mossel's [14] but where bounds do not depend on the degree of projections d [22]. This permits us to effortlessly bound terms of the form $\mathbf{E}[f^u \prod g^v]$. In fact, the soundness analysis of a term $\mathbf{E}[f^u \prod g]$ involving functions on both the small and large label sets – often considered the hardest part of soundness analysis – becomes the easiest step subject to this theorem.

It may be pedagogical to discuss what we require to employ our steps. For noise introduction, it suffices, with smoothness, that each string $\mathbf{y}^{(r)}$ has in the marginal distribution over a label $j \in L$ bounded correlation to arguments $\{\mathbf{y}^{(t)}\}_{t \neq r}$ conditioned on $\{\mathbf{x}^{(t)}\}_t$. For bounding products of the form $\mathbf{E}[\prod g^v]$, we require noise, smoothness, and a roughly $m/2$-wise independent balanced distribution for $\{\mathbf{y}^{(t)}\}_t$. For bounding products of the form $\mathbf{E}[f^u \prod g^v]$ in terms of influences, we require the weak conditions of uniform marginals and that any single string $\mathbf{y}^{(r)}$ is independent of $\{\mathbf{x}^{(t)}\}_t$. We note that the last step does not employ smoothness and in particular pairwise independence suffices.

3 Preliminaries

3.1 Basic Notation

For any real p, we denote by $\bar{p} = 1 - p$, while for a set A from a possibly implicit universe $\mathcal{U}$, $\bar{\text{A}}$ refers to its complementary set $\mathcal{U} \setminus \text{A}$. We use Iverson notation $[S]$ where S is a true/false statement to denote 1 whenever S is true and 0 otherwise. For a natural number n, the integral interval $\{1, \ldots, n\}$ is denoted $[n]$. In this treatise, we deal extensively with correlated spaces $\mathcal{P} = (\prod_{t=1}^{m} \Omega_t, \mu)$ over finite domains. When the sample space is clear from the context, we may also specify measures instead of probability spaces, and vice versa. Given an index set $A \subseteq [m]$, we call Ω_A the product space $\prod_{t \in A} \Omega_t$. Vectors may for clarity be denoted either by bold font. Given a tuple $\mathbf{x} = (x_i)_{i \in A}$ and a bijection $\sigma : A \leftrightarrow A$, $\mathbf{x} \circ \sigma$ denotes the tuple $(x_{\sigma(i)})_{i \in A}$.

3.2 Operators on Probability Spaces

Tensoring. Given a probability space $\mathcal{P} = (\Omega, \mu)$, the n^{th} tensor power of $\mathcal{P}$ is $\mathcal{P}^{\otimes n} = (\Omega^n, \mu' = \mu^{\otimes n})$ where $\mu'(\omega_1, \ldots, \omega_n) = \mu(\omega_1) \cdots \mu(\omega_n)$.

Noise Operators. So called noised functions are standard when analyzing PCP's and we extend the notion somewhat to encompass also probability spaces.

Definition 1. *Let $\mathcal{P} = (\Omega_1, \mu)$ be a probability space, n an natural number, and $f : \Omega^n \to \mathbb{R}$ a function on $\mathcal{P}^{\otimes n}$. The noise operator, also called the Bonami-Beckner operator, $\mathrm{T}_{\mathcal{P},\gamma}(f) : \Omega^n \to \Omega^n$ with parameter $\gamma \in [0,1]$ applied to f takes an argument $\mathbf{x} = (x_i)_{i\in[n]}$, and yields the expectation of f where for every i, x_i is independently resampled from $\mathcal{P}$ with probability $\bar{\gamma}$.*

We generally let the distribution be implicit. The noise operator is more commonly defined by a parameter specifying the noise, whereas we specify the *correlation*, a more natural quantity in our eyes. The relation between the two definitions is immediate, substituting γ for $\bar{\gamma}$.

It is convenient for our proofs to extend the definition of noise operators to probability spaces. In particular, let $\mathcal{P} = (\prod^m \Omega_t, \mu)$ be a correlated probability space, $A \subseteq [m]$ an index set, and γ a parameter. Then, $\mathrm{T}^A_\gamma \mathcal{P}$ is defined as the probability space which first draws from $\mathcal{P}$ and with probability $\bar{\gamma}$ resamples Ω_A from its marginal of μ. When A is singleton $\{x\}$, we merely denote the noise operator by T^x_γ rather than $\mathrm{T}^{\{x\}}_\gamma$. As an example, let $\mathcal{P} = (\Omega_1 \times \Omega_2, \mu)$ be a correlated space, and, for $t = 1, 2$, consider functions $f_t : \Omega_t^n \to \mathbb{R}$; then, $\mathbf{E}_{\mathrm{T}^2_\gamma \mathcal{P}^{\otimes n}}[f_1 f_2] = \mathbf{E}_{\mathcal{P}^{\otimes n}}[f_1 \mathrm{T}_\gamma f_2]$.

The Projection Operator. In order to conveniently analyze projection-game-based PCP's, we introduce a *projection operator* on correlated spaces. Intuitively, the operator yields a correlated space which first samples a subset of spaces Ω_A and then a number of times independently samples the remaining spaces $\Omega_{\bar{A}}$ conditioned on Ω_A.

Definition 2. *The* degree-d projection *from an index set $A \subseteq [m]$ on a correlated space $\mathcal{P} = (\prod^m \Omega_t, \mu)$ is defined as $\mathcal{P}^{d\text{-proj-}A} \stackrel{\text{def}}{=} (\prod^m \Omega'_t, \mu')$, where $\Omega'_t = \Omega_t$ if $t \in A$ and otherwise Ω^d_t, and*

$$\mu'(\omega'_1, \ldots, \omega'_n) = \mathbf{P}_\mu(\Omega_A = \boldsymbol{\omega}'_A) \prod_{i=1}^{d} \mathbf{P}_\mu\big(\forall_{t\notin A} \Omega_t = \omega'_{t,i} \mid \Omega_A = \boldsymbol{\omega}'_A\big)\,.$$

3.3 Influences

A useful concept of functions is the *influence* of a coordinate. Intuitively, for a function $f : \{-1,1\}^n \to \mathbb{R}$, the influence of coordinate i is how much $f(\mathbf{x})$ changes on average with x_i. When analyzing positive instances in long-code-based PCP's, the functions in question are *dictators* of the encoded assignments; formally, $f^u(\mathbf{x}) = x_{\lambda(u)}$ where $\lambda(u)$ is the assignment to the vertex u in the reduced-from Label Cover instance. In the other direction, whenever a protocol accepts with a non-negligible probability over a random assignment, one would like to argue that the functions must essentially have significant influences and additionally so, for multiple functions, of coordinates consistent with projections.

Definition 3. *Let $f : \Omega^n \to \mathbb{R}$ be a function and $i \in [n]$ a coordinate. The influence of coordinate i is $\mathrm{Inf}_i(f) = \mathbf{E}_{\mathbf{x}_{-i}}[\mathbf{Var}_{x_i}[f(\mathbf{x})]]$, where the implicit distributions are uniform over Ω^n.*

In a similar way, *noisy influences* are defined as $\mathrm{Inf}_i^{(\gamma)}(f) \stackrel{\text{def}}{=} \mathrm{Inf}_i(\mathrm{T}_\gamma f)$ where $\gamma \in [0,1]$ is noise parameter. We note that the total influence of a function with codomain $[-1,1]$ can be of the order n while the total noisy influence for $\gamma < 1$ is always bounded from above by a constant depending only on γ.

3.4 Correlations

Intimately connected with noise operators is the concept of *correlation* between sample spaces. We note that correlations are always bounded by one and noise operators applied to individual sample spaces can only decrease correlation.

Definition 4. *The correlation $\rho(\Omega_1, \Omega_2;\ \mu)$ between Ω_1 and Ω_2 with respect to the probability space $\mathcal{P} = (\Omega_1 \times \Omega_2, \mu)$ is*

$$\rho_{\mathcal{P}}(\Omega_1, \Omega_2) \stackrel{\text{def}}{=} \rho(\Omega_1, \Omega_2;\ \mathcal{P}) \stackrel{\text{def}}{=} \max_{\phi,\psi} \mathbf{E}_\mu[\phi\psi]\,,$$

where the maximum is over functions $\phi : \Omega_1 \to \mathbb{R}, \psi : \Omega_2 \to \mathbb{R}$ such that $\mathbf{E}[\phi] = 0$ and $\mathbf{Var}[\phi] = \mathbf{Var}[\psi] = 1$.

3.5 Smooth Label Cover

Formally, we define the smoothness of a Label Cover instance as follows.

Definition 5. *A* Label Cover *instance is (J,κ)-smooth if for any vertex $v \in V$ and any set of labels $S \subseteq L, |S| \leq J$, over a uniformly at random neighbor $u \in U$ of v, $\mathbf{P}_{u\sim v}\big(|\pi^{\{u,v\}}(S)| < |S|\big) \leq \kappa$.*

We have adapted the original definition of Smooth Label Cover somewhat, choosing bipartite projection games over hypergraphs and characterizing Definition 5 as the essential property of smoothness. Our definition carries more similarity to smoothness as used in Håstad's simplified proof – attributed to Khot – of the approximation resistance of Max 3SAT [7]. In particular, the Label Cover variant our hardness result reduces from is the following. The theorem follows from standard arguments and is proved in the full version of this paper.

Theorem 1. *For any parameters $\varepsilon > 0, \kappa > 0, J \in \mathbb{N}$, there exists $k = k(\varepsilon)$ such that Gap-$(1,\varepsilon)$* Label Cover$_k$ *with the following properties is $\mathcal{NP}$-hard.*

- *The constraint graph G is left and right regular.*
- *Instances are (J,κ)-smooth.*
- *For some integer $u = u(\varepsilon)$ and $v = v(J,\kappa)$, the cardinalities of K and L are $k \stackrel{\text{def}}{=} 2^u 10^{u(v-1)} v$ and 10^{uv}, respectively.*
- *Projections are $d(\varepsilon) = 5^{u(\varepsilon)}$-regular.*

4 Main Theorem

The main theorem of the paper is the following.

Theorem 2. *The arity-4 predicate* "0, 1, or 3" *with negation is approximation resistant for satisfiable instances. Put differently, for every* $\epsilon > 0$*, it is* $\mathcal{NP}$*-hard to distinguish whether a* Max-CSP$^{\pm}$("0, 1, or 3") *instance has value 1 or value at most* $|P|/2^{-4} + \epsilon = 9/16 + \epsilon$.

Our proof defines a distribution on the predicate "0, 1, or 3" and shows that every non-constant term in the Fourier expansion of *any* predicate P must be small in the negative case. This proof in fact establishes that the predicate is hereditarily approximation resistant for satisfiable instances and, by symmetry, any predicate which strictly contains even or odd parity on four bits.

4.1 The Protocol

The hardness of Max-CSP$^{\pm}$(P) follows by a reduction from Smooth Label Cover as it appears in Theorem 1 with soundness $\varepsilon = \varepsilon(\epsilon)$, and label sets $K = [k(\varepsilon, J, \kappa)]$ and $L = K \times [d(\varepsilon)]$.

To define the reduction R from an instance $\mathcal{I}$, take as variables for the CSP$^{\pm}$(P) instance R($\mathcal{I}$) for every vertex $u \in U$, $2^{2^{|K|}}$ Boolean variables and for every vertex $v \in V$, $2^{2^{|L|}}$ variables. As is standard, we see these variables as functions $f^u : \{0,1\}^K \to \{0,1\}$ and $g^v : \{0,1\}^L \to \{0,1\}$. Let $\mathcal{D}$ be the uniform distribution on "1 or 3" and let $\mathcal{E}$ be the distribution which chooses u.a.r. from $\{0000, 0111\}$ with probability 0.5 and otherwise from $\{1000, 1110, 1101, 1011\}$. Define constraints corresponding to the following probabilistic verifier.

1. Pick a random vertex $u \in U$ and a random neighbor $v \in V$. Sample $\pi = \pi^{\{u,v\}}$ as defined by the Smooth Label Cover instance and let $\bar{\pi}$ be an arbitrary bijection $L \leftrightarrow L$ such that for every $i, i' \in K$ and $r \in [d]$, $\pi(i, r) = i'$ iff $\exists_{r' \in [d]} \bar{\pi}(i, r) = (i', r')$.
2. Sample random folding constants $a, b \sim \{0,1\}$. Define $f_a(\mathbf{x}) = a \oplus f^u(a \oplus \mathbf{x})$ and $g_b(\mathbf{y}) = b \oplus g^v(b \oplus \mathbf{y} \circ \bar{\pi})$.
3. For each $i \in K$, independently choose x_i u.a.r. from $\{0,1\}$. For each $j \in L$, independently sample $(x_{\pi(j)}, y_j^{(2)}, y_j^{(3)}, y_j^{(4)})$ conditioned on $x_{\pi(j)}$ from $\mathcal{D}$ with probability δ and otherwise $\mathcal{E}$.
4. Accept iff $\left(f_a(\mathbf{x}), g_b(\mathbf{y}^{(2)}), g_b(\mathbf{y}^{(3)}), g_b(\mathbf{y}^{(4)})\right) \in \mathrm{P}$.

We note that queries $a \oplus f(a \oplus \cdot)$ are permitted in Max-CSP$^{\pm}$ where the operation $a \oplus \cdot$ act as a possible negation of a variable. This construct is called *folding* and ensures that $\mathbf{E}_{\mathbf{x}}[f] = \mathbf{E}_{\mathbf{y}}[g] = 0$.

The goal is to show the following two properties of the protocol from which Theorem 2 follows. Completeness follows by inspection and is standard. We elaborate on the more interesting soundness bound in the following section. For constants and complete arguments, we refer to the full version.

Proposition 1. *The protocol has completeness 1. Said equivalently, if* $\mathrm{Val}(I) = 1$*, then* $\mathrm{Val}(\mathrm{R_P}(I)) = 1$*.*

Proposition 2. *The protocol has soundness* $|\mathrm{P}|/16 + \epsilon = 9/16 + \epsilon$*. More specifically, if* $\mathrm{Val}(I) \leq \varepsilon = \varepsilon(\epsilon)$*, then* $\mathrm{Val}(\mathrm{R_P}(I)) \leq 9/16 + \epsilon$ *where* $\epsilon(\varepsilon) \to 0$ *as* $\varepsilon \to 0$*.*

4.2 Soundness

As is usual, we establish the soundness through the contradiction of its contrapositive: supposing that the acceptance probability of $\mathrm{R_P}(I)$ is greater than $|\mathrm{P}|/16 + \epsilon$, we show that there is a labeling of the Smooth Label Cover instance I achieving value greater than $\varepsilon = \varepsilon(\epsilon)$. The dependency in particular is $\varepsilon(\epsilon) = 2^{-8}\bar{\eta}\bar{\gamma}^3\epsilon^2$ where the noise constants η and γ appear below.

Denote by $\mathcal{T}_0$ the distribution $\delta\mathcal{D} + \bar{\delta}\mathcal{E}$, by $\mathcal{T}_0'$ the distribution $\mathcal{T}_0^{d\text{-proj-1}^{\otimes K}}$, and by $\mathcal{T}_1'$ the distribution $\left(\mathrm{T}_\eta^1\mathrm{T}_\gamma^2\mathrm{T}_\gamma^3\mathrm{T}_\gamma^4\mathcal{T}_0\right)^{d\text{-proj-1}^{\otimes K}}$. The test distribution of the protocol is $\mathcal{T}_0'$ and we wish to argue that it behaves as $\mathcal{T}_1'$. For notational simplicity, define $f = \mathbf{E}_a[f_a]$ and $g = \mathbf{E}_b[g_b]$. Let $q_1 = f(\mathbf{x}), q_2 = g(\mathbf{y}^{(2)}), \ldots, q_4 = g(\mathbf{y}^{(4)})$. As is usual for PCP analysis, we substitute 1 for -1 and 0 for 1.

Considering the Fourier transform $\{\hat{P}_\Gamma\}_{\Gamma\subseteq[4]}$ of the predicate, the acceptance probability of the protocol equals $\mathbf{E}_{E,\mathcal{T}_0'}\left[\sum_{\Gamma\subseteq[4]}\hat{\mathrm{P}}_\Gamma\chi_\Gamma(\mathbf{q})\right]$ where the distribution is over a random edge $e \in E$ from the Smooth Label Cover instance and the arguments from $\mathcal{T}_0'$. For an arbitrary $\Gamma \neq \emptyset$ and distribution $\mathcal{R}$, let us denote by $\psi_\Gamma(\mathcal{R}) = \mathbf{E}_{E,\mathcal{R}}[\chi_\Gamma(\mathbf{q})]$. Conceptually, we refer to these terms as $\mathbf{E}[\prod g]$ or $\mathbf{E}[f\prod g]$ for zero or more functions g. We also note that the acceptance probability in the new notation equals $\sum_\Gamma \psi_\Gamma(\mathcal{T}_0')$. The term with $\Gamma = \emptyset$ corresponds to the constant $|P|/2^{-4}$ and we wish to bound all other terms. This is handled by the following five propositions from which the soundness follows. Their respective proofs can be found in the full version of the paper. We note that the first two steps establish basic properties while the remaining three mimic the approach of O'Donnell and Wu [19].

The first proposition establishes basic properties of the distributions and follows from inspection of the distributions. If we did not have a predicate on at least four bits, we would not be able to in the general case define bounded-correlation distributions with this property.

Proposition 3. *Consider either distribution* $\mathcal{T}_0'$ *or* $\mathcal{T}_1'$*. The marginals of* $\mathbf{x}$ *and* $\mathbf{y}^{(t)}$ *are uniform and additionally* $\mathbf{y}^{(t)}$ *is independent of* $\mathbf{x}$ *for* $t = 2, 3, 4$*.*

The second proposition states that, due to the preceding independence, terms involving at most one $\mathbf{y}^{(t)}$ argument are zero.

Proposition 4. $\psi_\Gamma(\mathcal{T}_0') = 0$ *for* $\emptyset \neq \Gamma \subseteq [4], |\Gamma \cap \{2, 3, 4\}| \leq 1$*.*

The third lemma, which involves smoothness and significant technical work, argues that terms are in expectation roughly the same with the original test

distribution as the test distribution with noise, independent of d, on all arguments. We argue this in three steps. First, correlation bounds and a noise introduction lemma by Mossel [14] are used to introduce projected noise for $\mathbf{y}^{(t)}$ arguments, i.e. noise where all coordinates sharing projection are jointly resampled. Next, with projected noise, the string $\mathbf{x}$ has bounded correlation to $\{\mathbf{y}^{(t)}\}_{t=2}^{4}$ and permit by the same lemma introduction of noise for $\mathbf{x}$. Finally, we argue that if an argument $\mathbf{y}^{(t)}$ has projected noise γ for smoothness (J, κ), then the expectation changes by at most $2\sqrt{\kappa} + 2\gamma^J$ going to independent noise for $\mathbf{y}^{(t)}$. This latter step is probably the most interesting and involves analyzing Fourier expansions, or Efron-Stein decompositions. Under smoothness, for Fourier coefficients of cardinality at most J, with high probability, projections are unique and projected noise behave the same as independent noise; for Fourier coefficients of greater cardinality, with high probability, the set projects to a large number of different labels and makes an expectation small due to either noise.

We note that the constants ρ_0 and ρ_1 appearing in the proposition are correlation bounds appearing in the proofs and are bounded away from 1 depending only on δ and γ.

Proposition 5. $|\psi_\Gamma(\mathcal{T}'_0) - \psi_\Gamma(\mathcal{T}'_1)| \leq \sup_{k\geq 0} \rho_0^k(1-\gamma^k) + \sup_{k\geq 0} \rho_1^k(1-\eta^k) + 2\sqrt{\kappa} + 6\gamma^J \leq \epsilon/256$ *for any* $\Gamma \subseteq [4]$.

Fourth, over the noised distribution, terms of the form $\mathbf{E}[\prod g]$ are shown to have an expectation which approaches 0 as parameters are tweaked. The hardest case is $\mathbf{E}\left[g(\mathbf{y}^{(2)})g(\mathbf{y}^{(3)})g(\mathbf{y}^{(4)})\right]$, i.e. when all three g arguments are involved. We provide a brief sketch of the argument here but refer to the full version for definitions due to space limitations. The proof begins by arguing that because of noise and partial independence, $\mathbf{E}[ggg] \approx \mathbf{E}\left[g^{\leq k}g^{\leq k}g\right]$ where $g^{\leq k}$ is a "low-degree expansion" of g and $k = \lfloor J/2 \rfloor$. Because of uniform marginals, this implies $\mathbf{E}[ggg] \approx \mathbf{E}\left[g^{\leq k}g^{\leq k}g^{\leq 2k}\right]$. Second, one can argue that removing terms from the Fourier expansion of these functions which do not have "unique" projections will not change the expectation by too much. This step involves smoothness and going from higher ℓ_p norms of sums of Fourier terms to ℓ_2 via hypercontractivity and using that the functions are of low degree. For "unique" projections, the expectation of the expression is identical to that of the distribution which does not sample conditioned on the argument $\mathbf{x}$. Still, the expectation of a product of low-degree functions is not easily bounded and so we take all steps in reverse for the non-conditional distribution. This returns us to the expression $\mathbf{E}[ggg]$ but under a distribution where the expectation is at most $\sqrt{\delta}$ due to correlation bounds.

Proposition 6. $|\psi_\Gamma(\mathcal{T}'_1)| \leq 4\gamma^{J/2-1} + 6 \cdot 3^{3J/4}\sqrt{\kappa} + \sqrt{\delta} \leq \epsilon/256$ *for* $1 \notin \Gamma \subseteq [4], |\Gamma| \geq 2$.

Finally, we bound terms of the form $\mathbf{E}[f \prod g]$. This is often considered the hardest part of PCP analysis. However, the argument is almost immediate after we extend Mossel's multivariate invariance principle [14] to projection games.

Proposition 7. $|\psi_\Gamma(\mathcal{T}_1')| \leq 2\sqrt{\bar{\gamma}^{-1}\,\mathbf{E}_E\left[\sum_{(i,j)\in\pi}\mathrm{Inf}_i^{(\eta)}(f)\mathrm{Inf}_j^{(\gamma)}(g)\right]}$ *for* $1 \in \Gamma \subseteq [4], |\Gamma| \geq 3$.

As mentioned previously, the acceptance probability of the protocol equals $\sum_\Gamma \psi_\Gamma(\mathcal{T}_0')$. Each term, besides the constant term, is bounded by the preceding propositions. Proposition 4 and Proposition 6 bound terms by arbitrarily small constants while Proposition 7 bounds the expectation of a $\mathbf{E}[f\prod g]$ term by the sum of noisy "cross-influences" of coordinated coordinates. This quantity is easy to relate to the value of the (η,γ)-*Noisy Influence Assignment* which independently sets vertex u resp. v to label i resp. j with probability proportional to $\mathrm{Inf}_i^{(\eta)}(f^u)$ resp. $\mathrm{Inf}_j^{(\gamma)}(g^v)$. To conclude the proof, one argues that if these terms are not negligible, then the noisy influence assignment contradicts the assumption that the reduced-from Label Cover instance has a low value.

5 Discussion

We note that our result should generalize to any predicate of width at least four strictly containing parity, exploiting the roughly $m/2$-wise independent distribution conditioned on the first variable which one can define on the generalized predicate "odd or zero" with strictly positive weight on the all-zeroes outcome. As all our proofs are based on Efron-Stein decompositions, which have similar properties also for larger domains, one may venture that similar results hold also for predicates $P \subseteq \mathbb{F}_q$ strictly containing linear equations. Considering generalizations of our proofs, it also appears feasible that supporting roughly $2m/3$-wise independence suffices for the unconditional approximation resistance of a predicate. This is an interesting question as the present gap is essentially the greatest non-trivial possible: $(m-1)$-wise independence suffices under $\mathcal{P} \neq \mathcal{NP}$ [9], while two-wise independence suffices under UGC [2].

However, we face technical difficulties addressing the width-three case, even for Boolean variables. The main technical limitation of this extension is that we are unable to simultaneously make the coordinates independent in the argument $\mathbf{y}^{(2)}$ for a function on the larger label set, and $\mathbf{y}^{(2)}$ independent of the argument $\mathbf{x}$ for a function on the smaller label set. These two properties are used in all of the major steps of our result. Curiously, Håstad has shown in parallel a result [8] similar to ours for the case $m = 3$. The methods of the two papers have similar foundations in that they rely on Smooth Label Cover to reduce the effect of projection degrees. They differ in that ours seek general techniques to eliminate this dependency while Håstad uses more direct methods and counteracts the dependency with massive smoothness constants.

It is interesting to consider whether the techniques could be used to circumvent the d-to-1 Conjecture for other results. Indeed, the sole reason d-to-1 Games rather than Unique Games is used as a starting point for reductions, is because Unique Games is not hard for satisfiable instances. As mentioned previously, the invariance argument of O'Donnell and Wu has appeared in other works. O'Donnell and Wu initially posed a three-bit dictatorship test [18] before

adapting it to a PCP assuming the d-to-1 Conjecture. It has been suggested [4] that other query-efficient dictatorship tests [3,20] may yield PCP's by similar methods; perhaps unconditionally by our methods.

Acknowledgement. The author would like to thank Johan Håstad for his invaluable advice, curious discussions, and intuitive explanations; Sangxia Huang for discussions and calling previous work to attention; and an anonymous referee for helpful comments.

References

1. Austrin, P., Håstad, J.: Randomly Supported Independence and Resistance. In: ACM Symp. on the Theory of Comp. (STOC), vol. 41 (2009)
2. Austrin, P., Mossel, E.: Approximation Resistant Predicates from Pairwise Independence. In: IEEE Conf. on Comp. Complexity (CCC), vol. 23 (2008)
3. Chen, V.: A Hypergraph Dictatorship Test with Perfect Completeness. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 448–461. Springer, Heidelberg (2009)
4. Chen, V.: Property Testing. Springer (2010)
5. Feldman, V., Guruswami, V., Raghavendra, P., Yi, W.: Agnostic Learning of Monomials by Halfspaces is Hard. In: (FOCS) IEEE Found. of Comp. Sc., vol. 50 (2009)
6. Guruswami, V., Raghavendra, P., Saket, R.: Bypassing UGC from Some Optimal Geometric Inapproximability Results. In: ACM-SIAM Symp. on Discrete Alg. (SODA), vol. 23 (2012)
7. Håstad, J.: On Linear Equations and Satisfiability (2011) (unpublished material)
8. Håstad, J.: On the NP-Hardness of Max-Not-2. In: Gupta, A., et al. (eds.) APPROX/RANDOM 2012. LNCS, vol. 7408, pp. 170–181. Springer, Heidelberg (2012)
9. Håstad, J.: Some Optimal Inapproximability Results. J. of ACM 48 (2001)
10. Huang, S.: Approximation Resistance on Satisfiable Instances for Predicates Strictly Dominating Parity. Elect. C. on Comp. Complexity (ECCC) (2012)
11. Khot, S.: Hardness Results for Coloring 3-Colorable 3-Uniform Hypergraphs. In: IEEE Foundations of Comp. Sc. (FOCS), vol. 43 (2002)
12. Khot, S.: On the Power of Unique 2-Prover 1-Round Games. In: ACM Symp. on the Theory of Comp. (STOC), vol. 34 (2002)
13. Khot, S., Saket, R.: A 3-query Non-Adaptive PCP with Perfect Completeness. In: Conf. on Comp. Complexity (CCC), vol. 21 (2006)
14. Mossel, E.: Gaussian Bounds for Noise Correlation of Functions. In: Geometric and Functional Analysis. Birkhauser, Basel (2010)
15. Mossel, E.: Gaussian Bounds for Noise Correlation of Functions and Tight Analysis of Long Codes. In: IEEE Found. of Comp. Sc. (FOCS), vol. 49 (2008)
16. Mossel, E., O'Donnell, R., Oleszkiewicz, K.: Noise stability of Functions with Low Influences: Invariance and Optimality. In: IEEE Foundations of Comp. Sc. (FOCS), vol. 46 (2005)
17. O'Donnell, R., Wright, J.: A New Point of NP-hardness for Unique Games. In: ACM Symp. on the Theory of Comp. (STOC), vol. 44 (2012)
18. O'Donnell, R., Yi, W.: 3-Bit Dictator Testing: 1 vs. 5/8. In: ACM-SIAM Symp. on Discrete Alg. (SODA), vol. 20 (2009)

19. O'Donnell, R., Wu, Y.: Conditional Hardness for Satisfiable 3-CSPs. In: ACM Symp. on the Theory of Comp. (STOC), vol. 41 (2009)
20. Tamaki, S., Yoshida, Y.: A Query Efficient Non-Adaptive Long Code Test with Perfect Completeness. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX and RANDOM 2010, LNCS, vol. 6302, pp. 738–751. Springer, Heidelberg (2010)
21. Tang, L.: Conditional Hardness of Approximating Satisfiable Max 3CSP-q. In: Dong, Y., Du, D.-Z., Ibarra, O. (eds.) ISAAC 2009. LNCS, vol. 5878, pp. 923–932. Springer, Heidelberg (2009)
22. Wenner, C.: Noise Introduction and Multivariate Invariance for Projection Games (2012) (unpublished manuscript)

Spectral Norm of Symmetric Functions*

Anil Ada, Omar Fawzi, and Hamed Hatami

School of Computer Science, McGill University
{aada,ofawzi,hatami}@cs.mcgill.ca

Abstract. The spectral norm of a Boolean function $f:\{0,1\}^n \to \{-1,1\}$ is the sum of the absolute values of its Fourier coefficients. This quantity provides useful upper and lower bounds on the complexity of a function in areas such as learning theory, circuit complexity, and communication complexity. In this paper, we give a combinatorial characterization for the spectral norm of symmetric functions. We show that the logarithm of the spectral norm is of the same order of magnitude as $r(f)\log(n/r(f))$ where $r(f)=\max\{r_0,r_1\}$, and r_0 and r_1 are the smallest integers less than $n/2$ such that $f(x)$ or $f(x)\cdot \text{PARITY}(x)$ is constant for all x with $\sum x_i \in [r_0, n-r_1]$. We mention some applications to the decision tree and communication complexity of symmetric functions.

1 Introduction

The study of Boolean functions $f:\{0,1\}^n \to \{-1,1\}$ is central to complexity theory and combinatorics as objects of interest in these areas can often be represented as Boolean functions. Fourier analysis of Boolean functions provides some of the strongest tools in this study with applications to graph theory, circuit complexity, communication complexity, hardness of approximation, machine learning, etc.

In many different settings, Boolean functions with "smeared out" Fourier spectrums have higher "complexity". There are various useful ways to measure the spreadness of the spectrum. Some notable ones are the spectral norm $\|\widehat{f}\|_1 = \sum_S |\widehat{f}(S)|$ (i.e., the ℓ_1 norm), the ℓ_∞ norm $\|\widehat{f}\|_\infty = \max_S |\widehat{f}(S)|$, and the Shannon entropy of the squares of the Fourier coefficients $H[\widehat{f}^2] = -\sum_S \widehat{f}(S)^2 \log \widehat{f}(S)^2$. The focus of this paper is on the spectral norm.

Spectral Norm of Boolean Functions. As $\sum_S \widehat{f}(S)^2 = 1$ for a Boolean function f, it is often useful to view the squares of the Fourier coefficients as a probability distribution over the subsets $S \subseteq [n]$. The spectral norm corresponds to the Rényi entropy of order $1/2$ of the squares of the Fourier coefficients, $H_{1/2}[\widehat{f}^2] = 2\log\left(\sum_S |\widehat{f}(S)|\right) = 2\log\|\widehat{f}\|_1$. It provides useful upper and lower bounds on the *complexity* of a function in settings such as learning theory, circuit complexity, and communication complexity. It is particularly useful in the settings where PARITY is considered a function of low complexity. We list some of the applications below.

In the setting of learning theory, the spectral norm is used in conjunction with the *Kushilevitz-Mansour Algorithm* [12]. This algorithm, using membership queries, learns

* A full version can be found online [1].

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 338–349, 2012.
© Springer-Verlag Berlin Heidelberg 2012

efficiently a concept class $\mathcal{C}$ where the Fourier spectrum of every function in $\mathcal{C}$ is concentrated on a small set of characters (This set can be different for different functions.). Kushilevitz and Mansour observe that an upper bound on the spectral norm implies such a concentration, and obtain:

> If $\mathcal{C} = \{f : \{0,1\}^n \to \{-1,1\} \mid \|\widehat{f}\|_1 \leq s\}$, then $\mathcal{C}$ is learnable with membership queries in time $\text{poly}(n, s, 1/\varepsilon)$.

Using the above result, they show that functions computable by small size parity decision trees[1] are efficiently learnable with membership queries. This is done by observing that a function computable by a size s parity decision tree satisfies $\|\widehat{f}\|_1 \leq s$. This inequality is also interesting since it provides a lower bound in terms of the spectral norm on the size of any parity decision tree computing f.

Threshold circuits (i.e., circuits composed of threshold gates) constitute an important model of computation (in part due to their resemblance to neural networks), and they have been studied extensively. A classical result of Bruck and Smolensky [3] states that a function with small spectral norm can be represented as the sign of a polynomial with few monomials. This in turn implies that functions with small spectral norm can be computed by depth 2 threshold circuits of small size. The result of Bruck and Smolensky has found other interesting applications (see for example [22, 5, 8, 14]).

We now turn our attention to communication complexity. Arguably the most famous conjecture in communication complexity is the Log Rank Conjecture which states that the deterministic communication complexity of a function $F : \{0,1\}^n \times \{0,1\}^n \to \{-1,1\}$ is upper bounded by $\log^c \operatorname{rank} M_F$ where the matrix M_F is defined as $M_F[x,y] = F(x,y)$. Grolmusz [7] makes a similar intriguing conjecture for the randomized communication complexity:

> There is a constant c such that the public coin randomized communication complexity of $F : \{0,1\}^n \times \{0,1\}^n \to \{-1,1\}$ is upper bounded by $\log^c \|\widehat{F}\|_1$.

In the same paper, Grolmusz is able to prove a much weaker upper bound of $O(\|\widehat{F}\|_1^2 \delta(n))$ with $\exp(-c\delta(n))$ probability of error. Even this weaker result has interesting applications in circuit complexity and decision tree complexity (see [7] for more details).

Another major open problem in communication complexity is whether the classical and quantum communication complexity of total Boolean functions $f : X \times Y \to \{-1,1\}$ (i.e., functions defined on all of $X \times Y$) are polynomially related. It is conjectured that this is so and research has been focused on establishing it for natural large families of functions. In an important paper [17] Razborov showed that the conjecture is true for functions of the form $F(x,y) = \text{SYM}(x \wedge y)$ where SYM denotes a symmetric function, and $x \wedge y$ is the bitwise AND of x and y. Shi and Zhang [20] verified the conjecture for $F(x,y) = \text{SYM}(x \oplus y)$ where $x \oplus y$ denotes the bitwise XOR. The next big targets are $F(x,y) = f(x \wedge y)$ and $F(x,y) = f(x \oplus y)$ for general f, but handling arbitrary f seems difficult at the moment.

[1] Parity decision trees generalize the usual decision tree model: in every node we branch according to the parity of a subset of the variables.

A variant of the spectral norm, *the approximate spectral norm*, is intimately related to the communication complexity of "xor functions". The ε-approximate spectral norm of f, denoted $\|\widehat{f}\|_{1,\varepsilon}$, is the smallest spectral norm of a function $g : \{0,1\}^n \to \mathbb{R}$ such that $\|f-g\|_\infty \leq \varepsilon$. It is known (see for example [13]) that $\log \|\widehat{f}\|_{1,\varepsilon}$ lower bounds the quantum bounded error communication complexity of $f(x \oplus y)$. We expect that the lower bound $\log \|\widehat{f}\|_{1,\varepsilon}$ is tight, and that this quantity characterizes the communication complexity of xor functions. More discussion on the communication complexity of xor functions, and how it relates to this work is given in Section 4.

This ends our discussion of the use of the spectral norm in learning theory, circuit complexity and communication complexity. We conclude this subsection by mentioning a relatively recent result that studies the spectral norm of Boolean functions. Green and Sanders [6] show that every Boolean function whose spectral norm is bounded by a constant can be written as a sum of constantly many $\pm$ indicators of cosets. This gives an interesting characterization of Boolean functions with small spectral norm.

Fourier Spectrum of Symmetric Functions. A function $f : \{0,1\}^n \to \{-1,1\}$ is called *symmetric* if it is invariant under permutations of the coordinates. In other words the value of $f(x)$ depends only on $\sum x_i$ (i.e., $f(x) = f(y)$ whenever $\sum_i x_i = \sum_i y_i$). Symmetric functions are at the heart of complexity theory as natural functions like AND, OR, MAJORITY, and MOD_m are all symmetric. They are often the starting point of investigation because the symmetry of the function can be exploited. On the other hand, they can also have surprising power. In several settings, functions such as PARITY and MAJORITY represent "hard" functions. Given their central role, it is of interest to gain insight into the Fourier spectrum of symmetric functions.

There are various nice results related to the Fourier spectrum of symmetric functions. We cite a few of them here. A beautiful result of Paturi [16] tightly characterizes the approximate degree of every symmetric function, and this has found many applications in theoretical computer science [17,2,18,4,19]. Kolountzakis *et al.* [11] studied the so called *minimal degree* of symmetric functions and applied their result in learning theory. Shpilka and Tal [21] later simplified and improved the work of Kolountzakis *et al.* Recently, O'Donnell, Wright and Zhou [15] verified an important conjecture in the analysis of Boolean functions, the Fourier Entropy/Influence Conjecture, in the setting of symmetric functions. In fact we make use of their key lemma in this paper.

1.1 Our Results and Proof Overview

We give a combinatorial characterization of the spectral norm of symmetric functions. For $x \in \{0,1\}^n$, define $|x| \stackrel{\text{def}}{=} \sum x_i$. For a function $f : \{0,1\}^n \to \{-1,1\}$, let r_0 and r_1 be the minimum integers less than $n/2$ such that $f(x)$ or $f(x) \cdot \text{PARITY}(x)$ is constant for x with $|x| \in [r_0, n-r_1]$. Define $r(f) \stackrel{\text{def}}{=} \max\{r_0, r_1\}$. We show that $\log \|\widehat{f}\|_1$ is of the same order of magnitude as $r(f)\log(n/r(f))$:

Theorem 1 (Main Theorem). *For any symmetric function $f : \{0,1\}^n \to \{-1,1\}$, we have*

$$\log \|\widehat{f}\|_1 = \Theta\left(r(f)\log\left(\frac{n}{r(f)}\right)\right)$$

whenever $r(f) > 1$. *If* $r(f) \leq 1$, *then* $\|\widehat{f}\|_1 = \Theta(1)$.

As an application, we give a characterization of the parity decision tree size of symmetric functions. As mentioned in Section 1, a parity decision tree computes a boolean function by querying the parities of subsets of the variables. The size of the tree is simply the number of leaves in the tree.

Corollary 1. *Let* $f : \{0,1\}^n \to \{-1,1\}$ *be a symmetric function. Then the parity decision tree size of* f *is* $2^{\Theta(r(f)\log(n/r(f)))}$.

We present the proof of this corollary in the full version. Note that the lower bound also applies in the case of the usual decision tree size (where one is restricted to query only variables). Decision tree size is an important measure in learning theory; algorithms for learning decision trees efficiently is of great interest both for practical and theoretical reasons. One of the most well-known and studied problems is whether small size decision trees are efficiently learnable from uniformly random examples.

As a second application, using the protocol of Shi and Zhang [20, Proposition 3.4], and the observation that $\|\widehat{F}\|_1 = \|\widehat{f}\|_1$ when $F(x,y) = f(x \oplus y)$, we verify Grolmusz's conjecture mentioned earlier in Section 1 in the setting of *symmetric xor functions*.

Corollary 2. *Let* $f : \{0,1\}^n \to \{-1,1\}$ *be a symmetric function and let* $F : \{0,1\}^n \times \{0,1\}^n \to \{-1,1\}$ *be defined as* $F(x,y) = f(x \oplus y)$. *Then the public coin constant error randomized communication complexity of* F *is upper bounded by* $O(\log^2 \|\widehat{F}\|_1)$.

We now give an outline for the proof of Theorem 1. The upper bound is quite straightforward and is given in Lemma 2. The lower bound is handled in two different cases: when $r(f)$ is bounded away from $n/2$ (Lemma 4) and when $r(f)$ is close to $n/2$ (Lemma 6).

We refer to the Fourier spectrum of f restricted to the sets $S \subseteq [n]$ of size k as the *k-th level* of the Fourier spectrum. Note that for a symmetric f, we have $\widehat{f}(S) = \widehat{f}(T)$ whenever $|S| = |T|$. Therefore the Fourier spectrum is maximally spread out in each level. The overall strategy for the lower bound is to show an appropriate lower bound on the ℓ_2 mass of the Fourier spectrum on a middle level. Middle levels have many Fourier coefficients, and therefore contribute significantly to the spectral norm provided there is enough ℓ_2 mass on them. An important tool in our analysis is the use of certain discrete derivatives of f. Indentify $\{0,1\}^n$ with $\mathbb{F}_2^n$ and let $e_1, \ldots, e_n$ denote the standard vectors in $\mathbb{F}_2^n$. For $i \neq j$, define $f_{ij}(x) \stackrel{\text{def}}{=} f(x + e_i + e_j) - f(x)$. We observe that

$$\sum_{i \neq j} \mathbf{E}\left[f_{ij}^2\right] = 8 \sum_S |S|(n - |S|)\widehat{f}(S)^2.$$

The quantity on the LHS, and therefore the RHS, can be lower bounded using $r(f)$ (Lemma 3). As the coefficient $|S|(n-|S|)$ increases as $|S|$ approaches $n/2$, we are able to give a lower bound on the ℓ_2 mass of the Fourier spectrum on the middle levels. This approach gives tight bounds for $r(f)$ bounded away from $n/2$, but not for a function such as MAJORITY.

To handle functions f with $r(f)$ close to $n/2$, we use ideas from [15]. The main lemma of [15] states that the first derivatives of a symmetric function are noise sensitive. We observe that this is also true for the derivatives f_{ij}. This allows us to derive the inequality

$$\sum_S |S|(n-|S|)\widehat{f}(S)^2(\rho^{|S|}+\rho^{n-|S|}) \leq \frac{8}{\sqrt{\pi c}} \cdot \sum_S |S|(n-|S|)\widehat{f}(S)^2,$$

where $\rho = (1-c/n)$. The quantity $\rho^{|S|}+\rho^{n-|S|}$ is decreasing in $|S|$ for $|S| \leq n/2$. Thinking of c as a large constant, we see that the dampening of the middle levels with $\rho^{|S|}+\rho^{n-|S|}$ decreases the value of the sum significantly. From this, we can lower bound the ℓ_2 mass of the middle levels. Note that if $\sum_S |S|(n-|S|)\widehat{f}(S)^2$ is small to begin with ($r(f)$ is small), the above inequality is not useful. On the other hand if $r(f)$ is large, $\sum_S |S|(n-|S|)\widehat{f}(S)^2$ is large, and the strategy just described gives good bounds.

2 Preliminaries

We view Boolean functions $f : \{0,1\}^n \to \{-1,1\}$ as residing in the vector space $\{f : \{0,1\}^n \to \mathbb{C}\}$. If we view the domain as the group $\mathbb{F}_2^n$, we can appeal to Fourier analysis, and express every $f : \{0,1\}^n \to \mathbb{C}$ (uniquely) as a linear combination of the characters of $\mathbb{F}_2^n$. That is every function $f : \mathbb{F}_2^n \to \mathbb{C}$ can be written as $f = \sum_{S\subseteq[n]} \widehat{f}(S)\chi_S$, where the characters χ_S are defined as $\chi_S : x \mapsto (-1)^{\sum_{i\in S} x_i}$, and $\widehat{f}(S) \in \mathbb{C}$ are their corresponding Fourier coefficients. Since the characters form an orthonormal basis for $\{f : \{0,1\}^n \to \mathbb{C}\}$, we have $\widehat{f}(S) = \langle f, \chi_S \rangle = \mathbf{E}_x[f(x)\chi_S(x)]$.

For a Boolean function f, we define $W_k[f] = \sum_{|S|=k} |\widehat{f}(S)|^2$. We simply use W_k when f is clear from the context. For a symmetric function, we often write $f(k)$ for $f(x)$ with $\sum_i x_i = k$ and $k \in [n]$. We use h to denote the binary entropy function $h(\alpha) = -\alpha\log(\alpha) - (1-\alpha)\log(1-\alpha)$.

Definition 1. *For any $f : \{0,1\}^n \to \mathbb{R}$, we define $R(f) \stackrel{def}{=} \sum_{S\subseteq[n]} |S|(n-|S|)\widehat{f}(S)^2$.*

For $a \in \mathbb{F}_2^n$, we define the derivative of $f : \mathbb{F}_2^n \to \mathbb{R}$ in the direction a as $\Delta_a f : x \mapsto f(x+a) - f(x)$. Let $e_1, \ldots, e_n$ denote the standard vectors in $\mathbb{F}_2^n$, and let $f : \{0,1\}^n \to \mathbb{R}$. For all $i \neq j$, define

$$f_{ij} \stackrel{\text{def}}{=} \Delta_{e_i+e_j} f. \tag{1}$$

Lemma 1. *For every $f : \{0,1\}^n \to \mathbb{R}$, we have $\sum_{i\neq j} \mathbf{E}\left[f_{ij}^2\right] = 8R(f)$.*

For a proof, we refer the reader to the full version.

3 Proof of Theorem 1

As mentioned earlier the upper bound is proved in Lemma 2. The proof of the lower bound is divided into two parts: Lemma 4 handles the case where r is bounded away from $n/2$ and Lemma 6 the case when r is close to $n/2$.

3.1 Upper Bound

Lemma 2. *For all $n \geq 1$ and every symmetric function $f : \{0,1\}^n \to \{-1,1\}$,*

$$\log \|\widehat{f}\|_1 \leq 2 \cdot r(f) \log(n/r(f)) + 3.$$

The proof can be found in the full version.

3.2 Lower Bound

We start by proving some simple observations.

Lemma 3. *Let $f : \{0,1\}^n \to \{-1,1\}$ be a symmetric function, and define $r_0 = r_0(f)$ and $r_1 = r_1(f)$. Then*

$$R(f) \geq \left((n-r_0+1)(n-r_0)\binom{n}{r_0-1} + (n-r_1+1)(n-r_1)\binom{n}{r_1-1} \right) 2^{-n}. \quad (2)$$

Moreover, assuming that $f(s) = 1$ for all $s \in \{r_0, \ldots, n-r_1\}$, we have

$$\sum_{S \neq \emptyset} \widehat{f}(S)^2 \leq 4 \left(\sum_{s<r_0} \binom{n}{s} + \sum_{s<r_1} \binom{n}{s} \right) 2^{-n}. \quad (3)$$

Lower Bound: $r \ll n/2$.

Lemma 4. *For every symmetric function $f : \{0,1\}^n \to \{-1,1\}$ with $r = r(f)$,*

$$\log \|\widehat{f}\|_1 \geq \Omega\left(\left(1 - \frac{2r-2}{n} \right) \cdot r \log(n/r) \right).$$

Proof. Observe that we can assume without loss of generality that $f(s) = 1$ for all $s \in \{r_0, \ldots, n-r_1\}$. In fact, to handle the case $f = -1$ or $f = \pm\text{PARITY}$ in $[r_0, n-r_1]$, it suffices to multiply the function by -1 or by $\pm\text{PARITY}$, respectively. This does not affect the spectral norm of the function.

We prove the statement by showing that a significant portion of the ℓ_2 mass of $\widehat{f}$ sits in the middle levels from m to $n-m$ for a well-chosen m depending on $r(f)$.

Define $\alpha_0 = \frac{r_0-1}{n} < 1/2$ and $\alpha_1 = \frac{r_1-1}{n}$. We also let for $i \in \{0,1\}$, $m_i = \left\lfloor n/2 \cdot (1 - \sqrt{4\alpha_i - 6\alpha_i^2 + 4\alpha_i^3}) \right\rfloor$. By Lemma 3, we have $\sum_{k>0} W_k \leq 4 \cdot \left(\sum_{s<r_0} \binom{n}{s} + \sum_{s<r_1} \binom{n}{s} \right) 2^{-n}$. Let U_k and V_k be so that $W_k = U_k + V_k$ and $\sum_{k>0} U_k \leq 4 \cdot 2^{-n} \sum_{s<r_0} \binom{n}{s}$ and $\sum_{k>0} V_k \leq 4 \cdot 2^{-n} \sum_{s<r_1} \binom{n}{s}$. Recall that our strategy is to obtain a lower bound on the ℓ_2 mass of the Fourier transform on the middle levels. More precisely, our objective will be to derive a lower bound on $\sum_{k=m_0}^{n-m_0} k(n-k)U_k + \sum_{k=m_1}^{n-m_1} k(n-k)V_k$ using Lemma 3.

$$\begin{aligned}
&\sum_{k=m_0}^{n-m_0} k(n-k)U_k + \sum_{k=m_1}^{n-m_1} k(n-k)V_k \\
&\quad = R(f) - \sum_{k\notin[m_0,n-m_0]} k(n-k)U_k - \sum_{k\notin[m_1,n-m_1]} k(n-k)V_k \\
&\quad \geq (n-r_0)(n-r_0+1)\binom{n}{r_0-1}2^{-n} - (m_0-1)(n-m_0+1)4\cdot 2^{-n}\sum_{s<r_0}\binom{n}{s} \\
&\qquad + (n-r_1)(n-r_1+1)\binom{n}{r_1-1}2^{-n} - (m_1-1)(n-m_1+1)4\cdot 2^{-n}\sum_{s<r_1}\binom{n}{s}. \quad (4)
\end{aligned}$$

Define $A_0 \stackrel{\text{def}}{=} (n-r_0)(n-r_0+1)\binom{n}{r_0-1}2^{-n} - (m_0-1)(n-m_0+1)4\cdot 2^{-n}\sum_{s<r_0}\binom{n}{s}$, and let A_1 be its analogue for r_1 so that the right hand side of (4) equals A_0+A_1.

Observe that $\binom{n}{s} = \frac{s+1}{n-s}\binom{n}{s+1}$, and $\frac{s+1}{n-s} \leq \frac{r_0-1}{n-(r_0-1)} = \frac{\alpha_0}{1-\alpha_0}$ for $s < r_0-1$. Thus

$$\begin{aligned}
A_0 &\geq \binom{n}{r_0-1}2^{-n}\left((n-\alpha_0 n-1)(n-\alpha_0 n) - 4(m_0-1)(n-m_0+1)\frac{1}{1-\alpha_0/(1-\alpha_0)}\right) \\
&\geq \binom{n}{r_0-1}2^{-n}\left(n^2\left((1-\alpha_0)^2 - (1-(4\alpha_0-6\alpha_0^2+4\alpha_0^3))\frac{1-\alpha_0}{1-2\alpha_0}\right) - (1-\alpha_0)n\right) \\
&= \binom{n}{r_0-1}2^{-n}(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2 - n\right). \quad (5)
\end{aligned}$$

An analogous inequality also holds for A_1. We now assume that $r_0 \geq r_1$. Observe that we then have $m_0 \leq m_1$. Combining (4) and (5), we get

$$n^2\sum_{k=m_0}^{n-m_0} W_k \geq \sum_{k=m_0}^{n-m_0} k(n-k)W_k \geq \binom{n}{r_0-1}2^{-n}(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2 - n\right).$$

Note that for symmetric functions $\|\widehat{f}\|_1 = \sum_{k=0}^n \sqrt{\binom{n}{k}W_k}$, and thus

$$\begin{aligned}
\|\widehat{f}\|_1 &\geq \sum_{k=m_0}^{n-m_0}\sqrt{\binom{n}{k}W_k} \geq \sqrt{\binom{n}{m_0}\sum_{k=m_0}^{n-m_0} W_k} \\
&\geq \sqrt{\binom{n}{m_0}\binom{n}{r_0-1}2^{-n}\frac{(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2-n\right)}{n^2}} \\
&\geq \sqrt{\left(\left\lfloor n/2(1-\sqrt{4\alpha_0-6\alpha_0^2+4\alpha_0^3})\right\rfloor\right)\binom{n}{\alpha_0 n}2^{-n}\frac{(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2-n\right)}{n^2}}. \quad (6)
\end{aligned}$$

Using standard estimates for binomial coefficients, we obtain

$$\|\widehat{f}\|_1^2 \geq \frac{2^{n\left(h\left(\frac{1}{2}-\frac{1}{2}\sqrt{4\alpha_0-6\alpha_0^2+4\alpha_0^3}\right)+h(\alpha_0)-1\right)}}{n(n+1)^2}\cdot\frac{(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2-n\right)}{n^2}.$$

As a result

$$\log \|\widehat{f}\|_1 \geq \frac{n}{2}\left(h\left(\frac{1}{2}-\frac{1}{2}\sqrt{4\alpha_0-6\alpha_0^2+4\alpha_0^3}\right)+h(\alpha_0)-1\right) + \frac{1}{2}\log\frac{(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2-n\right)}{n^3(n+1)^2}.$$

Claim. There exists a constant $c>0$ such that for all $\alpha_0 \in (0,1/2)$,

$$h\left(\frac{1}{2}-\frac{1}{2}\sqrt{4\alpha_0-6\alpha_0^2+4\alpha_0^3}\right)+h(\alpha_0)-1 \geq c\cdot(1-2\alpha_0)\cdot\alpha_0\cdot\log(1/\alpha_0).$$

Using the claim, which is proved in the full version, we get

$$\log \|\widehat{f}\|_1 \geq c(1-2\alpha_0)\cdot\alpha_0\log(1/\alpha_0)\cdot\frac{n}{2}+\frac{1}{2}\log\frac{(1-\alpha_0)\left(\alpha_0(1-2\alpha_0)n^2-n\right)}{n^3(n+1)^2}.$$

This proves the desired result provided $r(f)$ is larger than some constant. To handle small values of $r(f)$, we refer the reader to the full version.

Lower Bound: $r \approx n/2$**.** For the case $r \approx n/2$, we use a result of [15] that states that the derivative of a symmetric Boolean function is noise sensitive. Here, we use the noise sensitivity of the derivative f_{ij}. The following lemma is an analogue of [15, Theorem 6] and is proved in the full version [1].

Lemma 5. *Let f be a symmetric Boolean function and f_{ij} be defined as in (1). Then for $\rho = 1-c/n$, we have*

$$\sum_S \widehat{f_{ij}}(S)^2\rho^{|S|} \leq \frac{4}{\sqrt{\pi c}}\cdot\sum_S \widehat{f_{ij}}(S)^2, \tag{7}$$

for any $c \in [1,n]$. Summing over all i,j with $i \neq j$, we get

$$8\sum_S |S|(n-|S|)\widehat{f}(S)^2\rho^{|S|} \leq \frac{4}{\sqrt{\pi c}}\cdot 8R(f). \tag{8}$$

We are now ready to prove the following result.

Lemma 6. *There exists a constant $\gamma < 1/2$ such that for any symmetric Boolean function f with $r(f) \geq \gamma n$, we have $\log \|\widehat{f}\|_1 = \Omega(n)$.*

Proof. Let $\rho = 1-c/n$ where c is a constant chosen later, and let n be large enough so that $\rho \geq 1/2$. We apply (8) to $g \stackrel{\text{def}}{=} f\cdot\text{PARITY}$:

$$\sum_S |S|(n-|S|)\widehat{g}(S)^2\rho^{|S|} \leq \frac{4}{\sqrt{\pi c}}\cdot R(g).$$

Note that $\text{PARITY} = \chi_{[n]}$ which shows $\widehat{f}([n]\setminus S) = \widehat{g}(S)$ for all S, and in particular $R(g) = R(f)$. So we can rewrite the above inequality as

$$\sum_S |S|(n-|S|)\widehat{f}(S)^2\rho^{n-|S|} \leq \frac{4}{\sqrt{\pi c}}\cdot R(f). \tag{9}$$

Summing (8) and (9), we get

$$\sum_S |S|(n-|S|)\widehat{f}(S)^2(1-\rho^{|S|}-\rho^{n-|S|}) \geq \left(1-\frac{8}{\sqrt{\pi c}}\right)R(f). \tag{10}$$

Let $\beta < 1/2$ be a positive constant to be chosen later. We have

$$\sum_{|S|\leq \beta n} |S|(n-|S|)\widehat{f}(S)^2(\rho^{|S|}+\rho^{n-|S|}) \geq \sum_{|S|\leq \beta n} |S|(n-|S|)\widehat{f}(S)^2(\rho^{\beta n}+\rho^{(1-\beta)n})$$
$$\geq \sum_{|S|\leq \beta n} |S|(n-|S|)\widehat{f}(S)^2(1/2\cdot e^{-c\beta}+1/2\cdot e^{-c(1-\beta)}).$$

For the first equality, we used the fact that $\rho^{|S|}+\rho^{n-|S|}$ is decreasing in $|S|$ for $|S|\leq n/2$. For the second inequality, we used the inequality $(1-c/n)^{\beta n} \geq e^{-c\beta}/2$ when $1-c/n \geq 1/2$. Summing this inequality with an analogous one for $|S| \geq (1-\beta)n$, we obtain Summing the two inequalities, we obtain

$$\sum_{|S|\notin(\beta n,(1-\beta)n)} |S|(n-|S|)\widehat{f}(S)^2(\rho^{|S|}+\rho^{n-|S|})$$
$$\geq \frac{e^{-c\beta}+e^{-c(1-\beta)}}{2} \sum_{|S|\notin(\beta n,(1-\beta)n)} |S|(n-|S|)\widehat{f}(S)^2.$$

Combining this with (10), we obtain

$$\sum_{\beta n\leq |S|\leq (1-\beta)n} |S|(n-|S|)\widehat{f}(S)^2(1-\rho^{|S|}-\rho^{n-|S|})$$
$$= \sum_S |S|(n-|S|)\widehat{f}(S)^2(1-\rho^{|S|}-\rho^{n-|S|}) - \sum_{|S|\notin(\beta n,(1-\beta)n)} |S|(n-|S|)\widehat{f}(S)^2(1-\rho^{|S|}-\rho^{n-|S|})$$
$$\geq (1-\frac{8}{\sqrt{\pi c}})R(f) - (1-e^{-c\beta}/2-e^{-c(1-\beta)}/2) \sum_{|S|\notin(\beta n,(1-\beta)n)} |S|(n-|S|)\widehat{f}(S)^2.$$

As $e^{-c\beta}/2+e^{-c(1-\beta)}/2 < 1$, this leads to

$$\sum_{\beta n\leq |S|\leq (1-\beta)n} |S|(n-|S|)\widehat{f}(S)^2(1-\rho^{|S|}-\rho^{n-|S|}) \geq \left(\frac{e^{-c\beta}+e^{-c(1-\beta)}}{2}-\frac{8}{\sqrt{\pi c}}\right)R(f).$$

Consequently,

$$\frac{n^2}{4}\sum_{\beta n\leq |S|\leq (1-\beta)n} \widehat{f}(S)^2 \geq R(f)\left(\frac{e^{-c\beta}+e^{-c(1-\beta)}}{2}-\frac{8}{\sqrt{\pi c}}\right).$$

By picking $c = 10^4$ and $\beta = 10^{-4}\ln 2$, we have $\frac{e^{-c\beta}+e^{-c(1-\beta)}}{2}-\frac{8}{\sqrt{\pi c}} \geq \frac{1}{10}$. We conclude that $\sum_{\beta n\leq k\leq (1-\beta)n} W_k \geq \frac{4R(f)}{10n^2}$, and thus

$$\|\widehat{f}\|_1 = \sum_{k=0}^n \sqrt{\binom{n}{k}W_k} \geq \sqrt{\binom{n}{\beta n}R(f)\frac{4}{10n^2}}.$$

Using (2), it follows that

$$\|\widehat{f}\|_1 = \Omega\left(\sqrt{\binom{n}{\beta n}\binom{n}{r-1}2^{-n}}\right) = \Omega\left(2^{(h(\beta)+h(\alpha)-1)\frac{n}{2}}(n+1)^{-1}\right),$$

where $\alpha = (r-1)/n$. If α is such that $h(\alpha) \geq 1 - h(\beta)/2$, we obtain the desired bound $\log \|\widehat{f}\|_1 = \Omega(n)$.

4 Conclusion and Future Work

A natural next step is to extend Theorem 1 to *approximate* spectral norm. Indeed this would have interesting implications. Recall that the ε-approximate spectral norm of a Boolean function f is the smallest spectral norm of a function g with $\|f-g\|_\infty \leq \varepsilon$, i.e., for all x, $|f(x)-g(x)| \leq \varepsilon$. Trivially $\|\widehat{f}\|_{1,\varepsilon}$ is smaller than $\|\widehat{f}\|_1$. We conjecture that it cannot be much smaller.

Conjecture 1. For all symmetric functions $f : \{0,1\}^n \to \{\pm 1\}$,

$$\log \|\widehat{f}\|_1 = \Theta^*(\log \|\widehat{f}\|_{1,1/3})$$

where Θ^* suppresses $O(\log n)$ factors.

We now discuss some of the applications of the above conjecture in conjunction with Theorem 1.

Analog of Paturi's Result for Monomial Complexity. A famous result of Paturi [16] characterizes the approximate degree of all symmetric functions. Recall that the degree of a function f is the largest $|S|$ such that $\widehat{f}(S)$ is non-zero. Let t_0 and t_1 be the minimum integers such that $f(i) = f(i+1)$ for all $i \in [t_0, n-t_1]$.

Theorem 2 ([16]). *Let $f : \{0,1\}^n \to \{\pm 1\}$ be a symmetric function and let t_0 and t_1 be defined as above. Then,* $\deg_{1/3}(f) = \Theta(\sqrt{n(t_0+t_1)})$.

Paturi's result has found numerous applications in theoretical computer science [17, 2, 18, 4, 19].

The monomial complexity of a Boolean function f, denoted $\mathrm{mon}(f)$, is the number of non-zero Fourier coefficients of f. The monomial complexity appears naturally in various areas of complexity theory, and it is desirable to obtain simple characterizations for natural classes of functions. With some additional observations, the combination of Conjecture 1 with Theorem 1 shows that $r(f)$ characterizes the approximate monomial complexity of f:

Conjecture 2 (Consequence of Conjecture 1). For a symmetric function $f : \{0,1\}^n \to \{\pm 1\}$, $\log \mathrm{mon}_{1/3}(f) = \Theta^*(r(f))$.

Communication Complexity of Xor Functions. Recall the Log Rank Conjecture mentioned in the introduction. This conjecture has an analogous version for the randomized communication complexity model: "Log Approximation Rank Conjecture". The ε-approximate rank of a matrix M is denoted by $\mathrm{rank}_\varepsilon(M)$, and is the minimum rank of a matrix that ε approximates M. Denote by $\mathbf{R}^\varepsilon(F)$ the ε-error randomized communication complexity of F. It is known that $\mathbf{R}^\varepsilon(F) \geq \log \mathrm{rank}_{\varepsilon'}(M_F)$, where ε' is a constant that depends on ε and M_F is the matrix representation of F. Log Approximation Rank Conjecture states that this lower bound is tight:

Conjecture 3 (Log Approximation Rank Conjecture). There is a universal constant c such that for any 2 party communication problem F,

$$\log \mathrm{rank}_{\varepsilon'}(M_F) \leq \mathbf{R}^\varepsilon(F) \leq \log^c \mathrm{rank}_{\varepsilon'}(M_F).$$

The important paper of Razborov [17] established this conjecture for the functions $F(x,y) = f(x \wedge y)$ where f is symmetric. In fact, Razborov showed that the quantum and classical randomized communication complexities of such functions are polynomially related. Later, Shi and Zhang [20], via a reduction to the case $f(x \wedge y)$, showed the quantum/classical equivalence for symmetric xor functions $F(x,y) = f(x \oplus y)$. They show that the randomized and quantum bounded error communication complexities of F are both $\Theta(r(f))$, up to polylog factor. However, their result does not verify the Log Approximation Rank Conjecture for symmetric xor functions.

Conjecture 1 along with Theorem 1 would verify the Log Approximation Rank Conjecture for symmetric xor functions. Furthermore, we would obtain a direct proof of the result of Shi and Zhang. This is very desirable since a major open problem is to understand the communication complexity of $f(x \oplus y)$ for general f (with no symmetry condition on f). There is a sentiment that this should be easier to tackle than $f(x \wedge y)$ as xor functions seem more amenable to Fourier analytic techniques. A direct proof of the result of Shi and Zhang gives more insight into the communication complexity of xor functions.

Agnostically Learning Symmetric Functions. Let $\mathcal{C}$ be a concept class and $g_i : \{-1,1\}^n \to \mathbb{R}$ be functions for $1 \leq i \leq s$ such that every $f : \{-1,1\}^n \to \{-1,1\}$ in $\mathcal{C}$ satisfies $\|f - \sum_{i=1}^s c_i g_i\|_\infty \leq \varepsilon$, for some reals c_i. The smallest s for which such g_i's exist corresponds to the ε-approximate rank of $\mathcal{C}$. If each $g_i(x)$ is computable in polynomial time, then $\mathcal{C}$ can be agnostically learned under any distribution in time $\mathrm{poly}(n,s)$ and with accuracy ε [9].

Klivans and Sherstov [10] proved lower bounds on the approximate rank of the concept class of disjunctions $\{\bigvee_{i \in S} x_i : S \subseteq [n]\}$ and majority functions $\{\mathrm{MAJ}(\pm x_1, \pm x_2, \ldots, \pm x_n)\}$ thereby ruled out the possibility of applying the algorithm of [9] to agnostically learning these concept classes.

Theorem 1 together with Conjecture 1 provides additional negative results and gives strong lower bounds on the approximate rank of the concept class consisting of symmetric functions f with large $r(f)$.

References

1. Ada, A., Fawzi, O., Hatami, H.: Spectral norm of symmetric functions. arXiv:1205.5282 (2012)
2. Beals, R., Buhrman, H., Cleve, R., Mosca, M., de Wolf, R.: Quantum lower bounds by polynomials. J. ACM 48(4), 778–797 (2001)
3. Bruck, J., Smolensky, R.: Polynomial threshold functions, ac0 functions, and spectral norms. SIAM J. Comput. 21(1), 33–42 (1992)
4. de Wolf, R.: A note on quantum algorithms and the minimal degree of ε-error polynomials for symmetric functions. Quantum Inf. Comput. 8(10), 943–950 (2008)
5. Goldmann, M., Håstad, J., Razborov, A.A.: Majority gates vs. general weighted threshold gates. Comput. Complex., 277–300 (1992)
6. Green, B., Sanders, T.: Boolean functions with small spectral norm. Geom. Funct. Anal. 18(1), 144–162 (2008)
7. Grolmusz, V.: On the power of circuits with gates of low l1 norms. Theor. Comput. Sci. 188(1-2), 117–128 (1997)
8. Grolmusz, V.: Harmonic analysis, real approximation, and the communication complexity of boolean functions. Algorithmica 23(4), 341–353 (1999)
9. Kalai, A.T., Klivans, A.R., Mansour, Y., Servedio, R.A.: Agnostically learning halfspaces. SIAM J. Comput. 37(6), 1777–1805 (2008)
10. Klivans, A.R., Sherstov, A.A.: Lower bounds for agnostic learning via approximate rank. Comput. Complex. 19(4), 581–604 (2010)
11. Kolountzakis, M.N., Lipton, R.J., Markakis, E., Mehta, A., Vishnoi, N.K.: On the fourier spectrum of symmetric boolean functions. Combinatorica 29(3), 363–387 (2009)
12. Kushilevitz, E., Mansour, Y.: Learning decision trees using the fourier spectrum. In: Proceedings of the Twenty-Third Annual ACM Symposium on Theory of Computing, STOC 1991, pp. 455–464. ACM, New York (1991)
13. Lee, T., Shraibman, A.: Lower bounds in communication complexity, vol. 3. Now Publishers Inc. (2009)
14. O'Donnell, R., Servedio, R.A.: Extremal properties of polynomial threshold functions. J. Comput. Syst. Sci. 74(3), 298–312 (2008)
15. O'Donnell, R., Wright, J., Zhou, Y.: The Fourier Entropy–Influence Conjecture for Certain Classes of Boolean Functions. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part I. LNCS, vol. 6755, pp. 330–341. Springer, Heidelberg (2011)
16. Paturi, R.: On the degree of polynomials that approximate symmetric Boolean functions (preliminary version). In: Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing, pp. 468–474. ACM, New York (1992)
17. Razborov, A.: Quantum communication complexity of symmetric predicates. Izvestiya: Mathematics 67(1), 145–159 (2003)
18. Sherstov, A.A.: Approximate inclusion-exclusion for arbitrary symmetric functions. Comput. Complex. 18(2), 219–246 (2009)
19. Sherstov, A.A.: The pattern matrix method. SIAM J. Comput. 40(6), 1969–2000 (2011)
20. Shi, Y., Zhang, Z.: Communication complexities of symmetric XOR functions. Quantum Inf. Comput. (available at arXiv:0808.1762) 9, 255–263 (2009)
21. Shpilka, A., Tal, A.: On the minimal fourier degree of symmetric boolean functions. In: Proceedings of the 2011 IEEE 26th Annual Conference on Computational Complexity, CCC 2011, pp. 200–209. IEEE Computer Society, Washington, DC (2011)
22. Siu, K.-Y., Bruck, J.: On the power of threshold circuits with small weights. SIAM J. Discrete Math. 4(3), 423–435 (1991)

Almost K-Wise vs. K-Wise Independent Permutations, and Uniformity for General Group Actions

Noga Alon[1,⋆] and Shachar Lovett[2,⋆⋆]

[1] Tel-Aviv University and the Institute for Advanced Study
nogaa@tau.ac.il
[2] The Institute for Advanced Study
slovett@math.ias.edu

Abstract. A family of permutations in S_n is k-wise independent if a uniform permutation chosen from the family maps any distinct k elements to any distinct k elements equally likely. Efficient constructions of k-wise independent permutations are known for $k = 2$ and $k = 3$, but are unknown for $k \geq 4$. In fact, it is known that there are no nontrivial subgroups of S_n for $n \geq 25$ which are 4-wise independent. Faced with this adversity, research has turned towards constructing almost k-wise independent families, where small errors are allowed. Optimal constructions of almost k-wise independent families of permutations were achieved by several authors.

Our first result is that any such family with small enough error is statistically close to a distribution which is perfectly k-wise independent. This allows for a simplified analysis of algorithms: an algorithm which uses randomized permutations can be analyzed assuming perfect k-wise independence, and then applied to an almost k-wise independent family. In particular, it allows for an oblivious derandomization of two-sided randomized algorithms which work correctly given any k-wise independent distribution of permutations.

Another model is that of weighted families of permutations, or equivalently distributions of small support. We establish two results in this model. First, we show that a small random set of $n^{O(k)}$ permutations w.h.p supports a k-wise independent distribution. We then derandomize this by showing that any almost $2k$-wise independent family supports a k-wise independent distribution. This allows for oblivious derandomization of algorithms for search problems which work correctly given perfect k-wise independent distributions.

These results are all in fact special cases of a general framework where a group acts on a set. In the aforementioned case, the group of permutations acts on tuples of k elements. We prove all the above results in the general setting of the action of a finite group on a finite set.

⋆ Supported in part by an ERC advanced grant and by NSF grant DMS-0835373.
⋆⋆ Supported by NSF grant DMS-0835373.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 350–361, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1 Introduction

Small probability spaces of limited independence are widely used in many applications. Specifically, if the analysis of a randomized algorithm depends only on the assumption that the entries are k-wise independent, one can replace the random tape by a tape selected from a k-wise independent distribution. One application of this is a derandomization of the algorithm by enumerating over all possible random strings. Another application is when the random string needs to be saved, for example in data structures, where using k-wise independence allows one to maintain a succinct data structure.

The case of k-wise independent distributions over $\{0,1\}^n$ has been widely studied, and there are optimal constructions of k-wise independent probability spaces of size $n^{O(k)}$ (see e.g. [ABI86]). Moreover, these constructions are *strongly explicit*: given an index of an element $i \in [n^{O(k)}]$ and an index of a bit $j \in [n]$, one can compute the j-th bit of the i-th string in time $O(k \log n)$. This is crucial for several applications, for example for streaming algorithms and cryptography, where operations need to be performed in poly-logarithmic time.

Another widely studied case is that of k-wise independent permutations of n elements. This problem is motivated by cryptographic applications, as k-wise independent permutations allow perfect secrecy even if one allows k oracle queries to the encryption. For more details on the role of k-wise independent permutations in cryptography, see, e.g., [RW06, Vau98, Vau00, Vau03].

Here, the situation is much less understood. For $k = 2$ the group of invertible affine transformations $x \mapsto ax+b$ over a finite field $\mathbb{F}$ yields a 2-wise independent family; and for $k = 3$ the group of Möbius transformations $x \mapsto (ax+b)/(cx+d)$ with $ad - bc = 1$ over the projective line $\mathbb{F} \cup \{\infty\}$ yields a 3-wise independent family. For $k \geq 4$ (and n large enough), however, no k-wise independent family is known, other than the full symmetric group S_n and the alternating group A_n. In fact, it is known (c.f.., e.g., [Cam95], Theorem 5.2) that for $n \geq 25$ and $k \geq 4$ there are no other subgroups of S_n which form a k-wise independent family[1]. This is a major obstacle, while as groups are by no means the only way to produce such families, algebraic techniques are among the most useful in combinatorics, and the lack of algebraic structure is a serious drawback. Recently, Kuperberg, Lovett and Peled [KLP12] were able to show by a probabilistic argument that there exist small families of permutations which are k-wise independence. Still, it is unknown how to find these families efficiently.

Faced with this adversity, research has turned towards constructing families of permutations which are *almost k-wise independent*, allowing for small errors. There has been much research towards constructing explicit almost k-wise independent families of minimal size. This was achieved, up to polynomial factors, by Kaplan, Naor and Reingold [KNR05], who gave a construction of such a family of size $n^{O(k)}$. Alternatively, one can start with the constant size expanding set of S_n given by Kassabov [Kas07], and take a random walk on it of length

[1] In the language of group theory, these are k-transitive groups. The currently known proof of this fact is hard, as it requires the classification of finite simple groups.

$O(k \log n)$. Both of these constructions are also strongly explicit: given an index of a permutation $i \in [n^{O(k)}]$ and an element $j \in [n]$, one can compute the image of the i-th permutation on j in time $O(k \log n)$. Again, this is crucial for applications such as streaming algorithms or cryptography.

For many applications, almost k-wise independent families are just as good as perfect k-wise independent families. However, the analysis must take into account the error, which in some cases is not trivial. Our first result shows that by choosing the error small enough, one can analyze an algorithm using perfect k-wise independent permutations, and then apply almost k-wise independent permutations to achieve almost the same results.

Theorem 1. *Let μ be a distribution taking values in S_n which is almost k-wise independent with error $\varepsilon \cdot n^{-O(k)}$. Then there exists a distribution over permutations μ' which is k-wise independent, and such that the statistical distance between μ and μ' is at most ε.*

A similar result for k-wise independent hash functions was obtained by Alon, Goldreich and Mansour [AGM03], and more generally over product spaces by Rubinfeld and Xie [RX10]. Our proof technique is similar in spirit, although technically more involved. This allows for an oblivious derandomization of two-sided algorithms which "work" given any k-wise independent distribution over permutations: let f be a boolean function, and let A be a randomized algorithm such that

$$\Pr_{\pi \sim \mu}[A(x, \pi) = f(x)] \geq 2/3$$

for any k-wise independent distribution over permutations μ. Then A can be derandomized by letting π be chosen uniformly from an almost k-wise independent distribution with error $n^{-O(k)}$. Since such distributions can be generated strongly explicitly, the overhead (in terms of the number of bits needed to sample from the distribution) is just $O(k \log n)$.

A relaxation of the problem of constructing small families of k-wise independent permutations is that of considering weighted families, or equivalently distributions of small support which are k-wise independent. Contrary to the case of unweighted families, it is simple to establish that there exist distributions of small support which are k-wise independent. First, note that given a family S of permutations, it is easy to decide if there exists a distribution μ supported on S which is k-wise independent, using linear programming: for a permutation π define the matrix $M_k(\pi)$ to be the permutation on distinct k-tuples induced by π. It is an $(n)_k \times (n)_k$ permutation matrix, where $(n)_k := \prod_{i=0}^{k-1}(n-i)$. Let U denote the uniform matrix all whose elements are $(n-k)!/n!$. Then there exists a k-wise independent distribution supported on S iff U belongs to the convex hull of $\{M_k(\pi) : \pi \in S\}$. The latter condition can be easily verified using linear programming. Now, starting with any set of permutations which support k-wise independent permutations (for example the set of all permutations), one can apply Carathéodory theorem, and deduce that U lies in the convex hull of at most n^{2k} permutations. That is, there exist k-wise independent distributions

which are supported on at most n^{2k} permutations. Moreover, and somewhat surprisingly, one can algorithmically find a k-wise independent distribution with small support in a *weakly explicit* manner (i.e. in time $n^{O(k)}$) using the ideas of Karp and Papadimitriou [KP82] and Koller and Megiddo [KM94][2].

We consider the problem of constructing small explicit sets which support k-wise independent distributions. First, we establish that most small sets support k-wise independent distributions.

Theorem 2. *Let S be a random subset of S_n of size n^{6k}. Then with high probability (w.h.p, for short) there exists a distribution μ supported on S which is k-wise independent.*

A similar result for k-wise independent hash functions was obtained by Austrin and Håstad [AH11]. Our result implies a somewhat surprising consequence for search algorithms which "work" given any k-wise independent distribution over permutations, which allows to transform weak guarantees to strong guarantees. Let f be a function and A an algorithm, such that for any k-wise independent distribution μ,

$$\Pr_{\pi\sim\mu}[A(x,\pi) = f(x)] > 0.$$

Then since almost all sets of size $n^{O(k)}$ support such a distribution, we must have that A has a noticeable fraction of witnesses in S_n,

$$\Pr_{\pi\in S_n}[A(x,\pi) = f(x)] \geq n^{-O(k)}.$$

We also show that almost $2k$-wise independent permutations give an explicit construction of a set which supports k-wise independence, thus derandomizing Theorem 2.

Theorem 3. *Let S be a subset of S_n such that S is almost $2k$-wise independent with error $n^{-O(k)}$. Then there exists a distribution μ supported on S which is k-wise independent.*

We are not aware of a similar result, even in the case of k-wise independent hash functions. This allows for an oblivious derandomization of search algorithms which "work" given any k-wise independent distribution over permutations: let f be a function, and let A be a randomized algorithm such that

$$\Pr_{\pi\sim\mu}[A(x,\pi) = f(x)] > 0$$

for any k-wise independent distribution μ over permutations. Then taking S to be an almost $2k$-wise independent family of permutations with error $n^{-O(k)}$, we

[2] Essentially, the linear program for finding μ has $n!$ variables and $n^{O(k)}$ constraints. Its dual has $n^{O(k)}$ variables and $n!$ constraints. The dual problem can be solved efficiently using the ellipsoid method since it has an efficient separating-hyperplane oracle.

get that there exists $\pi \in S$ for which $A(x,\pi) = f(x)$, achieving an oblivious derandomization of A with overhead (measured in bits, as before) $O(k \log n)$.

Here is a toy example illustrating the way the last theorem and the discussion preceding it can be applied. Let $G = (V, E)$ be a graph on a set V of n vertices, and suppose that each vertex $v \in V$ has a real positive weight $w(v)$. Let $d(v)$ be the degree of v, and assume all degrees are bounded by k. We claim that G contains an independent set $U \subset V$ of total weight $W(U) = \sum_{u \in U} w(u)$ at least $\sum_{v \in V} \frac{w(v)}{d(v)+1}$. To prove it, let π be a random permutation of the set of vertices V, and let U consist of all vertices u so that $\pi(u)$ precedes $\pi(v)$ for every neighbor v of u. It is clear that U is an independent set, and for any vertex $u \in V$ the probability that $u \in U$ is exactly $\frac{1}{d(u)+1}$, as this is the probability that u precedes all its neighbors. By linearity of expectation, the expected value of the total weight of U is $\sum_{v \in V} \frac{w(v)}{d(v)+1}$ and hence there exists an independent set U of total weight at least as claimed.

The above proof clearly works even if π is only assumed to be $(k+1)$-wise independent (in fact, a weaker condition suffices, we only need π to be $(k+1)$-minwise independent). Therefore, the discussion preceding Theorem 3 implies that if π is chosen uniformly at random, then the probability it provides a set U satisfying $W(U) \geq \sum_{v \in V} \frac{w(v)}{d(v)+1}$, is at least $n^{-O(k)}$. The theorem itself shows that the support of any set of almost $(2k+2)$-wise independent permutations with sufficiently small error must contain a permutation π that provides an independent set U as above.

A similar reasoning can be applied to other arrangement problems. Given a k-uniform hypergraph with a weight for each permutation of the vertices in each of its edges, one may want to find a permutation maximizing the total weight of all orders induced on the sets of vertices in the edges. Problems of this type are called k-CSP-rank problems, (see, e.g., [AA07]), and include Betweenness and Feedback Arc Set. In most of these problems, finding the precise optimum is NP-hard, and the reasoning above provides some insight about algorithms for the (much easier) problem of finding a permutation in which the total weight is at least as large as the expected weight in a uniform random permutation.

1.1 Group Action Uniformity vs. Almost Uniformity

We actually prove all the aforementioned results in the general setting of *group actions*, of which k-wise independent permutations as well as k-wise independent random variables form specific instances. A group G acts on a set X if G acts as a group of permutations on X. That is, $g : X \to X$ is a permutation of X for all $g \in G$, and $(gh)(x) = g(h(x))$ for all $g, h \in G$ and $x \in X$. This gives a general framework: k-wise independent permutations correspond to the case of $G = S_n$ the group of permutations, and $X = [n]_k = \{i_1, \ldots, i_k \in [n] \text{ distinct}\}$ the set of (ordered) distinct k-tuples, where the action of G on X is straightforward. The case of k-wise independent distributions over $\{0,1\}^n$ corresponds to $G = \mathbb{F}_2^n$ and $X = [n]_k \times \mathbb{F}_2^k$, where the action of $g = (g_1, \ldots, g_n) \in \mathbb{F}_2^n$ on $x = ((i_1, \ldots, i_k), (b_1, \ldots, b_k)) \in [n]_k \times \mathbb{F}_2^k$ is given by

$g(x) = ((i_1, \ldots, i_k), (b_1 + g_{i_1}, \ldots, b_k + g_{i_k}))$. Similarly, one can obtain in this way distributions supporting k-wise independent random variables, even when each variable is distributed over a different domain.

We now introduce some definitions. If G acts on X, a distribution μ over G is X-uniform if

$$\Pr_{g\sim\mu}[g(x) = y] = \Pr_{g\in G}[g(x) = y]$$

for all $x, y \in X$; and is almost X-uniform with error ε if

$$\left|\Pr_{g\sim\mu}[g(x) = y] - \Pr_{g\in G}[g(x) = y]\right| \leq \varepsilon$$

for all $x, y \in X$. These definitions coincide with k-wise independence and almost k-wise independence for permutations when $G = S_n$ and $X = [n]_k$. Theorem 1, Theorem 2 and Theorem 3 are immediate corollaries of the following general theorems, when applied to $G = S_n$ and $X = [n]_k$.

First, we show that distributions over G which are almost X-uniform with small enough error, are close in statistical distance to distributions which are X-uniform.

Theorem 4 (informal version). *Let μ be a distribution over G which is almost X-uniform with error $\varepsilon \cdot |X|^{-O(1)}$. Then there exists a distribution μ' on G which is X-uniform, and such that the statistical distance between μ and μ' is at most ε.*

Second, we show that a small random subset of G supports w.h.p a X-uniform distribution.

Theorem 5 (informal version). *Let $S \subset G$ be a random set of size $|X|^{O(1)}$. Then w.h.p there exists a distribution μ supported on S which is X-uniform.*

Finally, we derandomized Theorem 5. Recall that if G acts on X, then G also acts on $X \times X$ in the obvious manner, i.e. $g((x_1, x_2)) = (g(x_1), g(x_2))$. We show that if a distribution over G is almost $X \times X$ uniform with a small enough error, then it must support an X-uniform distribution.

Theorem 6 (informal version). *Let μ be a distribution supported on a set $S \subset G$ which is almost $(X \times X)$-uniform with error $|X|^{-O(1)}$. Then there exists a distribution μ' supported on S which is X-uniform.*

The proof of Theorem 5 is by a counting argument using the symmetry of the group action. The proofs of Theorem 4 and Theorem 6 rely on representation theory of finite groups. In the language of Fourier analysis literature, we prove results regarding quadrature rules for the representations appearing in the action of G on X. Technically, our arguments involving representation theory are quite basic, and as such are similar in spirit to several known results in the Fourier analysis literature. In particular, Theorem 5 is similar to theorems established in [KORT01, AR11]. However, our proof in arguably simpler, as it applies the Carathéodory theorem instead of a more involved second moment argument.

Also, some technical parts used in the proof of Theorem 6 are related to known results in the Fourier analysis literature, e.g. in [Mas98, RS09].

Paper organization Theorem 4 is proved in Section 2, Theorem 5 in Section 3 and Theorem 6 in Section 4. We conclude with some open problems in Section 5. For lack of space, some preliminary definitions and many proofs are omitted and can be found in the full version of this paper [AL11]. We note that throughout the paper we do not attempt to optimize constants.

2 Almost X-Uniform Distributions are Statistically Close to X-Uniform Distributions

We prove in this section Theorem 4, which states that almost X-uniform distributions with small enough error are statistically close to X-uniform distributions.

Theorem 4. *Let μ be a distribution on G which is almost X-uniform with error ε. Then there exists a distribution μ' on G which is X-uniform, and such that the statistical distance between μ and μ' is at most $\varepsilon \cdot 3|X|^4$.*

We first rephrase the conditions for a distribution to be X-uniform, or almost X uniform, in terms of representations. Let R_X be the representation of the action of G on X, i.e. $R_X(g)_{x,y} = 1_{g(x)=y}$. Let U_G denote the uniform distribution over G.

Proposition 1. *Let μ be a distribution on G. Then μ is X-uniform iff $R_X(\mu) = R_X(U_G)$; and μ is almost X-uniform with error ε iff $\|R_X(\mu) - R_X(U_G)\|_\infty \leq \varepsilon$.*

Proof. Omitted.

The first step is to decompose R_X into its irreducible representations. Let $R_X \equiv e_0\mathbf{1} + e_1R_1 + \ldots + e_tR_t$, where $R_1, \ldots, R_t$ are unitary nonequivalent non-trivial irreducible representations, and e_i is the multiplicity of R_i in R_X. We next transform the conditions of Proposition 1 to the basis of the irreducible representations.

Proposition 2. *Let μ be a distribution on G. Then μ is X-uniform iff $R_i(\mu) = 0$ for all $i \in [t]$; and if μ is almost X-uniform with error ε then $\|R_i(\mu)\|_\infty \leq \varepsilon|X|$ for all $i \in [t]$.*

Proof. Omitted.

The main idea in the proof of Theorem 4 is to "correct" each element of $R_i(\mu)$ to be zero by making a small statistical change in μ, and without affecting the other elements of R_i or in any other $R_{i'}$. This is analogous to the proof idea of [AGM03] for almost k-wise independent bits (see also [AAK+07]). Performing all these local changes sequentially over all elements of R_i, $i \in [t]$, will shift μ into an X-uniform distribution. Actually, as a first step we will get a general element in $\mathbb{C}[G]$, which we then rectify to be a distribution.

Let R_i be one of the irreducible representations, and let $d_i = \dim(R_i)$ be its dimension. For $j, k \in [d_i]$ we define $\Delta_{i,j,k} \in \mathbb{C}[G]$ as

$$\Delta_{i,j,k}(g) = \frac{d_i}{|G|}\overline{R_i(g)_{j,k}}.$$

We consider how shifting μ by a small multiple of $\Delta_{i,j,k}$ affects the entries of $R_1, \ldots, R_t$.

Proposition 3. *Let $i \in [t], j, k \in [d_i]$ and $i' \in [t], j', k' \in [d_{i'}]$. For any $\delta \in \mathbb{R}$ we have*

$$R_{i'}(\mu + \delta\Delta_{i,j,k})_{j',k'} = R_{i'}(\mu)_{j',k'} + \delta \cdot 1_{(i,j,k)=(i',j',k')}.$$

Proof. Omitted.

We will also need the following proposition, which asserts that $\mathbf{1}(\Delta_{i,j,k}) = 0$ and that $\|\Delta_{i,j,k}\|_\infty$ is bounded.

Proposition 4. *Let $i \in [t], j, k \in [d_i]$. Then $\mathbf{1}(\Delta_{i,j,k}) = 0$ and $\|\Delta_{i,j,k}\|_\infty \leq \frac{|X|}{|G|}$.*

Proof. Omitted.

Applying Proposition 3 and Proposition 4 iteratively over all elements of $R_1, \ldots, R_t$, we obtain the following corollary.

Corollary 1. *Let μ be a distribution over G which is almost X-uniform with error ε. Define $\Delta \in \mathbb{C}[G]$ by*

$$\Delta(g) = -\sum_{i \in [t]} \sum_{j,k \in [d_i]} R_i(\mu)_{j,k} \cdot \Delta_{i,j,k}(g).$$

Then $R_X(\mu + \Delta) = R_X(U_G)$ and $\|\Delta\|_\infty \leq \frac{\varepsilon |X|^4}{|G|}$.

Proof. Omitted.

We are nearly done. The only problem is that $\mu + \Delta$ may not be a distribution: it may be complex, or have negative values. This can be fixed, without increasing the statistical distance too much. This resolves the proof of Theorem 4. Details are omitted.

3 Random Sets Support X-Uniform Distributions

We establish Theorem 5 in this section, which states that w.h.p a random set of size $|X|^{O(1)}$ supports an X-uniform distribution.

Theorem 5. *Let $S \subset G$ be a random set of size $O(|X|^6)$. Then with probability 0.99 over the choice of S, there exists a distribution μ supported on S which is X-uniform.*

Recall that a distribution μ is X-uniform if $\Pr_{g\sim\mu}[g(x) = y] = \Pr_{g\in G}[g(x) = y]$ for all $x, y \in X$. We say a set S *supports X-uniformity* if there exists a distribution supported on S which is X-uniform. We first establish that this a purely geometric property of S.

Let R_X be the representation of the action of G on X, that is, $R_X(g)_{x,y} = 1_{g(x)=y}$. Let $U = R_X(U_G) = \mathbb{E}_{g\in G}[R_X(g)]$ denote the matrix which corresponds to the action on X of the uniform distribution over G. We consider these matrices as points in $\mathbb{R}^d$ for $d = |X|^2$.

Proposition 5. *A set $S \subset G$ supports X-uniformity iff the convex hull of the matrices $\{R_X(g) : g \in S\}$ contains the matrix U.*

Proof. Omitted.

Let $S \subset G$ be a random set. By Proposition 5 it is enough to show that the matrix U lies in the convex hull of $\{R_X(g) : g \in S\}$. Suppose this is not the case; then there must exist a hyperplane H in $\mathbb{R}^d$ which passes through U and such that all matrices $\{R_X(g) : g \in S\}$ lie on one side of H. We first show that any hyperplane which passes through U has a noticeable fraction of the matrices $\{R_X(g) : g \in G\}$ on both sides.

Proposition 6. *Let H be a hyperplane which passes through U. The number of matrices $\{R_X(g) : g \in G\}$ on any side of H is at least $|G|/(|X|^2 + 1)$.*

Proof. Omitted.

We now establish Theorem 5.

Proof (Proof of Theorem 5). Let $S \subset G$ be a random set of N elements, chosen with repetitions. Let $K \lhd G$ be the normal subgroup of G which acts trivially on X, i.e. $K = \{g \in G : g(x) = x\ \forall x \in X\}$. Observe that the quotient group G/K also acts on X, and that $\{R_X(g) : g \in G\} = \{R_X(g) : g \in G/K\}$. Thus the number of distinct matrices $R_X(g)$ is bounded by $|G/K| \le |X|!$, and by a standard VC dimension argument the number of ways to partition this set of matrices by any hyperplane, and in particular one which passes through U, is bounded by $(|X|!)^d$. Fix such a partition. The number of matrices $\{R_x(g) : g \in G\}$ which lies on each side of the partition is at least $|G|/(d+1)$ by Proposition 6. Hence, the probability that S is contained in one side of the partition is bounded by $2(1-1/(d+1))^N$. Thus, by the union bound, the probability that there exists a hyperplane passing through U, such that S is contained in one side of it, is at most

$$|G/K|^d \cdot 2\left(1 - \frac{1}{d+1}\right)^N \le 2\exp(-N/(d+1) + d\log(|X|!)),$$

which is at most 0.01 for $N = O(d^2 \log(|X|!)) \le O(|X|^6)$.

4 Almost X-Uniform Distributions Support X-Uniform Distributions

We prove in this section Theorem 6, which states that if μ is an almost $X \times X$-uniform distribution with small enough error, then there exists an X-uniform distribution μ' supported on the support of μ.

Theorem 6. *Let μ be a distribution supported on a set $S \subset G$ which is almost $(X \times X)$-uniform with error $\varepsilon < 0.5|X|^{-7}$. Then there exists a distribution μ' supported on S which is X-uniform.*

Fix such a distribution μ, and let S denote its support, $S = \{g : \mu(g) > 0\}$. Let R_X be the representation of G acting on X. By Proposition 5, S supports an X-uniform distribution iff $R_X(U_G) = \mathbb{E}_{g \in G}[R_X(g)]$ lies in the convex hull of $\{R_X(g) : g \in S\}$. Assume this is not the case; then there exists an hyperplane H which passes through $R_X(U_G)$ and such that all $\{R_X(g) : g \in S\}$ lie on one side of H.

We first project H into an hyperplane with a simpler representation. Let $R_X \equiv e_0\mathbf{1}+e_1R_1+\ldots+e_tR_t$ denote the decomposition of R_X into unitary nonequivalent irreducible representation, and let $d_i = \dim(R_i)$ denote the dimension of each irreducible representation. Essentially, we will project H to "use" only one copy from each nontrivial irreducible representation. That is, we will show that H can be projected to a hyperplane separating 0 from $\{R_1(g) \times \ldots \times R_t(g) : g \in S\}$.

Proposition 7. *There exists a map $L : G \to \mathbb{R}$ given by*

$$L(g) := \sum_{i \in [t]} \sum_{j,k \in [d_i]} \lambda_{i,j,k} \cdot R_i(g)_{j,k}$$

for some coefficients $\{\lambda_{i,j,k} \in \mathbb{C} : i \in [t], j, k \in [d_i]\}$ such that $\mathbb{E}_{g \in G}[L(g)] = 0$ and $L(g) > 0$ for all $g \in S$.

Proof. Omitted.

We may assume w.l.o.g that $\mathbb{E}_{g \in G}[L^2(g)] = 1$ by multiplying all coefficients $\lambda_{i,j,k}$ by an appropriate factor. The main idea is to show that if μ is almost $X \times X$ uniform, then $\mathbb{E}_{g \sim \mu}[L^2(g)] \approx \mathbb{E}_{g \in G}[L^2(g)] = 1$ while $\mathbb{E}_{g \sim \mu}[L(g)] \approx \mathbb{E}_{g \in G}[L(g)] = 0$. Combining this with a bound on $\|L\|_\infty$ a simple calculation shows that it cannot be the case that $L(g) > 0$ for all g in the support of μ.

The first step is to show that the coefficients $\lambda_{i,j,k}$ cannot be very large.

Proposition 8. $\sum_{i \in [t]} \sum_{j,k \in [d_i]} \frac{|\lambda_{i,j,k}|^2}{d_i} = 1$. *In particular, $|\lambda_{i,j,k}| \leq |X|^{1/2}$ for all i, j, k.*

Proof. Omitted.

An immediate corollary is that $L(g)$ can never be very large.

Corollary 2. *$|L(g)| \leq |X|^{2.5}$ for all $g \in G$.*

Proof. Omitted.

The bound on $|\lambda_{i,j,k}|$ together with the assumption that μ is almost $X \times X$-uniform, implies that the first and second moment of L are approximately the same under μ and under the uniform distribution over G.

Proposition 9. *Let μ be a distribution which is almost $X \times X$-uniform with error ε. Then $|\mathbb{E}_{g\sim\mu}[L(g)]| \leq \varepsilon|X|^{4.5}$ and $|\mathbb{E}_{g\sim\mu}[L^2(g)] - 1| \leq \varepsilon|X|^7$.*

Proof. Omitted.

Proof (Proof of Theorem 6). Let μ be almost $X \times X$ uniform with error $\varepsilon \leq 0.5|X|^{-7}$. Summarizing Corollary 2 and Proposition 9, we have $\|L\|_\infty \leq |X|^{2.5}$, $E_{g\sim\mu}[L(g)] \leq \varepsilon|X|^{4.5}$ and $E_{g\sim\mu}[L(g)^2] \geq 1 - \varepsilon|X|^7$. However, since we assumed by contradiction that $L(g) > 0$ for all g in the support of μ, we have $\mathbb{E}_{g\sim\mu}[L(g)^2] \leq \|L(g)\|_\infty \cdot \mathbb{E}_{g\sim\mu}[L(g)] \leq |X|^{2.5} \cdot \varepsilon|X|^{4.5}$, i.e. we have $1 - \varepsilon|X|^7 \leq \varepsilon|X|^7$, which is false whenever $\varepsilon < 0.5|X|^{-7}$.

5 Summary and Open Problems

We showed that almost X-uniform (or $X \times X$-uniform) distributions are close to perfect X uniform distributions in two ways: they are statistically close to some X-uniform distribution μ', and they support a X-uniform distribution μ''. It may be possible that both can be realized by the same X-uniform distribution, i.e. that $\mu' = \mu''$. We leave this as an open problem.

Another interesting combinatorial problem is to construct small sets which are perfectly uniform. This is unknown even in the special case of k-wise independent permutations. Recently, Kuperberg et al. [KLP12] gave a non-explicit proof for the existence of small families of k-wise independent permutations. It is intriguing whether their argument can be adapted to efficiently construct these families.

Acknowledgements. We thank Avi Wigderson for helpful discussions and reference to the work of Karp and Papadimitriou [KP82].

References

[AA07] Ailon, N., Alon, N.: Hardness of fully dense problems. Inform. and Comput. 205(8), 1117–1129 (2007)

[AAK+07] Alon, N., Andoni, A., Kaufman, T., Matulef, K., Rubinfeld, R., Xie, N.: Testing k-wise and almost k-wise independence. In: STOC 2007, pp. 496–505. ACM, New York (2007)

[ABI86] Alon, N., Babai, L., Itai, A.: A fast and simple randomized algorithm for the maximal independent set problem. Journal of Algorithms 7, 567–583 (1986)

[AGM03] Alon, N., Goldreich, O., Mansour, Y.: Almost k-wise independence versus k-wise independence. Inf. Process. Lett. 88, 107–110 (2003)

[AH11] Austrin, P., Håstad, J.: Randomly supported independence and resistance. SIAM J. Comput. 40(1), 1–27 (2011)

[AL11] Alon, N., Lovett, S.: Almost k-wise vs. k-wise independent permutations, and uniformity for general group actions. Electronic Colloquium on Computational Complexity, ECCC (2011)

[AR11] Alagic, G., Russell, A.: Spectral Concentration of Positive Functions on Compact Groups. Journal of Fourier Analysis and Applications, 1–19 (February 2011)

[Cam95] Cameron, P.J.: Permutation groups. In: Handbook of Combinatorics, vol. 1, 2, pp. 611–645. Elsevier, Amsterdam (1995)

[Kas07] Kassabov, M.: Symmetric groups and expanders. Inventiones Mathematicae 170(2), 327–354 (2007)

[KLP12] Kuperberg, G., Lovett, S., Peled, R.: Probabilistic existence of rigid combinatorial structures. In: Proceedings of the 44th Symposium on Theory of Computing, STOC 2012, pp. 1091–1106. ACM, New York (2012)

[KM94] Koller, D., Megiddo, N.: Constructing small sample spaces satisfying given constraints. SIAM Journal on Discrete Mathematics 7, 260–274 (1994)

[KNR05] Kaplan, E., Naor, M., Reingold, O.: Derandomized Constructions of k-Wise (Almost) Independent Permutations. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX and RANDOM 2005. LNCS, vol. 3624, pp. 354–365. Springer, Heidelberg (2005)

[KORT01] Kueh, K., Olson, T., Rockmore, D., Tan, K.: Nonlinear approximation theory on compact groups. Journal of Fourier Analysis and Applications 7, 257–281 (2001)

[KP82] Karp, R.M., Papadimitriou, C.H.: On linear characterizations of combinatorial optimization problems. SIAM Journal on Computing 11(4), 620–632 (1982)

[Mas98] Maslen, D.: Efficient computation of fourier transforms on compact groups. Journal of Fourier Analysis and Applications 4, 19–52 (1998)

[RS09] Roy, A., Scott, A.J.: Unitary designs and codes. Des. Codes Cryptography 53, 13–31 (2009)

[RW06] Russell, A., Wang, H.: How to fool an unbounded adversary with a short key. IEEE Transactions on Information Theory 52(3), 1130–1140 (2006)

[RX10] Rubinfeld, R., Xie, N.: Testing Non-uniform k-Wise Independent Distributions over Product Spaces. In: Abramsky, S., Gavoille, C., Kirchner, C., Meyer auf der Heide, F., Spirakis, P.G. (eds.) ICALP 2010, Part I. LNCS, vol. 6198, pp. 565–581. Springer, Heidelberg (2010)

[Vau98] Vaudenay, S.: Provable Security for Block Ciphers by Decorrelation. In: Morvan, M., Meinel, C., Krob, D. (eds.) STACS 1998. LNCS, vol. 1373, pp. 249–275. Springer, Heidelberg (1998)

[Vau00] Vaudenay, S.: Adaptive-Attack Norm for Decorrelation and Super-Pseudorandomness. In: Heys, H.M., Adams, C.M. (eds.) SAC 1999. LNCS, vol. 1758, pp. 49–61. Springer, Heidelberg (2000)

[Vau03] Vaudenay, S.: Decorrelation: a theory for block cipher security. Journal of Cryptology 16(4), 249–286 (2003)

Testing Permanent Oracles – Revisited

Sanjeev Arora*, Arnab Bhattacharyya**, Rajsekar Manokaran*, and Sushant Sachdeva*

Department of Computer Science and Center for Computational Intractability, Princeton University, USA
{arora,arnabb,rajsekar,sachdeva}@cs.princeton.edu

Abstract. Suppose we are given an oracle that claims to approximate the permanent for most matrices X, where X is chosen from the *Gaussian ensemble* (the matrix entries are i.i.d. univariate complex Gaussians). Can we test that the oracle satisfies this claim? This paper gives a polynomial-time algorithm for the task.

The oracle-testing problem is of interest because a recent paper of Aaronson and Arkhipov showed that if there is a polynomial-time algorithm for simulating boson-boson interactions in quantum mechanics, then an approximation oracle for the permanent (of the type described above) exists in $\mathsf{BPP}^{\mathsf{NP}}$. Since computing the permanent of even 0/1 matrices is #P-complete, this seems to demonstrate more computational power in quantum mechanics than Shor's factoring algorithm does. However, unlike factoring, which is in NP, it was unclear previously how to test the correctness of an approximation oracle for the permanent, and this is the contribution of the paper.

The technical difficulty overcome here is that univariate polynomial self-correction, which underlies similar oracle-testing algorithms for permanent over *finite fields* —and whose discovery led to a revolution in complexity theory—does not seem to generalize to complex (or even, real) numbers. We believe that this tester will motivate further progress on understanding the permanent of Gaussian matrices.

1 Introduction

The permanent of an n-by-n matrix $X = (x_{i,j})$ is defined as

$$\mathsf{Per}(X) = \sum_{\pi} \prod_{i=1}^{n} x_{i,\pi(i)},$$

where π ranges over all permutations from $[n]$ to $[n]$. A recent paper of Aaronson and Arkhipov [1] (henceforth referred to as AA) introduced a surprising connection between quantum computing and the complexity of computing the permanent (which is well-known to be #P-complete to compute in the worst case [2]). They define and study a formal model of quantum computation with

* This work is supported by the NSF grants CCF-0832797 and CCF-1117309.
** This work is supported by NSF Grants CCF-0832797, 0830673, and 0528414.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 362–373, 2012.
© Springer-Verlag Berlin Heidelberg 2012

non-interacting bosons in which n bosons pass through a "circuit" consisting of optical elements. Each boson starts out in one of m different phases and, at the end of the experiment, is in a superposition of the basis states—one for each possible partition of the n bosons into m phases.

AA proceed to show that if there is an efficient classical randomized algorithm $\mathcal{A}$ that simulates the experiment, in the sense of being able to output random samples from the final distribution (up to a small error in total variation distance) of the Bosonic states at the end of the experiment, then there is a way to design an *approximation* algorithm $\mathcal{B}$ in $\mathsf{BPP}^{\mathsf{NP}}$ for the permanent problem for an interesting family of random matrices. The random matrices are drawn from the Gaussian ensemble—each entry is an independent standard Gaussian complex number—and the algorithm computes an additive approximation, in the sense that,

$$|\mathcal{B}(X) - \mathsf{Per}(X)|^2 \leqslant \delta^2 n!\,, \tag{1}$$

for at least a fraction $1 - \eta$ of the input matrices X. (Note that the *variance* of $\mathsf{Per}(X)$ is $n!$ for Gaussian ensembles, so this approximation is nontrivial.) The running time of $\mathcal{B}$ is $\mathrm{poly}(n, 1/\delta, 1/\eta)$ with access to an oracle in $\mathsf{NP}^{\mathcal{A}}$. In other words, $\mathcal{B} \in \mathsf{BPP}^{\mathsf{NP}^{\mathcal{A}}}$ for $\eta, \delta = \Omega\left(1/\mathrm{poly}(n)\right)$ (refer to Problem 2 and Theorem 3 in [1]). The authors go on to conjecture that obtaining an additive approximation as in eq. (1) is #P-hard (this follows from Conjectures 5 and 6, and Theorem 7 in [1]). If true, this conjecture has surprising implications for the computational power of quantum systems. By contrast, the crown jewel of quantum computing, Shor's algorithm [3], implies that the ability to simulate quantum systems would allow us to factor integers in polynomial time, but factoring (as well as other problems known to be in BQP) is not even known to be NP-Hard.

As evidence for their conjecture, Arkhipov and Aaronson point to related facts about the permanent problem for matrices over integers and finite fields. It is known that that if there is a constant factor approximation algorithm for computing $\mathsf{Per}(X)$ where X is an arbitrary matrix of integers, then one can solve #P problems in polynomial time. Thus, approximation on *all* inputs seems difficult[1]. Likewise, starting with a paper of Lipton, researchers have studied the complexity of computing the permanent (exactly) for *many matrices*. For example, given an algorithm that computes the permanent exactly for $1/\mathrm{poly}(n)$ fraction of all matrices X over a finite field $GF(p)$ (where p is a sufficiently large prime), one can use self-correction procedures for univariate polynomials [7,8,9] to again obtain efficient randomized algorithms for #P-hard problems.

Thus, either restriction —approximation on all matrices, or the ability to compute exactly on a significant fraction of matrices— individually results in a #P-hard problem. What makes the AA conjecture interesting is that it involves the conjunction of the two restrictions: the oracle in question *approximates* the value of the permanent for *most* matrices.

The focus of the current paper is the following question: *given an additive approximation oracle for permanents of Gaussian matrices ($\mathcal{B}$ in eq. (1) above),*

[1] Note that approximating the permanent is known to be feasible for the special case of non-negative real matrices [4,5,6].

how can we test that the oracle is correct? We want a tester that accepts with high probability when $\mathcal{B}$ satisfies the condition in eq. (1) and rejects with high probability when $\mathcal{B}$ does not approximate well on a substantial fraction of inputs. Note that the testing problem is a non-issue for previous quantum algorithms such as Shor's algorithm, since the correctness of a factoring algorithm is easy to test.

The testing question has been studied for the permanent problem over finite fields. Given an oracle that supposedly computes $\mathsf{Per}(\cdot)$ for even, say, $^3/_4{}^{\text{th}}$ of the matrices over $GF(p)$, one can verify this claim using self-correction for polynomials over finite fields and the *downward self-reducibility* of $\mathsf{Per}(\cdot)$, as described below in more detail in Section 1.1. (In fact, if the oracle satisfies the claim, then one can compute $\mathsf{Per}(\cdot)$ on all matrices with high probability.) However, as noted in AA, these techniques that work over finite fields fail badly over the complex numbers. The authors in AA also seem to suggest that techniques analogous to self-correction and downward self-reducibility can be generalized to complex numbers in some way, but this remains open.

In this paper, we solve the testing problem using downward self-reducibility alone. Perhaps this gives some weak evidence for the truth of the AA conjecture. Note that since we lack self-correction techniques, we do not get an oracle at the end that computes the permanent for all matrices as in the finite field case. Incidentally, an argument similar to the one presented in this paper works in the finite field case also, giving an alternate tester for the permanent that does not use self-correction of polynomials over finite fields.

1.1 Related Work

As mentioned above, testing an oracle for the permanent over finite fields has been extensively studied. The approach, basically arising from [10], uses self-correction of polynomials over finite fields and downward self-reducibility of the permanent. Let us revisit the argument.

Suppose we are given a sequence of oracles $\{\mathcal{O}_k\}_k$, where for each k, $\mathcal{O}_k$ allegedly computes the permanent for a 9/10 fraction of all k-by-k matrices over the field. The argument proceeds by first applying a self-correction procedure for low-degree polynomials (see [8]), noting that the permanent is a k-degree multilinear polynomial in the k^2 entries of the matrix, treated as variables.

The correction procedure, on input X, queries $\mathcal{O}_k$ at $\text{poly}(n)$ points, and outputs the correct value of $\mathsf{Per}(X)$ with $1-\exp(-n)$ probability (over the coin tosses of the procedure). Thus, the procedure acts as a proxy for the oracle, providing $\{\mathcal{O}_k^\star\}_k$ which can now be tested for mutual consistency using the downward self-reducibility of the permanent:

$$\mathsf{Per}(X) = \textstyle\sum_j x_{1,j} \cdot \mathsf{Per}(X_j). \tag{2}$$

Here, X_j is the submatrix formed by removing the first row and j^{th} column. Finally, since $\mathcal{O}_1$ can be verified by direct computation, this procedure tests and accepts sequences where $\mathcal{O}_k$ computes the permanent of a fraction 9/10

of all $k \times k$ matrices; while rejecting sequences of oracles where for some k, $\mathcal{O}_k(X) \neq \mathsf{Per}_k(X)$ on more than, say a fraction 3/10, of the inputs.

A natural attempt to port this argument to real/complex gaussian matrices runs into fatal issues with the self-correction procedures: since the oracles are only required to *approximate* the value of the permanent, a polynomial interpolation procedure incurs an exponential (in the degree) blow-up in the error at the point of interest (see [11]). In our work, we circumvent polynomial interpolation and only deal with self-reducibility, noting that eq. (2) expresses the permanent as a *linear* function of permanent of smaller matrices.

1.2 Overview of the Tester

We work with the following notion of quality of an oracle, naturally inspired by the AA conjecture: the approximation guarantee is achieved by the oracle on all but a small fraction of the inputs.

Definition 1. *For an integer n, an oracle $\mathcal{O}_n : \mathbb{C}^{n^2} \to \mathbb{C}$, is said to be (δ, η)-good if $|\mathcal{O}_n(X) - \mathsf{Per}(X)|^2 \leqslant \delta^2 n!$, with probability at least $1-\eta$ over the choice of $n \times n$ matrices, X, from the Gaussian ensemble.*

Note that since the tester is required to be efficient, we (necessarily) allow even good oracles to answer arbitrarily on a small fraction of inputs, because the tester will not encounter these bad inputs with high probability. As an aside, there is also the issue of *additive* vs *multiplicative* approximation, which AA conjecture have similar complexity. In this paper, we stick with additive approximation as defined above.

Our main result is stated informally below (see Theorem 2 for a precise statement).

Theorem 1 (Main theorem – informal). *There exists an algorithm $\mathcal{A}$ that, given a positive integer n, an error parameter[2] $\delta \geqslant 1/\mathrm{poly}(n)$, and access to oracles $\{\mathcal{O}_k\}_{1 \leqslant k \leqslant n}$ such that $\mathcal{O}_k : \mathbb{C}^{k^2} \to \mathbb{C}$, has the following behavior:*

- *If for every $k \leqslant n$, the oracle $\mathcal{O}_k$ is $(\delta, 1/\mathrm{poly}(n))$-good, then $\mathcal{A}$ accepts with probability at least $1 - 1/\mathrm{poly}(n)$.*
- *If there exists a $k \leqslant n$ such that the oracle $\mathcal{O}_k$ is not even $(\mathrm{poly}(n) \cdot \delta, 1/\mathrm{poly}(n))$-good, then $\mathcal{A}$ rejects with probability at least $1 - 1/\mathrm{poly}(n)$.*
- *The query complexity as well as the time complexity of $\mathcal{A}$ is $\mathrm{poly}(n/\delta)$.*

We conduct the test in n stages, one stage for each submatrix size. Let $k \leqslant n$ denote a fixed stage, and let $X \in \mathbb{C}^{k^2}$. Now, using downward self-reducibility (eq. (2)), we have,

$$|\mathcal{O}_k(X) - \mathsf{Per}_k(X)| \leqslant \underbrace{\left|\mathcal{O}_k(X) - \textstyle\sum_j x_j \mathcal{O}_{k-1}(X_j)\right|}_{\text{(A)}} + \underbrace{\left|\textstyle\sum_j x_j \left[\mathcal{O}_{k-1}(X_j) - \mathsf{Per}_{k-1}(X_j)\right]\right|}_{\text{(B)}}. \quad (3)$$

[2] All of the poly(·) are fixed polynomials, hidden for clarity.

Recall that X_j is the submatrix formed by removing the first row and j^{th} column (often referred to as a minor).

We bound term (A) above, by checking if $\mathcal{O}_k$ is a linear function in the variables along the first row (x_j in above), when the rest of the entries of the matrix are fixed; the coefficients of the linear function are determined by querying $\mathcal{O}_{k-1}$ on the k minors along the first row. The tolerance needed in the test is estimated as follows: a good collection of oracles estimates Per_{k-1} up to $\delta\sqrt{(k-1)!}$, and Per_k up to $\delta\sqrt{k!}$ additive error. Further, since the expression is identically zero for the permanent function, we have:

$$\begin{aligned}(\text{A}) &\leqslant |\mathcal{O}_k(X) - \mathsf{Per}_k(X)| + \left|\textstyle\sum_j x_j \left(\mathcal{O}_{k-1}(X_j) - \mathsf{Per}_{k-1}(X_k)\right)\right| \\ &\leqslant \delta\sqrt{k!} + \left|\textstyle\sum_j x_j \delta\sqrt{(k-1)!}\right| \leqslant \delta\sqrt{k!} \cdot (1 + O(\sqrt{\log n})),\end{aligned}$$

where the last inequality follows from standard Gaussian tail bounds.

We test this by simply querying the oracles for random X and the minors obtained thereof and checking if the downward self-reducibility condition is approximately met.

The second term, term (B), is linear in the error $\mathcal{O}_{k-1}$ makes on the minors, say $\varepsilon_{k-1}\sqrt{(k-1)!}$ on each minor. A naive argument as above says term (B) is at most $\varepsilon_{k-1}\sqrt{k!} \cdot \Theta(\sqrt{\log n})$. From this and eq. (3), the error in $\mathcal{O}_k$ is at most a $\Theta(\sqrt{\log n})$ factor times the error in $\mathcal{O}_{k-1}$. However, this bound is too weak to conclude anything useful about $\mathcal{O}_n$.

We overcome this issue by measuring the error in a root-mean-square (RMS or ℓ_2) sense as follows:

$$\mathsf{err}_2(\mathcal{O}_k) = \sqrt{\mathop{\mathbf{E}}_X \left[\mathcal{O}_k(X) - \mathsf{Per}_k(X)\right]^2} = \|\mathcal{O}_k - \mathsf{Per}_k\|_2.$$

Now,

$$\|\mathcal{O}_k - \mathsf{Per}_k\|_2 \leqslant \|\mathcal{O}_k - \textstyle\sum_j (x_j \mathcal{O}_{j-1}(X_j))\|_2 + \sqrt{\mathbf{E}\left[\textstyle\sum_j x_j (\mathcal{O}_{k-1} - \mathsf{Per}_{k-1})\right]^2}.$$

The first term is still $\delta\sqrt{k!} \cdot O(\sqrt{\log n})$ assuming the linearity test passes. Since each x_i is an independent standard Gaussian, the second term is at most $\sqrt{k} \cdot \mathsf{err}_2(\mathcal{O}_{k-1}) = \varepsilon_{k-1} \cdot \sqrt{k!}$. Then, $\mathsf{err}_2(\mathcal{O}_k) \leqslant (\delta\sqrt{\log n} + \varepsilon_{k-1}) \cdot \sqrt{k!}$, and thus $\mathsf{err}_2(\mathcal{O}_n)$ is at most $\mathrm{poly}(n)\delta\sqrt{n!}$ as we set out to prove! The caveat however is that err_2 as defined cannot be bounded precisely because we necessarily need to discount a small fraction of the inputs: the oracles could be returning arbitrary values on a small fraction, outside the purview of any efficient tester. We deal with this by using a more sophisticated RMS error that discounts an η-fraction of the input:

$$\mathsf{err}_{2,\eta}(\mathcal{O}_k) = \inf_{S:\mu(S)\leqslant\eta} \sqrt{\mathop{\mathbf{E}}_X \left[1_s(\mathcal{O}_k(X) - \mathsf{Per}_k(X))\right]^2},$$

where 1_S denotes the indicator function of the set S. We then use a tail inequality on the permanent based on its fourth moment to carry through the inductive argument set up above. This requires a *Tail Test* on the oracles to check that the oracles have a tail similar to the permanent. Our analysis shows that the Linearity and Tail test we design are sufficient and efficient, proving Theorem 1.

Organization. In the next section, we set up the notation. Section 3 describes the test we design and follows it up with its analysis. All missing proofs are deferred to the final version.

2 Preliminaries

Notation and Setup. We deal with complex valued functions on the space of square matrices over the complex numbers, $\mathbb{C}^{k\times k}$ for some integer k. We assume $\mathbb{C}^{k\times k}$ is endowed with the standard Gaussian measure $\mathcal{N}(0,1)^{k\times k}_{\mathbb{C}}$. We use the notation $\mathbf{P}_X[E]$ to denote the probability of an event E, when $X \sim \mathcal{N}(0,1)^{k\times k}_{\mathbb{C}}$. We denote by $\mathbf{E}_X[Y]$ to denote the expectation of the random variable Y, when $X \sim \mathcal{N}(0,1)^{k\times k}_{\mathbb{C}}$.

Functions from $\mathbb{C}^d$ to $\{0,1\}$ are called indicator functions (since they indicate inclusion in the set of points where the function's value is 1). We denote the indicator function for a predicate $q(X)$ by $\mathbf{I}[q(X)]$ and define it to be 1 when $q(X)$ is true and 0 otherwise. For example, $\mathbf{I}[|x| \geqslant 2]$ is 1 for all x whose magnitude is at least 2, and 0 otherwise.

Error and ℓ_2 norm of Oracles. The (standard) ℓ_2 norm of a square-integrable function $f : \mathbb{C}^d \to \mathbb{C}$ is denoted by $\|f\|_2$ and is equal to $\mathbf{E}_X[|f|^2]$. An oracle for the permanent is simply a function $\mathcal{O}_k : \mathbb{C}^{k\times k} \to \mathbb{C}$ that can be queried in a single time unit. We will work with a sequence of oracles $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$, one for every dimension k less than n.

Moments of Permanents. The first and the second moments of the permanent under the Gaussian distribution on $k \times k$ matrices are easy to compute: $\mathbf{E}_X[\mathsf{Per}_k(X)] = 0$, $\mathbf{E}_X[|\mathsf{Per}_k(X)|^2] = k!$. We also know the fourth moment of the permanent function for Gaussian matrices, $\mathbf{E}_X[|\mathsf{Per}_k(X)|^4] = (k+1)(k!)^2$ (Lemma 56, [1]). This fact and Markov's inequality immediately imply:

Lemma 1 (Tail Bound for Permanent). *For every positive integer k, the permanent satisfies* $\mathbf{P}_X[|\mathsf{Per}_k(X)| > T\sqrt{k!}] \leqslant {}^{(k+1)}/_{T^4}$.

3 Testing Approximate Permanent Oracles

Our testing procedure, PTest, has three parameters: a positive integer n, the dimension of the matrices being tested; $\delta \in (0,1]$, the amount of error allowed; and $c \in (0,1]$, a completeness parameter. In addition, it has query access to the sequence of oracles, $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$ being tested. In the following, for a matrix X, we denote the entries in the first row of X by $x_{11}, \ldots, x_{1k}$, and by X_i the minor obtained by removing the first row and the i^{th} column from X. (There will be no confusion since we will only be working with expansion along the first row.)

The guarantees of the tester are twofold: it accepts with probability at least $1-c$ if $|\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2 \leqslant \delta^2 k!$ for every k, and every $X \in \mathbb{C}^{k\times k}$; on the other hand, the tester almost always rejects if for some $k \leqslant n$, $\mathcal{O}_k(X)$ is not $\mathrm{poly}(n)\delta\cdot\sqrt{k!}$ close to $\mathsf{Per}_k(X)$ for at least $1-\frac{1}{\mathrm{poly}(n)}$ measure of X's (see below for precise theorems). The query complexity of PTest is bounded by $\mathrm{poly}(n, 1/\delta, 1/c)$. Assuming that each oracle query takes constant time, the time complexity of PTest is also bounded by $\mathrm{poly}(n, 1/\delta, 1/c)$ (see below for precise bounds).

The test consists of two parts: The first is a *Linearity* test, that tests that the oracles $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$ satisfy $\mathcal{O}_k(X) \approx \sum_i x_{1i}\mathcal{O}_{k-1}(X_i)$ (observe that the permanent satisfies this exactly). The second part is a *Tail* test, that tests that the function does not take large values too often (the permanent satisfies this property too, as shown by Lemma 1).

LinearityTest(n, k, δ): Sample a $k \times k$ matrix $X \sim \mathcal{N}(0,1)^{k\times k}_{\mathbb{C}}$. If $k = 1$, output `Reject` unless $|\mathcal{O}_k(X) - X|^2 \leqslant n^2 \cdot \delta^2$. Else, test if: $\left|\mathcal{O}_k(X) - \sum_{i=1}^{k} x_{1i}\mathcal{O}_{k-1}(X_i)\right|^2 \leqslant n^2\delta^2 \cdot k!$. Output `Reject` if it does not hold.

TailTest(k, T): Sample a $k \times k$ matrix X. Test that $|\mathcal{O}_k(X)|^2 \leqslant T^2 k!$. Output `Reject` if it does not hold.

Parameters: A positive integer $n \in \mathbb{N}$, error parameter $\delta \in (0, 1]$, and completeness parameter $c \in (0, 1]$.
Requires: Oracle access to $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$, where $\mathcal{O}_k : \mathbb{C}^{k\times k} \to \mathbb{C}$.

1. Set the following variables: $T = 4n/\delta\sqrt{c}$, $d = 192n^2/\delta^4 c$.
2. For each $1 \leqslant k \leqslant n$,
   (a) Run LinearityTest(n, k, δ) d times.
   (b) Run TailTest(k, T) d times.
3. If none of the above tests output `Reject`, output `Accept`.

Fig. 1. The tester PTest

The procedure PTest is formally defined in Figure 1. In the rest of the paper, we prove the following theorem about PTest.

Theorem 2 (Main Theorem). *For all $n \in \mathbb{N}, \delta \in (0, 1]$, and $c \in (0, 1]$, satisfying $n = \Omega\left(\sqrt{\log \frac{1}{c\delta}}\right)$, given oracle access to $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$, where $\mathcal{O}_k : \mathbb{C}^{k\times k} \to \mathbb{C}$, the procedure* PTest *satisfies the following:*

1. **(Completeness).** *If, for every $k \leqslant n$, and every $X \in \mathbb{C}^{k\times k}$, $|\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2 \leqslant \delta^2 k!$, then* PTest *accepts with probability at least $1-c$.*
2. **(Soundness).** *For every $1 \leqslant k \leqslant n$, either*

There exists an indicator function $1_k : \mathbb{C}^{k\times k} \to \{0,1\}$ *satisfying* $\mathbf{E}_X[1_k(X)] \geqslant 1 - \frac{\delta^4 c}{64n}$, *such that,* $\mathbf{E}_X[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2] \leqslant (2nk\delta)^2 k!$.

or else,

PTest *outputs* `Reject` *with probability at least* $1 - e^{-n}$.

3. **(Complexity)** *The total number of queries made by* PTest *is* $O(n^4\delta^{-4}c^{-1})$. *Moreover, assuming that each oracle query takes constant time, the time required by* PTest *is also* $O(n^4\delta^{-4}c^{-1})$.

The completeness and soundness are proved below as Theorem 3, Theorem 4 in Sections 3.1, 3.2 respectively. The complexity of the test is immediate from the definitions.

Remark 1. Observe that, assuming both $1/c$ and $1/\delta$ are polynomial in n, the query complexity is poly(n), and hence, even if the oracles $\{\mathcal{O}_k\}_{k\leqslant n}$ satisfy $|\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2 \leqslant \delta^2 k!$ only with probability $1 - \frac{1}{\mathrm{poly}(n)}$, PTest would still accept with probability $1 - c - \frac{1}{\mathrm{poly}(n)}$.

Remark 2. Observe that the (informal) main theorem (Theorem 1) stated in the introduction follows from Theorem 2 from a simple Markov argument. Given $\delta = \Omega(1/\mathrm{poly}(n))$, set $c = \frac{1}{\mathrm{poly}(n)}$ and note that the completeness follows directly from Theorem 2 and the previous remark. Further, from the *Soundness* claim of Theorem 2, we have an indicator function $1_k : \mathbb{C}^{k\times k} \to \{0,1\}$ satisfying $\mathbf{E}_X[1_k(X)] \geqslant 1 - \frac{\delta^4 c}{64n} \geqslant 1 - \frac{1}{\mathrm{poly}(n)}$, such that, $\mathbf{E}_X[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2] \leqslant (2nk\delta)^2 k! \leqslant \mathrm{poly}(n) \cdot \delta^2 k!$. Applying Markov's inequality, we have that $\mathbf{P}\left[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k|^2 \geqslant \mathrm{poly}(n)\delta^2 k!\right] \leqslant 1/\mathrm{poly}(n)$. Now, note that 1_k is an indicator function, and $\mathbf{P}[1_k(X) = 0]$ is at most $1/\mathrm{poly}(n)$. This, along with the previous expression gives that the tester outputs `Reject` if the sequence of oracles is not even $(\mathrm{poly}(n) \cdot \delta, 1/\mathrm{poly}(n))$-good.

3.1 Completeness

We first prove the completeness of PTest: that a $(\delta, 0)$-good sequence of oracles is accepted with probability at least $1 - c$.

Theorem 3 (Completeness). *If, for every* $k \leqslant n$, *and every* $X \in \mathbb{C}^{k\times k}$, $|\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2 \leqslant \delta^2 k!$, *then* PTest *accepts with probability at least* $1 - c$.

Proof. Suppose we are given a sequence of oracles $\{\mathcal{O}_k\}_{k\leqslant n}$ such that for all $k \leqslant n$, we have that $|\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2 \leqslant \delta^2 \cdot k!$. Let X denote a randomly sampled $k \times k$ matrix.

We first bound the probability that the oracles $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$ fail a LinearityTest. For $k = 1$, it is easy to see that LinearityTest$(n, 1, \delta)$ never outputs `Reject` upon querying $\mathcal{O}_1$. For larger k, we have the following lemma that shows that $\mathcal{O}_k(X) \approx \sum_i x_{1i}\mathcal{O}_{k-1}(X_i)$, and hence LinearityTest outputs `Reject` only with small probability. We defer its proof to the full version.

Lemma 2 (Completeness for LinearityTest). *For every* $2 \leqslant k \leqslant n$, *the oracles* $\{\mathcal{O}_k\}_{\{k\leqslant n\}}$ *satisfy* $\mathbf{P}_X[|\mathcal{O}_k(X) - \sum_i x_{1i}\mathcal{O}_{k-1}(X_i)|^2 > n^2\delta^2 k!] \leqslant 2e^{-\frac{(n-1)^2}{2}}$.

This lemma implies that every call to $\mathsf{LinearityTest}(n, k, \delta)$ outputs `Reject` with probability at most $2e^{-\frac{(n-1)^2}{2}}$.

Next, we bound the probability that the oracles $\{\mathcal{O}_k\}_{\{k \leqslant n\}}$ fail a $\mathsf{TailTest}$. Using the tail bound for the permanent given by Lemma 1, we get, $\mathbf{P}_X[|\mathsf{Per}_k(X)| > (T-\delta)\sqrt{k!}] \leqslant {}^{(k+1)}/_{(T-\delta)^4}$. Since $|\mathcal{O}_k(X) - \mathsf{Per}_k(X)| \leqslant \delta \cdot \sqrt{k!}$, we use it in the above bound to get $\mathbf{P}_X[|\mathcal{O}_k(X)| > T\sqrt{k!}] \leqslant {}^{(k+1)}/_{(T-\delta)^4}$. Thus, every call to $\mathsf{TailTest}$ fails with probability at most $\frac{(n+1)}{(T-\delta)^4}$.

Now applying a union bound, we get that for n that is $\Omega\left(\sqrt{\log \frac{1}{\delta c}}\right)$, PTest outputs `Reject` with probability at most

$$(2e^{-\frac{(n-1)^2}{2}} + {}^{(n+1)}/_{(T-\delta)^4})dn \leqslant 384n^3/\delta^4 c \cdot e^{-(n-1)^2/2} + {}^{192(n+1)n^3 c}/_{(4n-\delta^2\sqrt{c})^4} \leqslant c.$$

□

3.2 Soundness

The interesting part of the analysis is the soundness for PTest, which we prove in this section. Given $\{\mathcal{O}_k\}_{\{k \leqslant n\}}$, we need to define the following indicator functions to aid our analysis:

$$1_k^{LIN}(X) = \begin{cases} \mathbf{I}[(\mathcal{O}_k(X) - X)^2 \leqslant n^2\delta^2], & \text{if } k = 1 \\ \mathbf{I}[(\mathcal{O}_k(X) - \sum_i x_{1i}\mathcal{O}_{k-1}(X_i))^2 \leqslant n^2\delta^2 k!\,], & \text{if } 2 \leqslant k \leqslant n \end{cases}$$

$$1_k^{TAIL}(X) = \mathbf{I}[\mathcal{O}_k(X)^2 \leqslant T^2 \cdot k!\,],$$

$$1_k^{PERM}(X) = \mathbf{I}[\mathsf{Per}_k(X)^2 \leqslant T^2 \cdot k!\,],$$

$$1_k(X) = 1_k^{LIN}(X) \wedge 1_k^{TAIL}(X) \wedge 1_k^{PERM}(X). \tag{4}$$

We now prove the following theorem.

Theorem 4 (Soundness). *Let the indicator function 1_k be as defined by Equation (4). For every $k \leqslant n$, either both of the following two conditions hold:*

1. *The indicator 1_k satisfies $\mathbf{E}_X[1_k(X)] \geqslant 1 - \frac{\delta^4 c}{64n}$.*
2. *The oracle $\mathcal{O}_k$ and the indicator 1_k satisfy $\mathbf{E}_X[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2] \leqslant (2nk\delta)^2 k!$,*

or else, PTest outputs `Reject` with probability at least $1 - e^{-n}$.

Proof. We first prove the following lemma that shows that for all $k \leqslant n$, the expectation of 1_k is large.

Lemma 3 (Large Expectation of 1_k). *Either, for every k, the indicator function 1_k satisfies $\mathbf{E}_X[1_k(X)] \geqslant 1 - \frac{\delta^4 c}{64n}$, or else, PTest outputs `Reject` with probability at least $1 - e^{-n}$.*

The first part of the theorem follows immediately from this lemma. The proof of this lemma is given later in this section.

For the second part of the theorem, we prove the following inductive claim about the oracles $\{\mathcal{O}_k\}$.

Lemma 4. *(Main Induction Lemma) If for some* $2 \leqslant k \leqslant n$, *we have*

$$\mathop{\mathbf{E}}_{X \in \mathbb{C}^{(k-1)\times(k-1)}}[1_{k-1}(X) \cdot |\mathcal{O}_{k-1}(X) - \mathsf{Per}_{k-1}(X)|^2] \leqslant \varepsilon_{k-1}^2 (k-1)!\,,$$

then, either $\mathbf{E}_{X \in \mathbb{C}^{k\times k}}[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2] \leqslant (\varepsilon_{k-1} + 2n\delta)^2 k!$, *or else,* PTest *outputs* `Reject` *with probability at least* $1 - e^{-n}$.

The proof of this lemma is also presented later in the section. Assuming this lemma, we can complete the proof of soundness for PTest.

For the second part of the theorem, we first show that the required bound holds for $k = 1$. We know that for any $X \in \mathbb{C}$, whenever $1_1(X) = 1$, we have $|\mathcal{O}_1(X) - X|^2 \leqslant n^2\delta^2$. Thus,

$$\mathop{\mathbf{E}}_{X}[1_1(X) \cdot |\mathcal{O}_1(X) - \mathsf{Per}_1(X)|^2] \leqslant \mathop{\mathbf{E}}_{X}[1_1^{LIN}(X) \cdot |\mathcal{O}_1(X) - X|^2] \leqslant n^2\delta^2 < (2n\delta)^2 \cdot 1!\,.$$

This gives us our base case. Assume that there is a $2 \leqslant j \leqslant n$ such that,

$$\mathop{\mathbf{E}}_{X \in \mathbb{C}^{(j-1)\times(j-1)}}[1_{j-1}(X) \cdot |\mathcal{O}_{j-1}(X) - \mathsf{Per}_{j-1}(X)|^2] \leqslant (2n(j-1)\delta)^2 \cdot (j-1)!\,.$$

Now, we use Lemma 4 to deduce that either, $\mathbf{E}_{X \in \mathbb{C}^{j\times j}}[1_j(X) \cdot |\mathcal{O}_j(X) - \mathsf{Per}_j(X)|^2] \leqslant (2nj\delta)^2 \cdot j!$, or else, PTest outputs `Reject` with probability at least $1 - e^{-n}$. Thus, by induction, either for every $k \leqslant n$, $\mathbf{E}_X[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2] \leqslant (2nk\delta)^2 \cdot k!$, or else, PTest outputs `Reject` with probability at least $1 - e^{-n}$. This completes the proof of the theorem. □

Large expectation of 1_k. We now prove Lemma 3.

Proof. (*of Lemma 3*). We begin by making several claims about the structure the oracles $\{\mathcal{O}_k\}_{\{k \leqslant n\}}$ must have with high probability, assuming that PTest accepts. First, we claim that $\mathcal{O}_1$ must be close to the identity function.

Claim (Soundness of LinearityTest *for* $\mathcal{O}_1$*).* Either the oracle $\mathcal{O}_1$ satisfies that $\mathbf{P}_X\left[|\mathcal{O}_1(X) - X|^2 > n^2\delta^2\right] \leqslant n/d$, or else, PTest outputs ***`Reject`*** with probability at least $1 - e^{-n}$.

The straightforward proof of this claim is omitted. We also need the following two claims stating that for every $2 \leqslant k \leqslant n$, $\mathcal{O}_k(X) \approx \sum_i x_{1i}\mathcal{O}_{k-1}(X_i)$ often and that $\mathcal{O}_k(X)$ does not take large values often.

Claim (Soundness of LinearityTest*).* Either the oracles $\{\mathcal{O}_k\}$ satisfy the inequality $\mathbf{P}_X\left[|\mathcal{O}_k(X) - \sum_i x_{1i}\mathcal{O}_{k-1}(X)|^2 > n^2\delta^2 k!\right] \leqslant n/d$ for every $2 \leqslant k \leqslant n$, or else, PTest outputs ***`Reject`*** with probability at least $1 - e^{-n}$.

Claim (Soundness of TailTest*).* Either the oracles $\{\mathcal{O}_k\}$ satisfy the following for every $k \leqslant n$, $\mathbf{P}_X\left[|\mathcal{O}_k(X)|^2 > T^2 \cdot k!\right] \leqslant n/d$, or else, PTest outputs ***`Reject`*** with probability at least $1 - e^{-n}$.

The proofs of these claims are very similar to that of the first Claim for soundness of LinearityTest for $\mathcal{O}_1$ and are omitted here. We can restate the above claims in terms of 1_k^{LIN} and 1_k^{TAIL} defined in (4) as follows: Either, for every $k \leqslant n$,

$$\mathop{\mathbf{E}}_{X}[1_k^{LIN}(X)] \geqslant 1 - n/d,\ \mathop{\mathbf{E}}_{X}[1_k^{TAIL}(X)] \geqslant 1 - n/d, \tag{5}$$

or else, PTest will output `Reject` with probability at least $1 - e^{-n}$.

From Lemma 1, we know that $\mathbf{P}_X[|\mathsf{Per}_k(X)|^2 > T^2 \cdot k!\,] \leqslant (k+1)/T^4$. Again, this implies that $\mathbf{E}_X[1_k^{PERM}] \geqslant 1 - (k+1)/T^4$.

We are now ready to prove our lemma. We know that $1_k = 1_k^{LIN} \wedge 1_k^{TAIL} \wedge 1_k^{PERM}$. We know that if either of the claims in Equation (5) does not hold, PTest outputs `Reject` with probability at least $1 - e^{-n}$. Thus, we assume that both the claims in Equation (5) hold and get that for large enough n,

$$\begin{aligned}\mathop{\mathbf{E}}_{X}[1_k(X)] &\geqslant 1 - \mathop{\mathbf{E}}_{X}[1 - 1_k^{LIN}(X)] - \mathop{\mathbf{E}}_{X}[1 - 1_k^{TAIL}(X)] - \mathop{\mathbf{E}}_{X}[1 - 1_k^{PERM}(X)] \\ &\geqslant 1 - n/d - n/d - k+1/T^4 \geqslant 1 - \delta^4 c/96n - (n+1)\delta^4 c^2/256n^4 \geqslant 1 - \delta^4 c/64n. \qquad \square\end{aligned}$$

Main Induction Lemma. We now give a proof of the main induction lemma.

Proof. (*of Lemma 4*). Recall that X_i is the minor obtained by deleting the first row and the i^{th} column from X. We first split the probability space for $X \in \mathbb{C}^{k\times k}$ according to whether all of its minors X_i satisfy $1_{k-1}(X_i) = 1$ or not.

$$\|1_k(X)(\mathcal{O}_k(X) - \mathsf{Per}_k(X))\|^2 = \overbrace{\|1_k(X) \cdot \textstyle\prod_i 1_{k-1}(X_i)(\mathcal{O}_k(X) - \mathsf{Per}_k(X))\|^2}^{\text{(C)}} + \underbrace{\|1_k(X)(1 - \textstyle\prod_i 1_{k-1}(X_i))(\mathcal{O}_k(X) - \mathsf{Per}_k(X))\|^2}_{\text{(D)}}$$

Let $\tilde{1}_k(X) = 1_k(X)\prod_i 1_{k-1}(X_i)$. Term (C), above, is bounded by adding and subtracting the expression $\sum_i x_{1i}\mathcal{O}_{k-1}(X_i)$ and then expanding the permanent along the first row.

$$\begin{aligned}\|\tilde{1}_k(X)(\mathcal{O}_k(X) - \mathsf{Per}_k(X))\| &\leqslant \|\tilde{1}_k(X)[\mathcal{O}_k(X) - \textstyle\sum_i x_{1i}\mathcal{O}_{k-1}(X_i)]\| \\ &\quad + \underbrace{\|\tilde{1}_k(X)[\textstyle\sum_i x_{1i}\mathcal{O}_{k-1}(X_i) - \sum_i x_{1i}\mathsf{Per}_{k-1}(X_i)]\|}_{\text{(E)}}\end{aligned} \tag{6}$$

We know that if $1_k(X) = 1$, then $|\mathcal{O}_k(X) - \sum_i x_{1i}\mathcal{O}_{k-1}(X_i)|^2$ is bounded by $n^2\delta^2 k!$. Thus, the first term in eq. (6) is at most $n^2\delta^2 k!$. As for Term (E):

$$\begin{aligned}\text{(E)} &= \left\|1_k(X) \cdot \textstyle\prod_i 1_{k-1}(X_i)\left[\sum_i \mathcal{O}_{k-1}(X_i) - \sum_i x_{1i}\mathsf{Per}_{k-1}(X_i)\right]\right\|^2 \\ &\leqslant \mathop{\mathbf{E}}_{X_1,\ldots X_k}\ \mathop{\mathbf{E}}_{x_{11},\ldots,x_{1k}}[\textstyle\prod_i 1_{k-1}(X_i) \cdot |\sum_i x_{1i}\mathcal{O}_{k-1}(X_i) - \sum_i x_{1i}\mathsf{Per}_{k-1}(X_i)|^2] \\ &\leqslant \mathop{\mathbf{E}}_{X_1,\ldots X_k}[\textstyle\prod_i 1_{k-1}(X_i) \cdot \sum_i |\mathcal{O}_{k-1}(X_i) - \mathsf{Per}_{k-1}(X_i)|^2] \\ &\leqslant \textstyle\sum_i \mathop{\mathbf{E}}_{X_i}\left[1_{k-1}(X_i) \cdot |\mathcal{O}_{k-1}(X_i) - \mathsf{Per}_{k-1}(X_i)|^2\right] \leqslant k\varepsilon_{k-1}^2(k-1)! = \varepsilon_{k-1}^2 k!\end{aligned}$$

Combining the bounds on the two terms of eq. (6), we get,

$$\text{(C)} = \mathop{\mathbf{E}}_{X}[1_k(X) \cdot \textstyle\prod_i 1_{k-1}(X_i) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2] \leqslant (\varepsilon_{k-1} + n\delta)^2 \cdot k!\,. \tag{7}$$

Next, we bound term (D) as follows. First use lemma 3 to deduce $\mathbf{P}_X[1_{k-1}(X_i) = 0] \leqslant \frac{\delta^4 c}{64n}$ (If it does not hold, we know that PTest outputs Reject with probability at least $1 - e^{-n}$). Since whenever $1_k(X) = 1$, we have $|\mathcal{O}_k(X)| \leqslant T\sqrt{k!}$ and $|\mathsf{Per}_k(X)| \leqslant T\sqrt{k!}$. This implies that $1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2 \leqslant 4T^2 k!$ everywhere. Thus, we have,

$$\begin{aligned}(D) &= \|1_k(X)(1 - \textstyle\prod_i 1_{k-1}(X_i))(\mathcal{O}_k(X) - \mathsf{Per}_k(X))\|^2 \\ &\leqslant 4T^2 k! \mathop{\mathbf{E}}_X[1 - \textstyle\prod_i 1_{k-1}(X_i)] \\ &\leqslant 4T^2 k! \mathop{\mathbf{E}}_X[\textstyle\sum_i (1 - 1_{k-1}(X_i))] \leqslant 4T^2 k! \cdot k \cdot \delta^4 c/64n \leqslant n^2\delta^2 \cdot k!\,. \end{aligned} \tag{8}$$

Combining eqs. (6) to (8) completes the proof:

$$\mathbf{E}\left[1_k(X) \cdot |\mathcal{O}_k(X) - \mathsf{Per}_k(X)|^2\right] \leqslant \left((\varepsilon_{k-1} + n\delta)^2 + n^2\delta^2\right) \cdot k! \leqslant (\varepsilon_{k-1} + 2n\delta)^2 \cdot k!\,.$$

Acknowledgements. The authors would like to thank Madhur Tulsiani and Rishi Saket for extensive discussions during early stages of this work. We would also like to thank Scott Aaronson, Alex Arkhipov, Swastik Kopparty and Srikanth Srinivasan for helpful discussions.

References

1. Aaronson, S., Arkhipov, A.: The computational complexity of linear optics. In: Fortnow, L., Vadhan, S.P. (eds.) Proc. 43rd Annual ACM Symposium on the Theory of Computing, pp. 333–342. ACM (2011)
2. Valiant, L.G.: The complexity of computing the permanent. Theor. Comp. Sci. 8(2), 189–201 (1979)
3. Shor, P.W.: Algorithms for quantum computation: Discrete logarithms and factoring. In: Proc. 35th Annual IEEE Symposium on Foundations of Computer Science, pp. 124–134. IEEE Computer Society (1994)
4. Broder, A.Z.: How hard is to marry at random (on the approximation of the permanent). In: Hartmanis, J. (ed.) Proc. 18th Annual ACM Symposium on the Theory of Computing, pp. 50–58. ACM (1986)
5. Jerrum, M., Sinclair, A.: Approximating the permanent. SIAM J. on Comput. 18(6), 1149–1178 (1989)
6. Jerrum, M., Sinclair, A., Vigoda, E.: A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. J. ACM 51(4), 671–697 (2004)
7. Gemmell, P., Lipton, R.J., Rubinfeld, R., Sudan, M., Wigderson, A.: Self-testing/correcting for polynomials and for approximate functions. In: Koutsougeras, C., Vitter, J.S. (eds.) Proc. 23rd Annual ACM Symposium on the Theory of Computing, pp. 32–42. ACM (1991)
8. Gemmell, P., Sudan, M.: Highly resilient correctors for polynomials. Inform. Process. Lett. 43(4), 169–174 (1992)
9. Cai, J.-Y., Pavan, A., Sivakumar, D.: On the Hardness of Permanent. In: Meinel, C., Tison, S. (eds.) STACS 1999. LNCS, vol. 1563, pp. 90–99. Springer, Heidelberg (1999)
10. Lund, C., Fortnow, L., Karloff, H., Nisan, N.: Algebraic methods for interactive proof systems. J. ACM 39(4), 859–868 (1992)
11. Arora, S., Khot, S.: Fitting algebraic curves to noisy data. J. Comp. Sys. Sci. 67(2), 325–340 (2003)

Limitations of Local Filters of Lipschitz and Monotone Functions*

Pranjal Awasthi[1], Madhav Jha[2], Marco Molinaro[1], and Sofya Raskhodnikova[2,**]

[1] Carnegie Mellon University, USA
{pawasthi,molinaro}@cmu.edu
[2] Pennsylvania State University, USA
{mxj201,sofya}@cse.psu.edu

Abstract. We study local filters for two properties of functions $f : \{0,1\}^d \to \mathbb{R}$: the Lipschitz property and monotonicity. A local filter with additive error a is a randomized algorithm that is given black-box access to a function f and a query point x in the domain of f. Its output is a value $F(x)$, such that (i) the *reconstructed function* $F(x)$ satisfies the property (in our case, is Lipschitz or monotone) and (ii) if the input function f satisfies the property, then for every point x in the domain (with high constant probability) the reconstructed value $F(x)$ differs from $f(x)$ by at most a. Local filters were introduced by Saks and Seshadhri (SICOMP 2010) and the relaxed definition we study is due to Bhattacharyya *et al.* (RANDOM 2010), except that we further relax it by allowing additive error. Local filters for Lipschitz and monotone functions have applications to areas such as data privacy.

We show that every local filter for Lipschitz or monotone functions runs in time exponential in the dimension d, even when the filter is allowed significant additive error. Prior lower bounds (for local filters with no additive error, i.e., with $a = 0$) applied only to more restrictive class of filters, e.g., *nonadaptive* filters. To prove our lower bounds, we construct families of hard functions and show that lookups of a local filter on these functions are captured by a combinatorial object that we call a c-connector. Then we present a lower bound on the maximum outdegree of a c-connector, and show that it implies the desired bounds on the running time of local filters. Our lower bounds, in particular, imply the same bound on the running time for a class of privacy mechanisms.

1 Introduction

In this work we study local reconstruction of properties of functions. Property-preserving data reconstruction [1] is a direction of research in sublinear algorithms that has its roots in property testing [14, 10]. Some related notions include locally decodable codes [12], program checking [7] and, more generally, local computation [15, 2].

To motivate the reconstruction model, consider an algorithm ALG that is computing on a large dataset and whose correctness is contingent upon the dataset satisfying a

* All omitted proofs appear in the full version [3].

** P.A. is supported by NSF grant CCF-1116892. M.J. and S.R. are supported by NSF CAREER grant CCF-0845701 and NSF grant CCF-0729171. M.M. is supported by NSF grant CMMI-1024554.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 374–386, 2012.
© Springer-Verlag Berlin Heidelberg 2012

certain structural property. For example, ALG may require that its input array be sorted or that its input function be Lipschitz. In such situations, ALG could access its input via a *filter* that ensures that data seen by ALG always satisfies the desired property, modifying it at few places on the fly, if required. We can represent the input to ALG as a function f, where $f(x)$ represents the portion of the data that can be accessed on query x. Instead of accessing $f(x)$ directly, ALG makes a *query* x to the filter. The filter *looks up* the value of f on a small number of points and returns $F(x)$, where F satisfies the desired property and is as close to the original function f as possible. Thus, ALG is computing with reconstructed data F instead of its original input f.

Saks and Seshadhri [16] introduced the stronger notion of a *local filter*. It has an additional requirement that the reconstruction of $f(x)$ and $f(y)$ on two different queries x and y should be done independently. In particular, the output function F is independent of the order of the queries x made to the filter.

Local filters have many desirable features: for example, they can be implemented in a distributed setting, where several processes need to access different parts of the input, and the filter has to ensure that all the parts together are consistent with some function F that satisfies the desired property. This global consistency guarantee enables several applications of local filters described in previous work [16, 5, 11], including the application to data privacy that we explain below.

The main goal of this paper is to understand limitations of local filters. This is crucial in order to identify the types of tradeoffs (i.e., output quality vs. lookup complexity) available for a given application. Two natural candidate properties for this evaluation are the Lipschitz property and monotonicity of functions[1] $f : [n]^d \to \mathbb{R}$, studied in previous work [1, 16, 5, 11]: the first is motivated by the privacy application explained below and the second is a 'benchmark' problem in property-preserving reconstruction and property testing. A function $f : [n]^d \to \mathbb{R}$ is *Lipschitz* (with respect to the ℓ_1 metric on $[n]^d$) if $|f(x) - f(y)| \leq \|x - y\|_1$ for all points x, y in the domain $[n]^d$. Intuitively, changing the argument to the Lipschitz function by a small amount does not significantly change the value of the function. A function $f : [n]^d \to \mathbb{R}$ is *monotone* if $f(x) \leq f(y)$ for all points $x \preceq y$ in the domain $[n]^d$, where $\preceq$ denotes the natural partial order on $[n]^d$: for $x = (x_1, \ldots, x_d) \in [n]^d$ and $y = (y_1, \ldots, y_d) \in [n]^d$, we have $x \preceq y$ iff $x_i \leq y_i$ for all coordinates $i \in [d]$. In other words, increasing the coordinates of the argument to a monotone function does not decrease the value of the function.

The original definition of local filters in [16] has a requirement that the filter be *distance-respecting*, that is, the reconstructed function F should not differ from the original function f on significantly more points than necessary. Bhattacharyya *et al.* [5] and Jha and Raskhodnikova [11] removed this requirement and demonstrated that it is not necessary in some applications. Their local filter is simply required to output $F = f$ if the original function has the property; otherwise, F can be an arbitrary function satisfying the property. We relax the notion of local filter further by allowing additive error. Our definition (see Definition 2.1) has an additional parameter a, and the function F can differ from f by a small amount on every point, even if f satisfies the property: namely, we require that for every x in the domain, with high constant probability $|F(x) - f(x)| \leq a$. Local filters considered in [5, 11] are a special case of our

[1] We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$.

local filters with $a = 0$. Our goal is to determine (for small a) if there are local filters that make only $\mathrm{poly}(n, d)$ lookups in order to output the reconstructed function $F(x)$ at a given point x.

Privacy Application. We observe that local filters with small additive error can still be used in the privacy application described in [11]. Consider a server which has a private database with information about individuals, modeled as a point x in $\{0,1\}^d$, representing which of d possible types of people are present in the database. (More generally, x is modeled as a point in $[n]^d$ representing a histogram that captures how many people of each type are present.) A user who does not have direct access to x can ask the server for some information about this database by specifying a function f for the server to evaluate at the point x. The server's goal is to output a value which is close to $f(x)$ but which reveals almost no information about any single individual. Recently, the latter notion has been made precise via the concept of *differential privacy* [9]. A standard way of obtaining such guarantees is to ask users to *submit only Lipschitz* functions[2], and have the server output $f(x)$ plus some random noise depending on the desired privacy guarantee [9]. However, if a malicious user submits a function which is not Lipschitz, the differential privacy guarantee is lost. A local filter with the following properties can then be used between the server and the submitted function f to ensure the desired privacy: (i) the reconstructed function F is always Lipschitz; (ii) if f is already Lipschitz, then with high probability $|F(x) - f(x)| \leq a$ for all x, where a is a given parameter. This way, the server always evaluates a Lipschitz function F and thus has the desired privacy guarantees. Furthermore, if the user provides a valid Lipschitz function f, the mechanism outputs a value $F(x)$ in the range $f(x) \pm a$ plus a random noise; if a is reasonably small it is then absorbed in the noise. Thus, bounds on the running time and additive error of the local filter translate directly into bounds on the running time and accuracy of the corresponding privacy mechanism.

1.1 Previous Results on Local Filters

Despite the fact that local filters have been thoroughly studied, lower bounds for general (not necessarily *distance-respecting*) *adaptive* filters remained a big challenge.

Saks and Seshadhri [16] present a distance-respecting local filter for monotonicity of functions $f : [n]^d \to \mathbb{R}$ with running time $(\log n + 1)^{O(d)}$ per query. For monotonicity of functions $f : \{0,1\}^d \to \mathbb{R}$, no nontrivial (i.e., performing $o(2^d)$ lookups per query) filter is known. Saks and Seshadhri also show that a *distance-respecting* local filter for monotonicity on the domain $\{0,1\}^d$ must perform $2^{\Omega(d)}$ lookups per query. This lower bound crucially uses the fact that the filter is distance respecting, and does not apply to general local filters (even when no additive error is allowed).

As we explained, in many applications the extra requirement that the filter be distance-respecting is not necessary. Bhattacharyya *et al.* [5] studied lower bounds for local monotonicity filters which are not necessarily distance-respecting. However, their

[2] More generally, if a user wants to evaluate a function f with Lipschitz constant at most ℓ, where $\ell > 1$, then the Lipschitz function f/ℓ can be submitted to the server. When the noisy answer returned by the server is multiplied by ℓ, the effect is to add noise proportional to ℓ.

super-polynomial lower bounds only hold for *nonadaptive* filter. For the domain $\{0,1\}^d$, Bhattacharyya *et al.* show that nonadaptive filters must perform $\Omega(\frac{2^{\alpha d}}{d})$ lookups per query in the worst case, where $\alpha \geq 0.1620$. For adaptive filters, their bound quickly degrades with the number of lookups performed to *incomparable* points in the domain ($x, y \in [n]^d$ are *comparable* if $x \preceq y$ or $y \preceq x$ and *incomparable* otherwise). Specifically, their lower bounds for adaptive filters is $\Omega(\frac{2^{\alpha d-\ell}}{d})$, where ℓ is the number of lookups to points incomparable to x made on query x; for arbitrary adaptive filters, this degrades to $\Omega(d)$. Prior to our work, no super-polynomial lower bound for adaptive local monotonicity filter was known.

For the Lipschitz property, Jha and Raskhodnikova [11] obtain a deterministic nonadaptive local filter that runs in time $O((\log n + 1)^d)$ per query. They also show that the lower bound from [5] for *nonadaptive* filters, with the same statement, applies to *nonadaptive* local filters of the Lipschitz property.

Previous work left open whether it is possible to obtain (adaptive and not necessarily distance-respecting) local filters monotonicity and Lipschitz properties that make only $\mathrm{poly}(n, d)$ lookups per query.

1.2 Our Results and Techniques

We consider local a-filters, which is the relaxation of local filters that allows additive error a, as described above and formally stated in Definition 2.1. These filters do not need to be distance-respecting and can be fully adaptive. Our main results, stated in more detail in Section 2, are that even such relaxed filters need to perform a number of lookups exponential in the dimension d in order to reconstruct a Lipschitz (resp., monotone) function. (This applies even to functions on the domain $\{0,1\}^d$).

Theorem 1.1 (Limitations of Lipschitz filters). *Consider the Lipschitz property of functions $f : \{0,1\}^d \to \mathbb{R}$ and any (randomized) local (not necessarily distance-respecting) $\frac{d}{402}$-filter for this property. Then there is a function f and a query x where, with constant probability, this filter makes $2^{\Omega(d)}$ lookups.*

The additive error $a = d/402$ in the theorem above is as large as possible up to a constant factor: the trivial filter that outputs $F(x) = (f(\mathbf{0}) + f(\mathbf{1}))/2$, where $\mathbf{0}$ and $\mathbf{1}$ are all-0 and all-1 vectors, respectively, is a local $\frac{d}{2}$-filter.[3] To see this, note that (i) the reconstructed function $F(x)$ is Lipschitz and (ii) if the input function $f(x)$ is Lipschitz then $|F(x) - f(x)| = \frac{1}{2}|f(\mathbf{0}) + f(\mathbf{1}) - 2f(x)| \leq \frac{1}{2}(|f(\mathbf{0}) - f(x)| + |f(\mathbf{1}) - f(x)|) \leq \frac{1}{2}(\|\mathbf{0} - x\|_1 + \|\mathbf{1} - x\|_1) = \frac{d}{2}$ for every $x \in \{0,1\}^d$.

For monotonicity, we can prove an analogous theorem with no upper bound on a. This is explained by the fact that monotonicity is determined by the order of the values at different points and not their magnitudes. To calibrate the additive error, we state the next theorem for functions with bounded range, namely, $[0, 2a + 1]$. The additive error in the theorem is also tight because for functions with that range, the trivial filter above that outputs $F(x) = (f(\mathbf{0}) + f(\mathbf{1}))/2$ is a local $(a + \frac{1}{2})$-filter.

[3] In order to simplify the presentation, we did not attempt to optimize this constant factor. In particular, the choice of weights $d/3$ and $2d/3$ in Definition 3.1 might not give the best factor.

Theorem 1.2 (Limitations of monotonicity filters). *Consider the monotonicity property of functions $f : \{0,1\}^d \to [0, 2a+1]$ and any (randomized) local a-filter for this property. Then there is a function f and query x where, with constant probability, this filter makes $2^{\Omega(d)}$ lookups.*

To introduce the ideas used in the proofs, we focus for now on deterministic filters. To obtain lower bounds for *nonadaptive* filters in [5, 11], the authors construct two collections of 'hard functions' $f^{(x,y)}$ and $f^{(\overline{x,y})}$ (satisfying the Lipschitz property) indexed by $x, y \in \{0,1\}^d$. They show that if a local filter works correctly on $f^{(x,y)}$ and $f^{(\overline{x,y})}$, as well as on a suitably defined function $h^{(x,y)}$ (violating the Lipschitz property on (x, y)), the lookups made on queries x and y need to have a structured interaction. (Note that in this case the lookups are independent of the input function because the filter is nonadaptive.) More precisely, they construct a graph over $\{0,1\}^d$ based on these interactions and show that it is a 2-transitive-closure-spanner (2-TC-spanner) for the hypercube. (TC-spanners were introduced in [6]; see Section 3 for definition and comparison with c-connectors that we introduce.) Using the lower bound on the size of a 2-TC-spanner for the hypercube from [5], it can be shown that any non-adaptive filter must use exponential lookups on one of the query points.

In the case of adaptive filters one cannot assume that the lookups made on a given query point are independent of the input function. One simple idea to try to overcome this obstacle is to consider, for each query x, the union of the lookups made on query x over all possible choice of hard functions. One can then try to apply the lower bound approach discussed in the previous paragraph. In fact this union of lookups still has strong interactions that imply a 2-TC-spanner. The problem is that this is clearly overcounting the number of lookups made by the filter on a single given function on query x. Due to the large number of 'hard functions' considered in [5, 11], this overcounting makes the bound coming from the 2-TC-spanners vacuous for adaptive filters; this is where the factor 2^ℓ lost in [5] mentioned above comes from.

In order to remedy this, we build a collection of hard functions which are much 'smoother' than those from [5, 11]. This allows us to use fewer functions. However, it comes at a cost: the interactions of the lookups caused by these functions are not as structured as before and do not imply a 2-TC-spanner. We introduce a type of directed graph called *c-connector* (Definition 3.2) which captures lookup interactions. When arc directions are ignored, a c-connector is a relaxation of 2-TC-spanners (as discussed in Section 3, our transformation to c-connectors preserves information on whether x is looked up on query y or vice versa, while this information is lost in the transformation to 2-TC-spanners in [5, 11]). Nevertheless, we can argue that a c-connector has a large maximum outdegree, which relates to the lookup complexity. Indeed, one of the key ingredients for our lower bound is recognizing the limitations of 2-TC-spanners in this context and finding a combinatorial structure with the right amount of flexibility. Given the importance of TC-spanners (see [13] for a survey), c-connectors might find use outside of this work.

Organization. Section 2 gives basic definitions and a more detailed statement of our main results. In Section 3, we define c-connectors, the graph objects on which our lower bounds are based. In Sections 4 and 5, we develop a connection between c-connectors and local filters for the Lipschitz property and monotonicity. In Section 6, we bound

the outdegree of c-connectors. Our lower bounds follow directly from putting these two parts together.

2 Definitions and Formal Statement of Results

Given a point $x \in \{0,1\}^d$, we use x_i to denote its ith coordinate and $|x|$ to denote its Hamming weight, that is, $|x| = \sum_i x_i$. We identify each point $x \in \{0,1\}^d$ with the subset of coordinates where it takes value 1, namely, $\{i : x_i = 1\}$. This gives meaning to expressions like $x \subseteq y$, $x \cap y$, $x \cup y$ and $x \setminus y$ for $x, y \in \{0,1\}^d$. For $x \in \{0,1\}^d$, the Hamming weight $|x|$ coincides with the cardinality of the set associated with x.

We now provide a formal definition of local a-filters that allow additive error a. It is stated for a general property P of functions with domain D; in our case, P will be either the Lipschitz property or monotonicity.

Definition 2.1 (Local a-filter). *Let* P *be a property of functions $f : D \to R$ for some $R \subseteq \mathbb{R}$. A local a-filter for* P *with error probability δ is a randomized algorithm which is given black-box access to a function $f : D \to R$ together with a* query *point $x \in D$. For each random seed σ in the algorithm's probability space $(\Omega, \Pr)$, the filter obtains the value of f on a sequence of points $L(\sigma, f, x) = \{y_1, y_2, \ldots, y_k\}$, called* lookups*, (where the choice of y_i depends only on x, σ and $f(y_1), f(y_2), \ldots, f(y_{i-1})$) and outputs a reconstructed value $F(\sigma, f, x)$ for x solely based on the values of f at $L(\sigma, f, x)$. The reconstructed function $F_{\sigma,f} : D \to R$ given by $F_{\sigma,f}(x) = F(\sigma, f, x)$ must obey two conditions: (i) $F_{\sigma,f}$ satisfies property* P *for all functions f and all random seeds σ; (ii) if f satisfies property* P *then for all $x \in D$ we have $\Pr_\sigma(F_{\sigma,f}(x) \in [f(x) - a, f(x) + a]) \geq 1 - \delta$.*

Notice that requirement (ii) in this definition is weaker than requiring that "if f satisfies property P then $\Pr_\sigma(\forall x \in D, F_{\sigma,f}(x) \in [f(x) - a, f(x) + a]) \geq 1 - \delta$"; therefore, we manage to obtain lower bounds for a more general class of filters. As a notational remark, we usually omit the probability space and denote a local a-filter by (L, F).

The next observation captures the structural rigidity of local filters exploited in our lower bounds. It states that if functions f and g are identical on the lookups performed on query x when the input function is f, then the filter will perform the same lookups on x for both f and g and, consequently, reconstruct the same value.

Observation 2.1. *Let (L, F) be a local a-filter. Then the following holds for every random seed σ and query point x: if f and g are functions such that $f|_{L(\sigma,f,x)} = g|_{L(\sigma,f,x)}$, then $F(\sigma, f, x) = F(\sigma, g, x)$.*

Now we restate Theorems 1.1 and 1.2, giving more details about parameters we obtain.

Theorem 2.1. *Fix a non-negative constant δ, consider a sufficiently large integer d (depending on δ) and let $a \in [0, d/402]$. Let (L, F) be a local a-filter for the Lipschitz property with error probability δ. Then there exists a function $f : \{0,1\}^d \to \mathbb{R}$ and a query $x \in \{0,1\}^d$ such that $\Pr_\sigma(|L(\sigma, f, x)| \geq 2^{0.009d}) \geq 1/2 - 1.1\delta$.*

Theorem 2.2. *Fix a non-negative constant δ, consider a sufficiently large integer d (depending on δ) and let $a \geq 0$. Let (L, F) be a local a-filter for monotonicity with error probability δ. Then there exists a function $f : \{0,1\}^d \to [0, 2a+1]$ and a query $x \in \{0,1\}^d$ such that $\Pr_\sigma(|L(\sigma, f, x)| \geq 2^{0.009d}) \geq 1/2 - 1.1\delta$.*

The proof of Theorem 2.1 (resp. Theorem 2.2) follows directly from Lemma 4.3 (resp. Lemma 5.3) and Theorem 6.1; details are given in the full version [3].

3 c-Connectors

In this section, we formally introduce the notion of c-connectors. This combinatorial structure can be represented as a directed graph on the vertex set $\{0,1\}^d$, where pairs of nodes need to share an out-neighbor with some prescribed properties. As we shall see next, c-connectors are related to 2-TC-spanners, although the full motivation for the exact definition will only become clear in Sections 4 and 5.

Definition 3.1. *Let X denote the set of points in $\{0,1\}^d$ with Hamming weight exactly $d/3$ and let Y denote the set of points in $\{0,1\}^d$ with Hamming weight exactly $2d/3$. Also let $\mathcal{P}$ denote the set of comparable pairs $(x, y) \in X \times Y$, namely, such that $x \prec y$.*

Definition 3.2 (c-connector). *Fix $c \in \mathbb{N}$. Given a subset $\mathcal{P}'$ of $\mathcal{P}$, a digraph G with the node set $\{0,1\}^d$ is a* c-connector *for $\mathcal{P}'$ if for every $(x, y) \in \mathcal{P}'$ there exists $z \in \{0,1\}^d$ with the following properties:*

- *(Connectivity) The arcs (x, z) and (y, z) belong to G.*
- *(Structure) $|z \setminus y| < c$ and $|z| > \frac{d}{3} - c$.*

A *2-TC-spanner* of the boolean hypercube (with the usual partial order) is a directed graph H on the node set $\{0,1\}^d$ with the property that for all $x \prec y$ there is a point z satisfying $x \preceq z \preceq y$, such that the arcs (x, z) and (z, y) belong to H [6]. If we reorient the arcs in a 2-TC-spanner of the hypercube, so that the nodes in Y only have outgoing arcs, we obtain a valid c-connector for every $c \geq 1$: this is because the requirement $x \preceq z \preceq y$ (in the definition of 2-TC-spanner) implies the structure requirement in a c-connector. Therefore, c-connectors relax 2-TC-spanners in two ways: first it requires that only pairs in $\mathcal{P}$ have a common neighbor with prescribed properties, and second it relaxes the required properties of this common neighbor. We remark that the direction of the arcs in c-connectors is important here, since in order to obtain the desired results we lower bound the outdegree. In contrast, in previous work [5, 11] the information of whether point x was looked up on query y or vice versa was lost in the transformation to the corresponding 2-TC-spanner and the lower bound on the number of arcs, not the outdegree, was used. This is one of the changes that gives us stronger lower bounds.

4 Local Filters for the Lipschitz Property Imply c-Connectors

In this section we focus on the Lipschitz property. We construct a family of functions such that a local a-filter that works correctly on functions from the family must preform lookups corresponding to a c-connector. The idea is to start with a Lipschitz function

f^0 and then construct other Lipschitz functions f^c_y which agree with f^0 on most points, but where $f^c_y(y)$ is much larger than $f^0(y)$. We argue that if a purported local a-filter makes only 'local' lookups when reconstructing at queries x and y, then we can create a function that looks like f^c_y around y (so that the filter is fooled and returns $F(y)$ in the range $f^c_y(y) \pm a$) and looks like f^0 around x (so that the filter is fooled and returns $F(x)$ in the range $f^0(x) \pm a \ll f^c_y(y) \pm a$). Thus, for the returned function, $F(x)$ and $F(y)$ are too far apart, ensuring that it is not Lipschitz.

4.1 Hard Functions for Filter

Recall from Definition 3.1 that Y denotes the set of points in $\{0,1\}^d$ with Hamming weight exactly $d/3$. In order to construct these hard functions, for a point $y \in Y$ let $T_y = \{x \in \{0,1\}^d : x \subseteq y, |x| \geq d/3\}$. Define the function f^0 by $f^0(z) = \max\{|z|, d/3\}$ for all $z \in \{0,1\}^d$. Intuitively, for $c \in \mathbb{N}$ and $y \in Y$, we define the function f^c_y as the smallest Lipschitz function which is at least $f^0 + c\chi_{T_y}$, where χ_{T_y} denotes the characteristic function of the set T_y. More specifically, we set $f^c_y(z) = \max\{|z| + c - |z \setminus y|, f^0(z)\}$ for all $z \in \{0,1\}^d$.

Clearly f^0 is Lipschitz, and the functions f^c_y can be shown to be Lipschitz as well.

Lemma 4.1. *For all $c \in \mathbb{N}$ and $y \in Y$ the function f^c_y is Lipschitz.*

For a point $y \in Y$ and a constant $c \in \mathbb{N}$, let $T^c_y \subseteq \{0,1\}^d$ be the set of points z, such that $f^c_y(z) \neq f^0(z)$. Then $T^1_y = T_y$ and the set T^c_y gets larger as c increases: specifically, $T^c_y \subseteq T^{c'}_y$ for $c < c'$. The definitions of f^c_y and f^0 directly give the following observation, which justifies the specific structure used in the definition of a c-connector.

Observation 4.1. *All elements z in the set T^c_y satisfy $|z \setminus y| < c$ and $|z| > \frac{d}{3} - c$.*

4.2 Correct Reconstruction of Hard Functions Implies c-Connector

Now we show that if a local a-filter is correct on the constructed functions, its lookups correspond to a c-connector for the interesting pairs $\mathcal{P}$ (recall that $\mathcal{P}$ is the set of pairs $(x,y) \in X \times Y$ such that $x \prec y$). We start by essentially focusing on deterministic filters or, alternatively, by looking at a 'good' seed of a randomized filter. The analysis for randomized filters is based on the ability to pick a few of these good seeds and then analyzing the 'union' of the behavior of the filter running with these seeds.

Consider a local a-filter (L,F). Given points $x \in X$ and $y \in Y$, we say that a random seed $\sigma \in \Omega$ is *good* for x and y if $F_{\sigma,f^0}(x) \in [f^0(x) - a, f^0(x) + a]$ and $F_{\sigma,f^c_y}(y) \in [f^c_y(y) - a, f^c_y(y) + a]$. Given a seed σ which is good for x and y, we define the digraph $G^{xy}_\sigma = (\{0,1\}^d, A^{xy}_\sigma)$ that captures the lookups made on queries x and y. Specifically, the set A^{xy}_σ consists of all the arcs $\{(x,z) : z \in L(\sigma, f^0, x) \cup \{x\}\}$ and $\{(y,z) : z \in L(\sigma, f^c_y, y) \cup \{y\}\}$.

Lemma 4.2 (Local filter implies c-connector). *Consider a local a-filter (L,F) for the Lipschitz property and an integer $c > 2a$. For all $(x,y) \in \mathcal{P}$, if $\sigma \in \Omega$ is good for x and y then G^{xy}_σ is a c-connector for (x,y).*

Proof. For the sake of contradiction suppose not. Unraveling the definitions and using Observation 4.1 this means that the sets $(L(\sigma, f^0, x) \cup \{x\}) \cap T_y^c$ and $(L(\sigma, f_y^c, y) \cup \{y\}) \cap T_y^c$ do not intersect. Then let A, B be a partition of T_y^c such that A contains $(L(\sigma, f^0, x) \cup \{x\}) \cap T_y^c$ and B contains $(L(\sigma, f_y^c, y) \cup \{y\}) \cap T_y^c$. Define the function f such that $f|_A = f^0|_A$, $f|_B = f_y^c|_B$, and $f|_{\{0,1\}^d \setminus (A \cup B)} = f^0|_{\{0,1\}^d \setminus (A \cup B)} = f_y^c|_{\{0,1\}^d \setminus (A \cup B)}$ (the last equation follows from the definition of T_y^c). To reach a contradiction, we show that the filter does not reconstruct f correctly.
Notice that $f^0|_{L(\sigma,f^0,x)} = f|_{L(\sigma,f^0,x)}$, so Observation 2.1 gives that $F(\sigma, f, x) = F(\sigma, f^0, x)$. Similarly, $f_y^c|_{L(\sigma,f_y^c,y)} = f|_{L(\sigma,f_y^c,y)}$ and hence $F(\sigma, f, y) = F(\sigma, f_y^c, y)$.

Now since σ is good for x and y, we have that $F(\sigma, f, x) = F(\sigma, f^0, x) \leq f^0(x) + a = \frac{d}{3} + a$ and $F(\sigma, f, y) = F(\sigma, f_y^c, y) \geq f_y^c(y) - a = \frac{2d}{3} + c - a$. Since $c > 2a$ we get $F(\sigma, f, y) - F(\sigma, f, x) > d/3 = \|x - y\|_1$, and hence the function $F_{\sigma,f}$ is not Lipschitz; this contradicts that (L, F) is a local a-filter and concludes the proof. □

Consider two subsets $\mathcal{P}_1, \mathcal{P}_2$ of $\mathcal{P}$. Notice that if G_1 is a c-connector for $\mathcal{P}_1$ and G_2 is a c-connector for $\mathcal{P}_2$ then the graph formed by the union of (the arcs of) G_1 and G_2 is a c-connector for $\mathcal{P}_1 \cup \mathcal{P}_2$. We remark that when we take this union we do not add parallel arcs. This directly gives the following result.

Corollary 4.1. *Consider a local a-filter (L, F) for the Lipschitz property and an integer $c > 2a$. Suppose that for each $(x, y) \in \mathcal{P}$ there is a random seed $\sigma(x, y) \in \Omega$ which is good for x and y. Then the graph obtained as the union of the graphs $\{G^{xy}_{\sigma(x,y)}\}_{(x,y)\in\mathcal{P}}$ is a c-connector for $\mathcal{P}$. Moreover, this graph has outdegree at most*

$$\max\left\{\max_{x\in X}\left\{\left|\bigcup_y L(\sigma(x,y), f^0, x)\right|\right\}, \max_{y\in Y}\left\{\left|\bigcup_x L(\sigma(x,y), f_y^c, y)\right|\right\}\right\} + 1. \quad (1)$$

Using this corollary, we show that a local a-filter with small 'average' number of lookups implies a c-connector for $\mathcal{P}$ with a small outdegree. The idea is to construct, via the probabilistic method, a set $\bar{S} \subseteq \Omega$ of good seeds which attains a small value in (1); details are provided in the full version [3].

Lemma 4.3. *Consider a local a-filter (L, F) for the Lipschitz property with error probability δ and an integer $c > 2a$. Consider $\alpha > 0$ and let $M = \max_{f,x} \Pr_\sigma(|L(\sigma, f, x)| > \alpha)$. If $\delta + M < 1/2$ then there is a c-connector for $\mathcal{P}$ with maximum outdegree at most $2d\alpha / \log\left(\frac{1}{2\delta+2M}\right) + 1$.*

5 Local Filters for Monotonicity Imply 1-Connectors

In this section, we consider the monotonicity property and show that again the lookups performed by a local a-filter give rise to a c-connector (in this case, with $c = 1$).

5.1 Hard Functions for Filter

Again, we start by defining functions f^0 and f_y^a, such that if a local filter is correct on these functions, its lookups correspond to a 1-connector. Recall that for a point $y \in Y$,

we define $T_y = \{x \in \{0,1\}^d : x \subseteq y, |x| \geq d/3\}$. Define the function f^0 by $f^0(z) = 2a+1$ if $|z| \geq d/3$ and $f^0(z) = 0$ if $|z| < d/3$. For a point $y \in Y$, we define the function f^a_y equal to $f^0 - (2a+1)\chi_{T_y}$, namely, $f^a_y(z) = 2a+1$ if $z \geq d/3$ and $z \notin T_y$ and $f^a_y(z) = 0$ otherwise. It can be easily verified that these functions are monotone.

Lemma 5.1. *For all $y \in Y$ and $a \geq 0$, the functions f^0 and f^a_y are monotone.*

Notice that the functions f^0 and f^a_y differ exactly on points in T_y and that T_y is the set of points which satisfy the structure property in the definition of a 1-connector.

5.2 Correct Reconstruction of Hard Functions Implies 1-Connector

Recall that $\mathcal{P}$ is the set of comparable pairs $(x,y) \in X \times Y$ or, equivalently, pairs where $x \in T_y$. Consider a local a-filter (L,F) for monotone functions. As before, given $x \in X$ and $y \in Y$, we say that a random seed $\sigma \in \Omega$ is *good* for x and y if $F_{\sigma,f^0}(x) \in [f^0(x)-a, f^0(x)+a]$ and $F_{\sigma,f^a_y}(y) \in [f^a_y(y)-a, f^a_y(y)+a]$. Given a seed σ which is good for x and y, we define the digraph $G^{xy}_\sigma = (\{0,1\}^d, A^{xy}_\sigma)$ in a way similar to what we did in the previous section: we add to A^{xy}_σ all the arcs $\{(x,z) : z \in L(\sigma, f^0, x) \cup \{x\}\}$ and $\{(y,z) : z \in L(\sigma, f^a_y, y) \cup \{y\}\}$.

Again the construction of our functions and the digraph G^{xy}_σ together with the behavior of local a-filters captured in Observation 2.1 give the following.

Lemma 5.2. *Take $a \geq 0$ and consider a local a-filter (L,F) for monotonicity. For any $(x,y) \in P$, if $\sigma \in \Omega$ is good for x and y then G^{xy}_σ is a 1-connector for (x,y).*

Finally, we can utilize the same technique for finding a set of good seeds which achieve small value in (1) as done in Lemma 4.3 to obtain the desired connection between local a-filters and 1-connectors for $\mathcal{P}$.

Lemma 5.3. *Take $a \geq 0$ and consider a local a-filter (L,F) for monotone functions with error probability δ. Consider $\alpha > 0$ and let $M = \max_{f,x} \Pr_\sigma(|L(\sigma,f,x)| > \alpha)$. If $\delta + M < 1/2$ then there is a 1-connector for $\mathcal{P}$ with maximum outdegree at most $2d\alpha / \log\left(\frac{1}{2\delta+2M}\right) + 1$.*

6 Lower Bound on the Maximum Outdegree of a c-Connector

Recall that $\mathcal{P}$ is the set of pairs $(x,y) \in X \times Y$ such that x and y are *comparable*. We show a lower bound on the maximum outdegree of a c-connector for $\mathcal{P}$. We remark that the constants in the bound are not optimized.

Theorem 6.1. *Consider $d \geq 40,200$ and let c be an integer in the range $[d/201, d/200]$. Then the maximum outdegree of any c-connector for $\mathcal{P}$ is at least $2^{0.01d}$.*

To prove this, let G be a c-connector for $\mathcal{P}$. Let $\tilde{T}^c_y = \{z : |z \setminus y| < c, |z| > d/3 - c\}$ be the points which satisfy the structure property in Definition 3.2. Then $T_y \subseteq T^c_y \subseteq \tilde{T}^c_y$ for all $y \in Y$, and for $x \in T_y$ and $z \in \tilde{T}^c_y$ we have $x \cup z \in \tilde{T}^c_y$. We say that a pair $(x,y) \in \mathcal{P}$ is *covered* by a point z if $z \in \tilde{T}^c_y$ and the arcs (x,z) and (y,z) belong to G.

Each pair in $\mathcal{P}$ needs to be covered by a point. For a fixed $x \in X$, the outdegree of x in G is at least the number of distinct points which are covering the pairs in $\mathcal{P}$ containing x (and similarly for a fixed $y \in Y$). The difficulty in lower bounding the outdegree of x is that many pairs containing it can be covered by the same point. The heart of the argument is to show that no point can cover too many such pairs. It relies on the fact that the sets $\tilde{T}_y^c$ are 'localized'. More precisely, consider a point z and let (x, y) be covered by it. Notice that $x \in T_y$ and $z \in \tilde{T}_y^c$, hence $x \cup z \in \tilde{T}_y^c$. If z is not near x, namely, $z \setminus x$ is large, then we argue that not too many points y satisfy $x \cup z \in \tilde{T}_y^c$, given the localization of $\tilde{T}_y^c$. On the other hand, if z is near x then there are not too many possibilities for x itself. Our bound is derived by putting these observations together.

In order to make the above argument work we divide the pairs in $\mathcal{P}$ into two groups based on the covers they have. Let $\alpha \in [1/15, 1/14]$ be such that αd is an integer, which exists since d is sufficiently large. For $(x, y) \in \mathcal{P}$ and z that covers (x, y), if $|z \setminus x| \leq \alpha d$, then we say that z is *near* x and that z is a *nearby cover* of (x, y). Let $\mathcal{N}$ denote the set of pairs $(x, y) \in \mathcal{P}$ which have a nearby cover and let $\mathcal{F} = \mathcal{P} \setminus \mathcal{N}$ be the remaining pairs. For a fixed $y \in Y$, define $\mathcal{N}_y$ as the pairs in $\mathcal{N}$ containing y and for $x \in X$ define $\mathcal{F}_x$ as the pairs in $\mathcal{F}$ containing x.

Let $Z \subseteq \{0, 1\}^d$ be the set of points which cover at least one pair in $\mathcal{P}$. For a given $x \in X$, we use Z_x to denote the set of points which cover at least one pair in $\mathcal{P}$ containing x. We define Z_y analogously. That is, Z is the union of sets Z_x and Z_y over all $x \in X$ and $y \in Y$.

Now we sketch the argument that upper bounds the number of pairs in $\mathcal{N}$ and $\mathcal{F}$; computations are presented in the full version [3]. In order to upper bound $\mathcal{N}$ we start by arguing that, for a fixed $y \in Y$, a point cannot be a nearby cover for many pairs (x, y) in $\mathcal{N}_y$. To see this, take $z \in Z_y$ and let $(x, y) \in \mathcal{P}$ be such that z is a nearby cover for it. Then notice that x and z are very similar: $|z \setminus x| \leq \alpha d$ and $|x \setminus z| \leq \alpha d + c$; the first bound follows from the definition of a nearby cover and the second uses $|z| \geq |x| - c$ from Observation 4.1. From these constraints, it follows that there are at most $d^2 \binom{d/3+\alpha d}{\alpha d} \binom{2d/3+c}{\alpha d+c}$ possibilities for such x's. Thus, for all $y \in Y$ we have $|\mathcal{N}_y| \leq |Z_y| \cdot d^2 \binom{d/3+\alpha d}{\alpha d} \binom{2d/3+c}{\alpha d+c}$. Adding over all y gives the desired bound.

Lemma 6.1. *Letting* $\Theta = d^2 \binom{d/3+\alpha d}{\alpha d} \binom{2d/3+c}{\alpha d+c}$, *the number of pairs in* $\mathcal{N}$ *is at most* $|Y| \cdot \Theta \cdot \max_{y \in Y}\{|Z_y|\}$.

To upper bound the size of $\mathcal{F}$ we start by showing that, for a fixed $x \in X$, a point cannot be a (non-nearby) cover for too many pairs in $\mathcal{F}_x$. To see this, take $z \in Z_x$ and suppose $(x, y) \in \mathcal{F}_x$ is covered by z. Notice that $x \cup z$ and y are very similar: $|(x \cup z) \setminus y| \leq c$ and $|y \setminus (x \cup z)| \leq d/3 - \alpha d + c$; the first bound follows from $x \subseteq y$ and Observation 4.1, and the second further uses the fact that $|x \cup z| \geq d/3 + \alpha d$ (since z is not a nearby cover). Then it is easy to see that there are at most $d^2 \binom{\frac{2d}{3}+c}{c} \binom{\frac{2d}{3}-\alpha d}{\frac{d}{3}-\alpha d+c}$ such y's. Thus, for each $x \in X$ we have $|\mathcal{F}_x| \leq |Z_x| \cdot d^2 \binom{\frac{2d}{3}+c}{c} \binom{\frac{2d}{3}-\alpha d}{\frac{d}{3}-\alpha d+c}$ and adding over all x gives the desired bound on $\mathcal{F}$.

Lemma 6.2. *Letting* $\Phi = d^2 \binom{\frac{2d}{3}+c}{c} \binom{\frac{2d}{3}-\alpha d}{\frac{d}{3}-\alpha d+c}$, *the number of pairs in* $\mathcal{F}$ *is at most* $|X| \cdot \Phi \cdot \max_{x \in X}\{|Z_x|\}$.

The maximum outdegree of the c-connector G is bounded from below by

$$M \triangleq \max\{\max_{x \in X}\{|Z_x|\}, \max_{y \in Y}\{|Z_y|\}\}.$$

Since $\mathcal{N}$ and $\mathcal{F}$ partition the set of pairs $\mathcal{P}$, we can add the bounds from Lemmas 6.1 and 6.2 and obtain that M is at least the size of $\mathcal{P}$ divided by $\binom{d}{d/3}(\Theta + \Phi)$, which gives $M \geq \binom{2d/3}{d/3}/(\Theta + \Phi)$. Standard computations can be used to lower bound the right-hand side of this expression by $2^{0.01d}$. This concludes the proof of Theorem 6.1.

7 Conclusion and Future Work

We show that local filters for the Lipschitz property and monotonicity require exponentially many (in the dimension) lookups, even when allowed additive error. One can try to further relax the requirements on local filters in order to overcome these lower bounds.

One possibility is to consider local filters whose output does not satisfy the desired property P with small probability. Such weaker guarantees can still be useful for other definitions of privacy [4, 8]. Another relaxation, specific to the Lipschitz property, is to allow the reconstructed function F to be b-Lipschitz, that is, to require only $|F(x) - F(y)| \leq b \cdot \|x - y\|_1$ for all $x, y \in \{0,1\}^d$. For the privacy application described, having a and b of order $O(\sqrt{d})$ is still acceptable. We remark that the techniques presented here yield similar lower bounds for b slightly larger than 1, but not for $b \geq 2$.

References

[1] Ailon, N., Chazelle, B., Comandur, S., Liu, D.: Property-preserving data reconstruction. Algorithmica 51(2), 160–182 (2008)

[2] Alon, N., Rubinfeld, R., Vardi, S., Xie, N.: Space-efficient local computation algorithms. In: Rabani, Y. (ed.) SODA, pp. 1132–1139. SIAM (2012)

[3] Awasthi, P., Jha, M., Molinaro, M., Raskhodnikova, S.: Limitations of local filters of lipschitz and monotone functions. Electronic Colloquium on Computational Complexity (ECCC) TR12-075 (2012)

[4] Bhaskar, R., Bhowmick, A., Goyal, V., Laxman, S., Thakurta, A.: Noiseless Database Privacy. In: Lee, D.H., Wans, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 215–232. Springer, Heidelberg (2011)

[5] Bhattacharyya, A., Grigorescu, E., Jha, M., Jung, K., Raskhodnikova, S., Woodruff, D.P.: Lower bounds for local monotonicity reconstruction from transitive-closure spanners. SIAM J. Discrete Math. 26(2), 618–646 (2012)

[6] Bhattacharyya, A., Grigorescu, E., Jung, K., Raskhodnikova, S., Woodruff, D.P.: Transitive-closure spanners. In: SODA, pp. 932–941 (2009)

[7] Blum, M., Luby, M., Rubinfeld, R.: Self-testing/correcting with applications to numerical problems. J. Comput. Syst. Sci. 47(3), 549–595 (1993)

[8] Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our Data, Ourselves: Privacy Via Distributed Noise Generation. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 486–503. Springer, Heidelberg (2006)

[9] Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating Noise to Sensitivity in Private Data Analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
[10] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. J. ACM 45(4), 653–750 (1998)
[11] Jha, M., Raskhodnikova, S.: Testing and reconstruction of Lipschitz functions with applications to data privacy. In: IEEE FOCS, pp. 433–442 (2011) full version available at, http://eccc.hpi-web.de/report/2011/057/
[12] Katz, J., Trevisan, L.: On the efficiency of local decoding procedures for error-correcting codes. In: STOC, pp. 80–86 (2000)
[13] Raskhodnikova, S.: Transitive-Closure Spanners: A Survey. In: Goldreich, O. (ed.) Property Testing. LNCS, vol. 6390, pp. 167–196. Springer, Heidelberg (2010)
[14] Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. SIAM J. Comput. 25(2), 252–271 (1996)
[15] Rubinfeld, R., Tamir, G., Vardi, S., Xie, N.: Fast local computation algorithms. In: ICS, pp. 223–238 (2011)
[16] Saks, M.E., Seshadhri, C.: Local monotonicity reconstruction. SIAM J. Comput. 39(7), 2897–2926 (2010)

Testing Lipschitz Functions on Hypergrid Domains*

Pranjal Awasthi[1], Madhav Jha[2], Marco Molinaro[1], and Sofya Raskhodnikova[2,**]

[1] Carnegie Mellon University, USA
{pawasthi,molinaro}@cmu.edu
[2] Pennsylvania State University, USA
{mxj201,sofya}@cse.psu.edu

Abstract. A function $f(x_1, ..., x_d)$, where each input is an integer from 1 to n and output is a real number, is Lipschitz if changing one of the inputs by 1 changes the output by at most 1. In other words, Lipschitz functions are not very sensitive to small changes in the input. Our main result is an efficient tester for the Lipschitz property of functions $f : [n]^d \to \delta\mathbb{Z}$, where $\delta \in (0, 1]$ and $\delta\mathbb{Z}$ is the set of integer multiples of δ.

The main tool in the analysis of our tester is a smoothing procedure that makes a function Lipschitz by modifying it at a few points. Its analysis is already non-trivial for the 1-dimensional version, which we call Bubble Smooth, in analogy to Bubble Sort. In one step, Bubble Smooth modifies two values that violate the Lipschitz property, i.e., differ by more than 1, by transferring δ units from the larger to the smaller. We define a *transfer graph* to keep track of the transfers, and use it to show that the ℓ_1 distance between f and BubbleSmooth(f) is at most twice the ℓ_1 distance from f to the nearest Lipschitz function. Bubble Smooth has other important properties, which allow us to obtain a *dimension reduction*, i.e., a reduction from testing functions on multidimensional domains to testing functions on the 1-dimensional domain, that incurs only a small multiplicative overhead in the running time and thus avoids the exponential dependence on the dimension.

1 Introduction

Property testing aims to understand how much information is needed to decide (approximately) whether an object has a property. A *property tester* [8, 5] is given oracle access to an object and a proximity parameter $\epsilon \in (0, 1)$. If an object has the desired property, the tester *accepts* it with probability at least $2/3$; if the object is ϵ-far from having the desired property then the tester *rejects* it with probability at least $2/3$. Specifically, for properties of functions, ϵ-far means that a given function differs on at least an ϵ fraction of the domain points from any function with the property. Properties of different types of objects have been studied, including graphs, metrics spaces, images and functions.

We present efficient testers for the Lipschitz property of functions[1] $f : [n]^d \to \delta\mathbb{Z}$, where $\delta \in (0, 1]$ and $\delta\mathbb{Z}$ is the set of integer multiples of δ. A function f is *c-Lipschitz*

* All omitted proofs appear in the full version [1].

** P.A. is supported by NSF grant CCF-1116892. M.J. and S.R. are supported by NSF CAREER grant CCF-0845701 and NSF grant CCF-0729171. M.M. is supported by NSF grant CMMI-1024554.

[1] The set $\{1, \ldots, n\}$ is denoted by $[n]$.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 387–398, 2012.
© Springer-Verlag Berlin Heidelberg 2012

(with respect to the ℓ_1 metric on the domain) if $|f(x) - f(y)| \leq c \cdot |x - y|_1$. Points in the domain $[n]^d$ can be thought of as vertices of a d-dimensional *hypergrid*, where every pair of points at ℓ_1 distance 1 is connected by an edge. Each edge (x, y) imposes a constraint $|f(x) - f(y)| \leq c$ and a function f is c-Lipschitz iff every edge constraint is satisfied. We say a function is *Lipschitz* if it is 1-Lipschitz. (Note that rescaling by a factor of $1/c$ converts a c-Lipschitz function into a Lipschitz function.)

Testing of the Lipschitz property was first studied by Jha and Raskhodnikova [7] who motivated it by applications to data privacy and program verification. They presented testers for the Lipschitz property of functions on the domains $\{0, 1\}^d$ (the *hypercube*) and $[n]$ (the *line*) that run in time $O(d^2/(\delta\epsilon))$ and $O(\log n/\epsilon)$, respectively. Even though the applications in [7] are most convincing for functions on general hypergrid domains (in one of their applications, for instance, a point in $[n]^d$ represents a histogram of a private database), no nontrivial tester for functions on such general domains was known prior to this work.

1.1 Our Results

We present two efficient testers of the Lipschitz property of functions of the form $f : [n]^d \to \delta\mathbb{Z}$ with running time polynomial in d, n and $(\delta\epsilon)^{-1}$. Our testers are faster for functions whose image has small diameter.

Definition 1.1 (Image diameter). *Given a function $f : [n]^d \to \mathbb{R}$, its* image diameter *is* $\mathrm{ImgD}(f) = \max_{x\in[n]^d} f(x) - \min_{y\in[n]^d} f(y)$.

Observe that a Lipschitz function on $[n]^d$ must have image diameter at most nd. However, image diameter can be arbitrarily large for a non-Lipschitz function.

Our testers are *nonadaptive*, that is, their queries do not depend on answers to previous queries. The first tester has *1-sided error*, that is, it always accepts Lipschitz functions. The second tester is faster (when $\sqrt{d} \gg \log(1/\epsilon)$ and $\mathrm{ImgD}(f)$ is large), but has 2-sided error, i.e., it can err on both positive and negative instances.

Theorem 1.1 (Lipschitz testers). *For[2] all $\delta, \epsilon \in (0, 1]$, the Lipschitz property of functions $f : [n]^d \to \delta\mathbb{Z}$ can be tested nonadaptively with the following time complexity:*

(1) in $O\left(\frac{d}{\delta\epsilon} \cdot \min\{\mathrm{ImgD}(f), nd\} \cdot \log\min\{\mathrm{ImgD}(f), n\}\right)$ time with 1-sided error.

(2) in $O\left(\frac{d}{\delta\epsilon} \cdot \min\left\{\mathrm{ImgD}(f), n\sqrt{d\log(1/\epsilon)}\right\} \cdot \log\min\{\mathrm{ImgD}(f), n\}\right)$ time with 2-sided error.

If $\mathrm{ImgD}(f)$, δ and ϵ are constant, then both testers run in $O(d)$ time. This is tight already for the range $\{0, 1, 2\}$, even for the special case of the hypercube domain [7].

1.2 Our Techniques

For clarity of presentation, we state and prove all our theorems for $\delta = 1$, i.e., for integer-valued functions. In the full version, by discretizing (as was done in [7]), we extend our results to the range $\delta\mathbb{Z}$.

[2] If $\delta > 1$ then f is Lipschitz iff it is 0-Lipschitz (that is, constant). Testing if a function is constant takes $O(1/\epsilon)$ time.

The main challenge in designing a tester for functions on the hypergrid domains is avoiding an exponential dependence on the dimension d. We achieve this via a *dimension reduction*, i.e., a reduction from testing functions on the hypergrid $[n]^d$ to testing functions on the line $[n]$, that incurs only an $O(d \cdot \min\{\mathrm{ImgD}, nd\})$ multiplicative overhead in the running time. In order to do this, we relate the distance to the Lipschitz property of a function f on the hypergrid to the average distance to the Lipschitz property of restrictions of f to 1-dimensional (axis-parallel) lines. For $i \in [d]$, let $e^i \in [n]^d$ be 1 on the ith coordinate and 0 on the remaining coordinates. Then for every dimension $i \in [d]$ and $\alpha \in [n]^d$ with $\alpha_i = 0$, the *line g of f along dimension i with position* α is the restriction of f defined by $g(x_i) = f(\alpha + x_i \cdot e^i)$, where x_i ranges over $[n]$. We denote the set of lines of f along dimension i by L_f^i and the set of all lines, i.e., $\bigcup_{i\in[d]} L_f^i$, by L_f. We denote the relative distance of a function h to the Lipschitz property, i.e., the fraction of input points where the function needs to be changed in order to become Lipschitz, by $\epsilon^{Lip}(h)$. The technical core of our dimension reduction is the following theorem that demonstrates that if a function on the hypergrid is far from the Lipschitz property then a random line from L_f is, in expectation, also far from it.

Theorem 1.2 (Dimension reduction). *For all functions $f : [n]^d \to \mathbb{Z}$, the following holds:* $\mathbb{E}_{g \leftarrow L_f}\left[\epsilon^{Lip}(g)\right] \geq \frac{\epsilon^{Lip}(f)}{2 \cdot d \cdot \mathrm{ImgD}(f)}$.

To obtain this result, we introduce a smoothing procedure that "repairs" a function (i.e., makes it Lipschitz) one dimension at a time, while modifying it at a few points. Such procedures have been previously designed for restoring monotonicity of Boolean functions [4, 3] and for restoring the Lipschitz property of functions on the hypercube domain [7]. The key challenge is to find a smoothing procedure that satisfies the following three requirements: *(1) It makes all lines along dimension i (i.e., in L_f^i) Lipschitz. (2) It changes only a small number of function values. (3) It does not make lines in other dimensions less Lipschitz, according to some measure.*

Smoothing Procedure for 1-Dimensional Functions. Our first technical contribution is a local smoothing procedure for functions $f : [n] \to \mathbb{Z}$, which we call **BubbleSmooth**, in analogy to Bubble Sort. In one *basic step*, **BubbleSmooth** modifies two consecutive values (i.e., $f(i)$ and $f(i+1)$ for some $i \in [n-1]$) that violate the Lipschitz property, namely, differ by more than 1. It decreases the larger and increases the smaller by 1, i.e., it transfers a unit from the larger to the smaller. See Algorithm 1 for the description of the order in which basic steps are applied. **BubbleSmooth** is a natural generalization of the *averaging operator* in [7], used to repair an edge of the hypercube, that can also be viewed as several applications of the basic step to the edge.

One challenge in analyzing **BubbleSmooth** is that when it is applied to all lines in one dimension, it may increase the average distance to the Lipschitz property for the lines in the remaining dimensions. Our second key technical insight is to use the ℓ_1 distance to the Lipschitz property to measure the performance of our procedure on the line and its effect on other dimensions. The ℓ_1 distance between functions f and f' on the same domain, denoted by $|f - f'|_1$, is the sum of $|f(x) - f'(x)|$ over all values x in the domain. The ℓ_1 distance of a function f to the nearest Lipschitz function over the same domain is denoted by $\ell_1^{Lip}(f)$. Observe that the Hamming distance and the ℓ_1

distance from a function to a property can differ by at most $\mathrm{ImgD}(f)$. Later, we leverage the fact that Lipschitz functions have a relatively small image diameter to relate the ℓ_1 distance to the Hamming distance.

We prove that **BubbleSmooth** returns a Lipschitz function and that it makes at most twice as many changes in terms of ℓ_1 distance as necessary to make a function Lipschitz.

Theorem 1.3. *Consider a function $f : [n] \to \mathbb{Z}$ and let f' be the function returned by **BubbleSmooth**(f). Then (1) function f' is Lipschitz and (2) $|f - f'|_1 \leq 2 \cdot \ell_1^{Lip}(f)$.*

The proof of the second part of this theorem requires several technical insights. One of the challenges is that **BubbleSmooth** changes many function values, but then undoes most changes during subsequent steps. We define a transfer graph to keep track of the transfers that move a unit of function value during each basic step. Its vertex set is $[n]$ and an edge (x, y) represents that a unit was transferred from $f(x)$ to $f(y)$. Since two transfers (x, y) and (y, z) are equivalent to a transfer (x, z), we can merge the corresponding edges in the transfer graph, proceeding with such merges until no vertex in it has both incoming and outgoing edges. As a result, we get a transfer graph, where the number of edges, $|E|$, is twice the ℓ_1 distance from the original to the final function.

To prove that $|E| \leq \ell_1^{Lip}(f)$, we show that the transfer graph has a matching with the violation score at least $|E|$. The *violation score* of an edge (or a pair) (x, y) is the quantity by which $|f(x) - f(y)|$ exceeds the distance between x and y. (Recall that $|f(x) - f(y)| \leq |x - y|$ for all Lipschitz functions f on domain $[n]$.) The violation score of a matching is the sum of the violation scores over all edges in the matching. We observe (in Lemma 2.3) that $\ell_1^{Lip}(f)$ is at least a violation score of any matching. The crucial step in obtaining a matching with a large violation score is pinpointing a provable, but strong enough property of the transfer graph that guarantees such a matching. Specifically, we show that the violation score of each edge in the graph is at least the number of edges adjacent to its endpoints at its (suitably defined) *moment of creation* (Lemma 2.1). E.g., this statement is not true for adjacent edges in the final transfer graph. The construction of a matching with a large violation score in the transfer graph is one of the key technical contributions of this paper. It is the focus of Section 2.

Dimension Reduction with Respect to ℓ_1. Our smoothing procedure for functions on the hypergrids applies **BubbleSmooth** to repair all lines in dimensions $1, 2, ..., d$, one dimension at a time. We show that for all $i, j \in [d]$ applying **BubbleSmooth** in dimension i does not increase the expected $\ell_1^{Lip}(f)$ for a random line g in dimension j. The key feature of our smoothing procedure that makes the analysis tractable is that it can be broken down into steps, each consisting of one application of the basic step of **BubbleSmooth** to the same positions $(k, k+1)$ on all lines in a specific dimension. This allows us to show that one such step does not make other dimensions worse in terms of the ℓ_1 distance to the Lipschitz property. The cleanest statement of the resulting dimension reduction is with respect to the ℓ_1 distance.

Theorem 1.4. *For all functions $f : [n]^d \to \mathbb{Z}$, we have: $\sum_{g \in L_f} \ell_1^{Lip}(g) \geq \frac{\ell_1^{Lip}(f)}{2}$.*

Our Testers and Effective Image Diameter. The main component of our tester repeats the following procedure: *Pick a line uniformly at random and run one step of the line*

tester. (We use the line tester from [7].) Our dimension reduction (Theorem 1.2) is crucial in analyzing this component. However, the bound in Theorem 1.2 depends on the image diameter of the function f. In the case of a non-Lipschitz function, it can be arbitrarily large, but for a Lipschitz function on $[n]^d$ it is at most the diameter of the space, namely nd (notice this factor in part (1) of Theorem 1.1). In fact, for our application we can also use the *observable diameter* of the space [6]: since the hypergrid exhibits Gaussian-type concentration of measure, a Lipschitz function maps the vast majority of points to an interval of size $O(n\sqrt{d})$ (notice this factor in part (2) of Theorem 1.1). Our testers use a preliminary step to rule out functions with large image diameter (resulting in 1-sided error) or with large observable diameter (resulting in 2-sided error).

1.3 Comparison to Previous and Concurrent Work

Jha and Raskhodnikova [7] gave a 1-sided error nonadaptive testers for the Lipschitz property of functions of the form $f : \{0,1\}^d \to \delta\mathbb{Z}$ and $f : [n] \to \mathbb{R}$ that run in time $O\left(\frac{d}{\delta\epsilon} \cdot \min\{\mathrm{ImgD}(f), d\}\right)$ and $O\left(\frac{\log n}{\epsilon}\right)$, respectively. They also showed that $\Omega(d)$ queries are necessary for testing the Lipschitz property on the domain $\{0,1\}^d$, even when the range is $\{0,1,2\}$. No nontrivial tester of the Lipschitz property of functions on the domain $[n]^d$ was known prior to this work.

Our first tester from Theorem 1.1 naturally generalizes the testers of [7] to functions on the domain $[n]^d$. As in [7], our tester has at most quadratic dependence on the dimension d. Our second tester from Theorem 1.1 gives an improvement in the running time over the hypercube tester in [7] at the expense of allowing 2-sided error. In this specific case, Theorem 1.1 gives a tester with running time $\tilde{O}(d^{1.5}/(\delta\epsilon))$.

Concurrently with our work, Chakrabarty and Seshadhri [2] gave an ingenious analysis of the simple edge test for the Lipschitz property (and monotonicity) of functions $f : \{0,1\}^d \to \mathbb{R}$ that shows that it is enough to run it for $O(d/\epsilon)$ time. Their analysis does not apply to functions on the domain $[n]^d$.

Organization. In Section 2, we present and analyze **BubbleSmooth**, our procedure for smoothing 1-dimensional functions, and prove Theorem 1.3. In Section 3, we use **BubbleSmooth** to construct a smoothing procedure for multidimensional functions that leads to the dimension reduction of Theorems 1.2 and 1.4. Our Lipschitz testers for functions on hypergrids claimed in Theorem 1.1 are presented in Section 4.

2 BubbleSmooth and Its Analysis

In this section, we describe **BubbleSmooth** and prove Theorem 1.3 which asserts that **BubbleSmooth**(f) outputs a Lipschitz function that does not differ too much from f in the ℓ_1 distance. In Section 2.1, we present **BubbleSmooth** (Algorithm 1) and show that it outputs a Lipschitz function. Sections 2.2 and 2.3 are devoted to proving part (2) of Theorem 1.3. At the high level, the proof follows the ideas explained in Section 1.2 (right after Theorem 1.3). In Section 2.2, we define our transfer graph (Definition 2.3)

and prove its key property (Lemma 2.1). In Section 2.3, we show that the existence of a matching with a large violation score implies that f is far from Lipschitz in the ℓ_1 distance (Lemma 2.3) and complete the proof of part (2) of Theorem 1.3 by constructing such a matching in the transfer graph.

2.1 Description of BubbleSmooth and Proof of Part (1) of Theorem 1.3

We begin this section by recalling two basic definitions from [7].

Definition 2.1 (Violation score). *Let f be a function and x, y be points in its domain. The pair (x, y) is* violated *by f if $|f(x)-f(y)| > |x-y|_1$. The* violation score *of (x, y), denoted by $\mathrm{vs}_f(x, y)$, is $|f(x) - f(y)| - |x - y|_1$ if it is violated and 0 otherwise.*

Definition 2.2 (Basic operator). *Given $f : [n]^d \to \mathbb{Z}$ and $x, y \in [n]^d$, where $|x-y|_1 = 1$ and vertex names x and y are chosen so that $f(x) \leq f(y)$, the basic operator $\mathbb{B}_{x,y}$ works as follows: If the pair (x, y) is not violated by f then $\mathbb{B}_{x,y}[f]$ is identical to f. Otherwise, $\mathbb{B}_{x,y}[f](x) = f(x) + 1$ and $\mathbb{B}_{x,y}[f](y) = f(y) - 1$.*

In this section, we view a function $f : [n] \to \mathbb{Z}$ as an integer-valued sequence $f(1), f(2), \ldots, f(n)$. We denote the subsequence $f(i), f(i+1), \ldots, f(j)$ by $f[i..j]$. Naturally, a sequence $f[i..j]$ is *Lipschitz* if $|f(k) - f(k+1)| \leq 1$ for all $i \leq k \leq j-1$. Algorithm 1 presents a formal description of **BubbleSmooth**.

Algorithm 1. BubbleSmooth (Input: an integer sequence $f[1 \ldots n]$)

```
for i = n − 1 to 1 do
    // Start phase i.
    while |f(i) − f(i + 1)| > 1 do // (i, i + 1) is violated by f
        LinePass(i).
return f
```

Algorithm 2. LinePass (Input: integer i)

```
for j = i to n − 1 do
    f ← B_{j,j+1}[f].
    // Apply basic operator (see Definition 2.2.)
```

We start analyzing the behavior of **BubbleSmooth** by proving part (1) of Theorem 1.3, which states that **BubbleSmooth** returns a Lipschitz function.

Proof (of part (1) of Theorem 1.3). Consider an integer sequence $f[1..n]$ and let $f'[1..n]$ be the sequence returned by **BubbleSmooth**(f). We prove that f' is Lipschitz by induction on the phase of **BubbleSmooth**. Initially, $f(n)$ is vacuously Lipschitz. We fix $i \in [n]$, assume $f[i+1..n]$ is Lipschitz at the beginning of phase i and show this phase terminates and that $f[i..n]$ is Lipschitz at the end of the phase.

Consider an execution of **LinePass**(i). Assume $f[i+1..n]$ is Lipschitz in the beginning of this execution. Let j be the index, such that at the beginning of the execution, $f[i..j]$ is the longest strictly monotone sequence starting from $f(i)$. Then **LinePass**(i) modifies two elements: $f(i)$ and $f(j)$. If $f(i) > f(j)$ then $f(i)$ is decreased by 1 and

$f(j)$ is increased by 1, i.e., 1 unit is *transferred* from i to j. Similarly, if $f(i) < f(j)$ then 1 unit is transferred from j to i. It is easy to see that after this transfer is performed, $f[i+1..n]$ is still Lipschitz. Moreover, each iteration of **LinePass**(i) reduces the violation score of the pair $(i, i+1)$ by at least 1. Thus, phase i terminates with $f[i..n]$ being Lipschitz. □

2.2 Transfer Graph

In the proof of part (1) of Theorem 1.3, we established that for all $i \in [n]$, each iteration of **LinePass**(i) transfers one unit to or from i. We record the transfers in the *transfer graph* $T = ([n], E)$, defined next. A transfer from x to y is recorded as a directed edge (x, y). The edges of the transfer graph are ordered (indexed), according to when they were added to the graph. The edge (i, j) (resp., (j, i)) corresponding to the most resent transfer is combined with a previously added edge (j, k) (resp., (k, j)) if such an edge exists. This is done because transfers from x to y and from y to z are equivalent to a transfer from x to z. If a new edge (x, y) is merged with an existing edge (y, z), the combined edge retains the index of the edge (y, z).

Definition 2.3 (Transfer graph). *The* transfer graph $T = ([n], E)$*, where the edge set* $E = (e_1, \ldots, e_t)$ *is ordered and edges are not necessarily distinct. The graph is defined by the following procedure. Initially,* $E = \emptyset$ *and* $t = 0$*. Each new run of* ***LinePass*** *during the execution of* ***BubbleSmooth****, transfers a unit from* i *to* j *(or resp., from* j *to* i*) for some* i *and* j*. If* j *has no outgoing (resp., incoming) edge in* T*, then increment* t *by 1 and add the edge* $e_t = (i, j)$ *(resp.,* $e_t = (j, i)$*) to* E*. Otherwise, let* e_s *be an outgoing edge* (j, k) *(resp., an incoming edge* (k, j)*) with the largest index* s*. Replace* (j, k) *with* (i, k)*, i.e.,* $e_s \leftarrow (i, k)$*. (Replace* (k, j) *with* (k, i)*, i.e.,* $e_s \leftarrow (k, i)$*.) The final transfer graph is denoted by* T^**.*

As mentioned previously, the order of creation of edges is important to formalize the desired property of the transfer graph, so we need to consider the subgraphs that consist of the first s edges $e_1, \ldots, e_s$ of E.

Definition 2.4 (Degrees). *Consider a transfer graph* T *at some time during the execution of* ***BubbleSmooth****. For all* $s \in \{0, \ldots, t\}$ *its subgraph graph* T_s *is defined as* $([n], (e_1, \ldots, e_s))$*, where* $(e_1, \ldots, e_t)$ *is the ordered edge set of* T*. (When* $s = 0$*, the edge set of* T_s *is empty.) The degree of a vertex* $x \in [n]$ *of* T_s *is denoted by* $deg_s(x)$*; when* T_s *is a subgraph of the final transfer graph, it is denoted by* $deg^*_s(x)$*.*

Observe that at no point in time can a vertex in T simultaneously have an incoming and an outgoing edge because such edges would get merged into one edge.

Lemma 2.1 (Key property of transfer graph). *Let* f *be an input function given to* ***BubbleSmooth****. Then for each edge* $e_s = (x, y)$ *of the final transfer graph* T^**, the following holds:* $\mathrm{vs}_f(x, y) \geq deg^*_s(x) + deg^*_s(y) - 1$.

To prove this lemma, we consider each phase of **BubbleSmooth** separately and formulate a slightly stronger invariant that holds at every point during that phase.

Definition 2.5. *For all $i \in [n-1]$, let Δ_i be the degree of i in the transfer graph at the end of phase i.*

The following stronger invariant of the transfer graph directly implies Lemma 2.1.

Claim 2.2 (Invariant for phase i) *Let f be an input function given to **BubbleSmooth**. At every point during the execution of **BubbleSmooth**(f), for each edge $e_s = (x, y)$ of the transfer graph T,*

$$f(x) - f(y) \geq deg_s(x) + deg_s(y) - 1 + |x - y|.$$

*Moreover, for each phase $i \in [n-1]$, after each execution of **LinePass**(i), for each edge e_s incident on vertex i, the following (stronger) condition holds:*

if the edge $e_s = (i, j)$, i.e., it is outgoing from i, then $f(i) - f(j) \geq \Delta_i + deg_s(j) - 1 + |i - j|$;

if the edge $e_s = (j, i)$, i.e., it is incoming into i, then $f(j) - f(i) \geq \Delta_i + deg_s(j) - 1 + |i - j|$.

Observe that all transfers involving i during phase i are in the same direction: if in the beginning of the phase we have $f(i) > f(i+1)$, then all transfers are from i; if we have $f(i) < f(i+1)$ instead, then all transfers are to i. In particular, whenever an edge incident to i is added, it is not modified subsequently during phase i. So for all s, $deg_s(i)$ never exceeds Δ_i during phase i and the condition in Claim 2.2 is indeed stronger than that in Lemma 2.1. The proof of Claim 2.2 is omitted.

2.3 Matchings of Violated Pairs

Part (2) of Theorem 1.3 states that the ℓ_1 distance between f and **BubbleSmooth**(f) is at most $2 \cdot \ell_1^{Lip}(f)$. By definition of the transfer graph $T = ([n], E)$, the distance $|f - \textbf{BubbleSmooth}(f)|_1 = 2|E|$. Lemma 2.3 shows that $\ell_1^{Lip}(f)$ is bounded below by the violation score of any matching. We complete the proof of Theorem 1.3 by showing that T has a matching with violation score $|E|$.

Lemma 2.3. *Let M be a matching of pairs (x, y), where x and y are in the (discrete) domain of a function f. Then $\ell_1^{Lip}(f) \geq \mathrm{vs}_f(M)$, where $\mathrm{vs}_f(M) = \sum_{(x,y)\in M} \mathrm{vs}_f(x, y)$ is the violation score of M.*

Proof. Let f^* be a closest Lipschitz function to f (on the same domain as f) with respect to the ℓ_1 distance, i.e., $|f - f^*|_1 = \ell_1(f, Lip)$. Consider a pair $(x, y) \in M$. Since $|f(x) - f(y)| = d(x, y) + \mathrm{vs}_f(x, y)$ and $|f^*(x) - f^*(y)| \leq d(x, y)$, it follows by the triangle inequality that $|f(x) - f^*(x)| + |f(y) - f^*(y)| \geq \mathrm{vs}_f(x, y)$. Since M is a matching, we can add over all of its pairs to obtain

$$\begin{aligned} \ell_1(f, Lip) = |f - f^*|_1 &\geq \sum_{(x,y)\in M} (|f(x) - f^*(x)| + |f(y) - f^*(y)|) \\ &\geq \sum_{(x,y)\in M} \mathrm{vs}_f(x, y) = \mathrm{vs}_f(M), \end{aligned}$$

which concludes the proof. □

Now using Lemma 2.1 we exhibit a matching in the final transfer graph which has large violation score, concluding the proof of Theorem 1.3.

Proof (of part (2) of Theorem 1.3). Let $T^* = ([n], E)$ be the final transfer graph corresponding to the execution of **BubbleSmooth** on f and let $E = \{e_1, \ldots, e_t\}$. By definition of the transfer graph, $|f - f'|_1 = \sum_{i \in [n]} deg_t(i) = 2|E|$. By Lemma 2.3, it is enough to show that there is a matching M of pairs violated by f with the violation score $\mathrm{vs}_f(M) \geq |E|$.

We claim that T contains such a matching. It can be constructed greedily by repeating the following step, starting with $s = t$: add e_s to M and then remove e_s and all other edges adjacent to its endpoints from T; set s to be the number of edges remaining in E. In each step, at most $deg_s(x) + deg_s(y) - 1$ are removed from T. ("At most" because T can have multiple edges.) By Lemma 2.1, $\mathrm{vs}_f(x, y) \geq deg_s(x) + deg_s(y) - 1$. So, at each step of the greedy procedure, the violation score of the pair (x, y) added to M is at least the number of edges removed from T. Therefore, $\mathrm{vs}_f(M) \geq |E|$. □

3 Dimension Reduction: Proof of Theorems 1.2 and 1.4

In this section, we explain the main ideas used to prove Theorems 1.2 and 1.4 that connect the distance of a function to being Lipschitz to the distance of its lines to being Lipschitz. Effectively, these results reduce the task of testing a multidimensional function to the task of testing its lines. Our main contribution in this section is a smoothing procedure that makes a function Lipschitz by modifying it at a few points by repairing one dimension at a time. In Definition 3.1, we present the *dimension operator* that repairs all lines in a specified dimension by applying **BubbleSmooth** to each of them. The important properties of the dimension operator are summarized in Lemma 3.1 which is the key ingredient in the proofs of Theorems 1.2 and 1.4. The derivation of Theorems 1.2 and 1.4 from Lemma 3.1 appears in the full version.

Recall from the discussion in Section 1.2 that we denote the set of lines of f along dimension i by L^i_f and the set of all lines of f by $L_f = L^i_f$.

Definition 3.1 (Dimension operator A_i). *Given $f : [n]^d \to \mathbb{Z}$ and dimension $i \in [d]$, the* dimension operator A_i *applies **BubbleSmooth** to every function $g \in L^i_f$ and returns the resulting function.*

Next lemma summarizes the properties of the dimension operator.

Lemma 3.1 (Properties of the dimension operator A_i). *For all $i \in [d]$, the dimension operator A_i satisfies the following properties for every function $f : [n]^d \to \mathbb{Z}$.*

1. *(Repairs dimension i.) Every $g \in L^i_{A_i[f]}$ is Lipschitz.*
2. *(Does not modify the function too much.) $|f - A_i[f]|_1 \leq 2 \cdot \sum_{g \in L^i_f} \ell_1^{Lip}(g)$.*
3. *(Does not spoil other dimensions.) For all $j \neq i$ in $[d]$, it does not increase the expected ℓ_1 distance of a random line in dimension j to the Lipschitz property, i.e., $\mathbb{E}_{g \leftarrow L^j_{A_i[f]}}[\ell_1^{Lip}(g)] \leq \mathbb{E}_{g \leftarrow L^j_f}[\ell_1^{Lip}(g)]$.*

Proof. **Item 1.** Item 1 follows from part (1) of Theorem 1.3.

Item 2. Since the dimension operator A_i operates by applying **BubbleSmooth** to all (disjoint) lines in L^i_f, we get $|f - A_i[f]|_1 = \sum_{g \in L^i_f} |g - \textbf{BubbleSmooth}[g]|_1$. The latter is at most $\sum_{g \in L^i_f} 2 \cdot \ell_1^{Lip}(g)$ by Part (2) of Theorem 1.3, thus proving the item.

Item 3. Fix i and j. First, we give a standard argument [4, 3, 7] that it is enough to prove this statement for $n \times n$ grids. Namely, every $\alpha \in [n]^d$ with $\alpha_i = \alpha_j = 0$ defines a restriction of a function f to an $n \times n$ grid by $h(x_i, x_j) = f(\alpha + x_i \cdot e^i + x_j \cdot e^j)$, where x_i and x_j range over $[n]$. (Recall that $e^i \in [n]^d$ is 1 on the ith coordinate and 0 on the remaining coordinates.) If the item holds for all 2-dimensional grids, we can average over all such grids defined by different α to obtain the statement for the d-dimensional function f. Now fix an arbitrary restriction $h : [n]^2 \to \mathbb{Z}$ as discussed and think of h as an $n \times n$ matrix with rows (resp., columns) corresponding to lines in dimension i (resp., in dimension j).

The key feature of our dimension operator A_i is that it can be broken down into steps, each consisting of one application of the basic step of **BubbleSmooth** to the same positions $(k, k+1)$ on all lines in dimension i. To see this, observe that we can replace the **while** loop condition on Line 2 of Algorithm 2 with "repeat t times", where t should be large enough to guarantee that the line segment under consideration is Lipschitz after t iterations of **LinePass**. (E.g., $t = n \cdot \mathrm{ImgD}(f)$ repetitions suffices.) If this version of **BubbleSmooth** is run synchronously and in parallel on all lines in dimension i, the the basic step will be applied to the same positions $(k, k+1)$ on all lines.

Since in each parallel update step only two adjacent columns of h are affected, it is sufficient to prove the item for two adjacent columns of h. Accordingly, consider two adjacent columns C_1 and C_2 of h. Let M_1 and M_2 be Lipschitz columns that are closest in the ℓ_1 distance to C_1 and C_2, respectively. Thus, $\ell_1^{Lip}(C_1) = |C_1 - M_1|_1$ and $\ell_1^{Lip}(C_2) = |C_2 - M_2|_1$. Let C'_1 and C'_2 be the columns of the matrix resulting from applying the basic operator to the rows of the matrix (C_1, C_2). Similarly, define M'_1 and M'_2 to be the columns of the matrix resulting from applying the basic operator to the rows of (M_1, M_2). We prove in the full version that applying the basic operator to the rows of a matrix consisting of two Lipschitz columns results in a matrix whose columns are still Lipschitz, that is, M'_1 and M'_2 are Lipschitz. Therefore, $\ell_1^{Lip}(C'_1) \leq |C'_1 - M'_1|_1$ and $\ell_1^{Lip}(C'_2) \leq |C'_2 - M'_2|_1$. Finally, using the inequality $|C'_1 - M'_1|_1 + |C'_2 - M'_2|_1 \leq |C_1 - M_1|_1 + |C_2 - M_2|_1$ whose proof is deferred to the full version, the proof of Item 3 is completed as follows: $\ell_1^{Lip}(C_1) + \ell_1^{Lip}(C_2) = |C_1 - M_1|_1 + |C_2 - M_2|_1 \geq |C'_1 - M'_1|_1 + |C'_2 - M'_2|_1 \geq \ell_1^{Lip}(C'_1) + \ell_1^{Lip}(C'_2)$. □

4 Algorithms for Testing the Lipschitz Property on Hypergrids

In this section, we present our testers for the Lipschitz property of functions $f : [n]^d \to \mathbb{Z}$. Theorem 1.2 relates the distance of a function f from the Lipschitz property to the (expected) distance of its lines to this property. The resulting bound, however, depends on the image diameter of f. The image diameter is small (at most nd) for Lipschitz functions, but can be arbitrarily large otherwise. The high-level description of our testers is the following: *(i) estimate the image diameter of f and **reject** if it is too large; (ii)*

repeatedly sample a line g of f at random, run one step of a Lipschitz tester for the line on g and ***reject*** *if a violated pair is discovered; otherwise,* ***accept****.* Step (i) ensures that a small sample of lines is enough to succeed with constant probability. The testers differ only in one parameter which quantifies what "too large" means in Step (i).

4.1 Estimating the Effective Image Diameter

As mentioned before, a Lipschitz function on $[n]^d$ has image diameter at most nd, which can serve as a threshold for rejection in Step (i) of the informal procedure above. However (if we are willing to tolerate two-sided error), it is sufficient to use a smaller threshold, equal the *effective* diameter of the function. For a given $\epsilon \in (0,1]$, define $\mathrm{ImgD}_\epsilon(f)$ as the smallest value α such that f is ϵ-close to having image diameter α:

$$\mathrm{ImgD}_\epsilon(f) = \min_{U \subseteq [n]^d : |U| \geq (1-\epsilon)n^d} \{\max_{x \in U} f(x) - \min_{x \in U} f(x)\}.$$

Although the image diameter of a Lipschitz function f can indeed achieve value nd, the effective $\mathrm{ImgD}_\epsilon(f)$ is upper bounded by the potentially smaller quantity $O(n\sqrt{d \ln(1/\epsilon)})$. The next lemma makes this precise, and follows directly from McDiarmid's inequality.

Lemma 4.1 (Effective image diameter). *For all $\epsilon \in (0,1]$, each Lipschitz function $f : [n]^d \to \mathbb{R}$ is $(\epsilon/21)$-close to having image diameter at most $n\sqrt{d \ln(42/\epsilon)}$.*

Our testers use estimates of image diameter or effective diameter to reject functions. The next lemma, proved in the full version, shows that we can get such estimates efficiently. An algorithm satisfying parts (i) and (ii) of the lemma was obtained in [7].

Lemma 4.2. *There is a randomized algorithm* SAMPLE-DIAMETER *that, given a function $f : [n]^d \to \mathbb{R}$ and $\epsilon \in (0,1]$, outputs an estimate $r \in \mathbb{R}$ such that: (i) $\mathrm{ImgD}_\epsilon(f) \leq r$ with probability at least $5/6$; (ii) $r \leq \mathrm{ImgD}(f)$ (always) and (iii) $r \leq \mathrm{ImgD}_{\epsilon/21}(f)$ with probability at least $2/3$. Moreover, the algorithm runs in time $O(1/\epsilon)$.*

4.2 Tester for Hypergrid Domains

Our tester for functions on hypergrids uses a tester for functions on lines from [7].

Lemma 4.3 (Full version of [7]). *Consider a function $g : [n] \to \mathbb{R}$ and $r \geq \mathrm{ImgD}(g)$. Then there is a 1-sided error algorithm* LINE-TESTER *which on input g and r rejects with probability at least $\frac{\epsilon^{Lip}(g)}{6 \log \min\{r,n\}}$.*

To analyze our testers, we also need to estimate the probability that a random line $g \leftarrow L_f$ is rejected by LINE-TESTER(g,r) with $r \geq \mathrm{ImgD}_{\epsilon/2}(f)$. Such bound r will be obtained via Lemma 4.2. Since r may be much smaller than $\mathrm{ImgD}(f)$, Lemma 4.3 does not apply directly. Nevertheless, the next lemma (proved in the full version) shows how to circumvent this difficulty.

Lemma 4.4. *Let $f : [n]^d \to \mathbb{Z}$ be ϵ-far from Lipschitz. Consider a real $r \geq \mathrm{ImgD}_{\epsilon/2}(f)$. For a random line $g \leftarrow L_f$, the probability that* LINE-TESTER(g,r) *rejects is at least $\frac{\epsilon}{24dr \log \min\{r,n\}}$.*

Algorithm 3 presents our tester for the Lipschitz property on hypergrid domains. One of its inputs is a threshold R for rejection in Step 1. The testers in Theorem 1.1 are obtained by setting R appropriately.

Algorithm 3. Tester for Lipschitz property on hypergrid.

input : function $f : [n]^d \to \mathbb{Z}$, $\epsilon \in (0, 1]$, and value $R \in \mathbb{R}$

1 Let $r \leftarrow$ SAMPLE-DIAMETER$(f, \epsilon/2)$. If $r > R$, **reject**.
2 **for** $i = 1$ **to** $\ell = \frac{48d \cdot r \log \min\{r,n\}}{\epsilon}$ **do**
3 Select a line g uniformly from L_f and **reject** if LINE-TESTER(g, r) does.
4 **Accept**.

Proof (of Theorem 1.1). We claim that Algorithm 3 run with $R = nd$ (respectively, $R = n\sqrt{d \ln(84/\epsilon)}$) gives the tester in part (1) (respectively, part (2)) of Theorem 1.1. Suppose that the input function f is Lipschitz. When $R = nd$, the algorithm accepts f with probability 1; when $R = n\sqrt{d \ln(84/\epsilon)}$, Lemmas 4.2 and 4.1 guarantee that it accepts with probability at least $2/3$. Now suppose that f is ϵ-far from Lipschitz. Conditioning on the event that $r \geq \mathrm{ImgD}_{\epsilon/2}(f)$ (which holds with probability at least $5/6$ by Lemma 4.2), we get from Lemma 4.4 that f is rejected with probability at least $4/5$ in Step 3. Removing the conditioning gives that f is rejected with probability at least $2/3$ (regardless of R). Further details and the analysis of the running time are omitted. □

References

[1] Awasthi, P., Jha, M., Molinaro, M., Raskhodnikova, S.: Testing Lipschitz functions on hypergrid domains. Electronic Colloquium on Computational Complexity (ECCC) TR12-076 (2012)

[2] Chakrabarty, D., Seshadhri, C.: Optimal bounds for monotonicity and Lipschitz testing over the hypercube. Electronic Colloquium on Computational Complexity (ECCC) TR12-030 (2012)

[3] Dodis, Y., Goldreich, O., Lehman, E., Raskhodnikova, S., Ron, D., Samorodnitsky, A.: Improved Testing Algorithms for Monotonicity. In: Hochbaum, D.S., Jansen, K., Rolim, J.D.P., Sinclair, A. (eds.) RANDOM-APPROX 1999. LNCS, vol. 1671, pp. 97–108. Springer, Heidelberg (1999)

[4] Goldreich, O., Goldwasser, S., Lehman, E., Ron, D., Samorodnitsky, A.: Testing monotonicity. Combinatorica 20(3), 301–337 (2000)

[5] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. J. ACM 45(4), 653–750 (1998)

[6] Gromov, M.: Metric Structures for Riemannian and non-Riemannian Spaces (1999)

[7] Jha, M., Raskhodnikova, S.: Testing and reconstruction of Lipschitz functions with applications to data privacy. In: IEEE FOCS, pp. 433–442 (2011) full version available at, http://eccc.hpi-web.de/report/2011/057/

[8] Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. SIAM J. Comput. 25(2), 252–271 (1996)

Extractors for Polynomials Sources over Constant-Size Fields of Small Characteristic

Eli Ben-Sasson[1,⋆] and Ariel Gabizon[2,⋆⋆]

[1] Department of Computer Science, Technion, Haifa, Israel and Microsoft Research New-England, Cambridge, MA
[2] Department of Computer Science, Technion, Haifa, Israel

Abstract. A polynomial source of randomness over $\mathbb{F}_q^n$ is a random variable $X = f(Z)$ where f is a polynomial map and Z is a random variable distributed uniformly on $\mathbb{F}_q^r$ for some integer r. The three main parameters of interest associated with a polynomial source are the field size q, the (total) degree D of the map f, and the "rate" k which specifies how many different values does the random variable X take, where rate k means X is supported on at least q^k different values. For simplicity we call X a (q, D, k)-source.

Informally, an extractor for (q, D, k)-sources is a deterministic function $E : \mathbb{F}_q^n \to \{0,1\}^m$ such that the distribution of the random variable $E(X)$ is close to uniform on $\{0,1\}^m$ for any (q, D, k)-source X. Generally speaking, the problem of constructing deterministic extractors for such sources becomes harder as q and k decrease and as D grows larger.

The only previous work of [Dvir et al., FOCS 2007] construct extractors for such sources when $q \gg n$. In particular, even for $D = 2$ no constructions were known for any fixed finite field.

In this work we construct for the first time extractors for (q, D, k)-sources for constant-size fields. Our proof builds on the work of DeVos and Gabizon [CCC 2010] on extractors for affine sources, with two notable additions (described below). Like [DG10], our result makes crucial use of a theorem of Hou, Leung and Xiang [J. Number Theory 2002] giving a lower bound on the dimension of products of subspaces. The key insights that enable us to extend these results to the case of polynomial sources of degree D greater than 1 are

1. A source with support size q^k must have a linear span of dimension at least k, and in the setting of low-degree polynomial sources it suffices to increase the dimension of this linear span.
2. Distinct Frobenius automorphisms of a (single) low-degree polynomial source are 'pseudo-independent' in the following sense: Taking the product of distinct automorphisms (of the very same source) increases the dimension of the linear span of the source.

⋆ The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 240258.

⋆⋆ The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 240258.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 399–410, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1 Introduction

This paper is part of a long and active line of research devoted to the problem of "randomness extraction": Given a family of distributions all guaranteed to have a certain structure, devise a method that can convert a sample from any distribution in this family to a sequence of uniformly distributed bits — or at least a sequence *statistically close* to the uniform distribution. Usually, it is easy to prove that a random function is, with high probability, a good extractor for the given family, and the challenge is to give an explicit construction of such an extractor.

The first example of a randomness extraction problem was given by von-Neumann [20], who gave an elegant solution to the following problem: How can a biased coin with unknown bias be used to generate 'fair' coin tosses? In this case the input distribution consists of independent identically distributed bits which makes the extraction task simpler. Since then many families of more complex distributions were studied. Also, the concept of randomness extraction has proven to be useful for various applications. The reader is referred to the introduction of [9] for more details on the classes of distributions studied, references and motivation.

1.1 Polynomial Sources

In this paper we construct extractors for *polynomial sources* — distributions that are sampled by applying low-degree polynomials to uniform inputs as defined next. Throughout this paper if Ω is a finite set we let U_Ω denote the uniform distribution on Ω.

Definition 1 (Polynomial sources and extractors). *Fix integers n, d, k with $k \leq n$ and a field $\mathbb{F}_q$. We define $\mathcal{M}[n,d,k]$ to be the set of mappings $f : \mathbb{F}_q^r \mapsto \mathbb{F}_q^n$, where r is an integer counting the number of inputs to the source and*

$$f(Z_1,\ldots,Z_r) = (f_1(Z_1,\ldots,Z_r),\ldots,f_n(Z_1,\ldots,Z_r))$$

such that

- *for every $i \in [n]$, f_i is a polynomial in $\mathbb{F}_q[Z_1,\ldots,Z_r]$ of individual degree at most d.*
- *The range, or support, of f is of size at least q^k. Formally,*

$$|\{f(z_1,\ldots,z_r) \mid (z_1,\ldots,z_r) \in \mathbb{F}_q^r\}| \geq q^k.$$

*A (n,k,d)-*polynomial source *is a distribution of the form $f(U_{\mathbb{F}_q^r})$ for some and $f \in \mathcal{M}[n,k,d]$ with r inputs. (When the parameters n,k,d are clear from context we shall omit them and, simply, use the term "polynomial source".)*

*Let Ω be some finite set. A function $E : \mathbb{F}_q^n \mapsto \Omega$ is a (k,d,D,ϵ)-*polynomial source extractor *if for every $f \in \mathcal{M}[n,d,k]$ of total degree at most D and r*

inputs, $E(f(U_{\mathbb{F}_q^r}))$ *is* ϵ*-close to uniform, where a distribution* P *on* Ω *is* ϵ-close to uniform *if for every* $A \subseteq \Omega$

$$|\Pr_{x \leftarrow P}(x \in A) - |A|/|\Omega|| \leq \epsilon.$$

Remark 1. A few words are in order regarding the above definitions.

- The number of inputs used by our source — denoted by r in the definitions above — does not affect the parameters of our extractors hence we omit this parameter from the definition of polynomial sources and extractors.
- In the context of extractors what might have seemed more natural is to require the distribution $f(U_{\mathbb{F}_q^r})$ to have *min-entropy*[1] $k \cdot \log q$. Our requirement on the size of the range of f is weaker, and suffices for our construction to work.
- Individual degree plays a larger role than total degree in our results. In fact, the first stage of our construction — constructing a non-constant polynomial over $\mathbb{F}_q$- requires a field of size depending *only* on individual degree. This is why it is more convenient to limit individual degree and not total degree in the definition of $\mathcal{M}[n, d, k]$.

Motivation. To motivate our study of extractors for polynomials sources, we mention four distinct applications of such extractors for the simplest class of sources — affine ones, in which the degree of the source is 1 (see definition below). Demenkov and Kulikov [8] showed, using elementary methods, that any circuit over the full binary basis that computes an affine disperser for min-entropy rate $o(1)$ must contain at least $3n(1-o(1))$ gates, and this matches the previous best circuit lower bound of Blum from 1984 [4]. Another application of affine extractors was given by Viola [19] and independently by De and Watson [7] showing how to use them to construct extractors for bounded depth circuits. A third application was given by Ben-Sasson and Zewi [3] who showed how to construct two-source extractors and bipartite Ramsey graphs from affine extractors. Recent work of Guruswami [14] and of Dvir and Lovett [12] use "subspace evasive functions" which are closely related to affine extractors to get better algorithms for list-decoding of folded Reed-Solomon codes. These applications lead us to believe that extractors for general low-degree sources of the kind defined next will similarly be useful in other branches of computational complexity.

1.2 Previous Work and Our Result

Polynomial source extractors are a generalization of affine source extractors — where the source is sampled by a degree one map. There has been much work recently on affine source extractors [1,5,22,13,9,16] and related objects called

[1] The min-entropy of a distribution P is the largest ℓ such that for every fixed x, $\Pr(P = x) \leq 2^{-\ell}$. This is the standard measure of randomness in the context of extractors originating from Chor and Goldreich [6].

affine source dispersers [2,18] where the output is required to be non-constant but not necessarily close to uniform.

Turning to extractors for non-affine, low-degree sources, the only previous work is by Dvir, Gabizon and Wigderson [11], and it requires large fields. In particular, to extract a single bit [11] needs a field of size at least n^c where $c > 1$ is a constant and n is number of inputs to the extractor, i.e., the number of outputs of the polynomial source. (In a related albeit different vein, Dvir [10] constructed extractors for distributions that are uniform over low-degree algebraic varieties, which are sets of common zeros of a system of low-degree multivariate polynomials.)

In this work we construct polynomial source extractors over much smaller fields than previously known, assuming the characteristic of the field is significantly smaller than the field size.

Theorem 1 (Main — Extractor). *Fix a field $\mathbb{F}_q$ of characteristic p, integers $d, D, 4 \leq k \leq n$ where $n \geq 25$, and a positive integer $m < 1/2 \cdot \log_p q$. Let $\alpha = 3D \cdot (p \cdot d)^{3n/k}$. Assume that $q \geq 2 \cdot \alpha^2$. There is an explicit (k, d, D, ϵ)-polynomial source extractor $E : \mathbb{F}_q^n \mapsto \mathbb{F}_p^m$ with error $\epsilon = p^{m/2} \cdot \alpha \cdot q^{-1/2}$.*

In particular, when $D, n/k$ and p are constant we get a polynomial source extractor for constant field size. We state such an instantiation.

Corollary 1 (Extractor for quadratic sources of min-entropy rate half over fields of characteristic 2). *There is a universal constant C such that the following holds. For any $\epsilon > 0$ and any $q > C/\epsilon^2$ which is a power of 2, there is an explicit $(n/2, 2, 2, \epsilon)$-polynomial source extractor $E : \mathbb{F}_q^n \mapsto \{0,1\}$.*

Non-boolean dispersers for smaller fields. Along the way of our proof we construct a weaker object called a *non-boolean disperser*. A non-boolean disperser maps the source into a relatively small (but not $\{0,1\}$) domain and guarantees the output is non-constant. The advantage of this part of the construction is that it works for smaller fields than the extractor, and moreover, the field size for which it works depends only on the *individual* degrees of the source polynomials. In the theorem and corollary below we use an implicit isomorphism of $\mathbb{F}_q^n$ and $\mathbb{F}_{q^n}$. See an explanation of this in the beginning of Section 3.

Theorem 2 (Main — Disperser). *Fix a prime power $q = p^\ell$. Fix integers $k \leq n$ and $d < s$ such that n is prime and s is a power of p. Fix a non-trivial $\mathbb{F}_q$-linear map $T : \mathbb{F}_q^n \mapsto \mathbb{F}_q$. Let $u = \lceil (n-k)/(k-1) \rceil$. Define $P : \mathbb{F}_q^n \mapsto \mathbb{F}_q$ by $P(x) \triangleq T(x^{1+s+s^2+\ldots+s^u})$. Assume that $q > d \cdot \frac{s^{u+1}-1}{s-1}$. Then, for any $f(\boldsymbol{Z}) = f(Z_1, \ldots, Z_r) \in \mathcal{M}[n, k, d]$, $P(f(\boldsymbol{Z}))$ is a non-constant function from $\mathbb{F}_q^r$ into $\mathbb{F}_q$.*

We instantiate this result for the smallest field it works for — $\mathbb{F}_4$.

Corollary 2 (Disperser for min-entropy rate half over $\mathbb{F}_4$). *Let n be prime. Define the function $P : \mathbb{F}_4^n \mapsto \mathbb{F}_4$ as follows. Think of the input x as an element of $\mathbb{F}_{4^n}$ and compute x^3. Now output the first coordinate of the*

vector x^3. *Then for any* $f \in \mathcal{M}[n, \lceil n/2+1 \rceil, 1]$ — *that is any multilinear* $f \in \mathbb{F}_{4^n}[Z_1, \ldots, Z_r]$ *that has support size at least* $4^{\lceil n/2+1 \rceil}$, *the polynomial* $P(f(Z_1, \ldots, Z_r))$ *is a non-constant function from* $\mathbb{F}_4^r$ *into* $\mathbb{F}_4$.

2 Overview of the Proof

Our goal is to describe an explicit function $E : \mathbb{F}_q^n \to \{0,1\}^m$ such that for any (n,k,d)-polynomial source X we have that $E(X)$ is ϵ-close to the uniform distribution on $\{0,1\}^m$ and we do this in two steps. First we construct a function E_0, called a *non-boolean disperser*, that is guaranteed to be non-constant on X, i.e., such that the distribution $Y = E_0(X)$ has support size greater than 1. This part is done in Section 4. Then we apply a second function E_1 to the output of E_0 and prove that the distribution $E_1(Y) = E_1(E_0(X))$ is ϵ-close to uniform. This "disperser–to–extractor" part is described in the full version. We now informally describe the two functions assuming for simplicity the field $\mathbb{F}_q$ is of characteristic 2 and that n is prime. Before starting let us recall the notion of a Frobenius automorphism. If $\mathbb{K}$ is a finite field of characteristic 2 then the mapping

$$\sigma_i : \mathbb{K} \to \mathbb{K}, \quad \sigma_i(z) = z^{2^i}$$

is a *Frobenius automorphism of* $\mathbb{K}$ *over* $\mathbb{F}_2$. (These mappings can be defined over larger fields as well, cf. Section 3.2.) The three elementary properties of this mapping that we use below are first its $\mathbb{F}_2$-*linearity* — that $\sigma_i(a+b) = \sigma_i(a) + \sigma_i(b)$, second its *distinctness*, i.e., that if $\mathbb{K}$ is an extension of $\mathbb{F}_2$ of degree at least t and $0 \le i < j \le t-1$ then σ_i and σ_j are different, and third its *dimension-preservation:* If $\mathbb{K} \supset \mathbb{F}_q \supset \mathbb{F}_2$ then $A \subset \mathbb{K}$ and $\sigma_i(A) \triangleq \{\sigma_i(a) \mid a \in A\}$ span spaces of equal dimension over $\mathbb{F}_q$ (see Claim 3.2).

A different view on low-degree sources. The first part of our analysis uses a somewhat nonstandard view of low-degree sources that we need to highlight. The random variable X ranges over $\mathbb{F}_q^n$ and is the output of n degree-d polynomials over $\mathbb{F}_q$. Let $\mathbb{F}_q^{\le d}[Z_1, \ldots, Z_r]$ denote the set monomials over $\mathbb{F}_q$ of individual degree at most d where $d < q$. (We use Z variables to denote inputs of the polynomial source and X variables for its output.) Suppose the ith coordinate of X is

$$X_i = P^{(i)}(Z_1, \ldots, Z_r) = \sum_{M \in \mathbb{F}_q^{\le d}[Z_1, \ldots, Z_r]} a_M^{(i)} \cdot M(Z_1, \ldots, Z_r)$$

where $a_M^{(i)} \in \mathbb{F}_q$ and $Z_1, \ldots, Z_r$ are independent random variables distributed uniformly over $\mathbb{F}_q$. Applying an $\mathbb{F}_q$-linear bijection $\phi : \mathbb{F}_q^n \to \mathbb{F}_{q^n}$, let $a_M = \phi(a_M^{(1)}, \ldots, a_M^{(n)})$ denote the sequence of coefficients of the monomials M, viewed now as a single element in $\mathbb{F}_{q^n}$. Our nonstandard view is that our source is

$$X = P(Z_1, \ldots, Z_r) = \sum_{M \in \mathbb{F}_q^{\le d}[Z_1, \ldots, Z_r]} a_M \cdot M(Z_1, \ldots, Z_r) \tag{1}$$

where the coefficients a_M and the random variable X come from the "large" field $\mathbb{F}_{q^n}$ but the random variables $Z_1, \ldots, Z_r$ still range over the "small" field $\mathbb{F}_q$. This large-field-small-field view will be important in what comes next. In particular, we shall use the following claim which reduces the problem of constructing a non-boolean disperser to that of constructing a polynomial whose coefficients span $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.

Claim (Full-span polynomials are non-constant coordinate-wise). Suppose P has individual degree smaller than q. If the set of coefficients $A = \{a_M \mid \deg(M) > 0\}$ appearing in (1) spans $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ then $X_i = P^{(i)}(Z_1, \ldots, Z_r)$ is a non-constant function for every $i \in \{1, \ldots, n\}$.

Proof. By way of contradiction. If $P^{(i)}$ is constant on $\mathbb{F}_q^r$ and has individual degrees smaller than q, then all its nonzero coefficients are zero in which case A spans a strict subspace of $\mathbb{F}_{q^n}$.

Non-boolean disperser. We start with the simplest nontrivial case to which our techniques apply and construct a non-boolean disperser for homogeneous multilinear quadratic sources with min-entropy rate greater than half over the finite field with 4 elements (this is a special case of Corollary 2). Using $\binom{r}{2}$ to denote the set $\{(i,j) \mid 1 \leq i < j \leq r\}$ and writing X as in (1) we get

$$X = \sum_{(i,j) \in \binom{r}{2}} a_{ij} Z_i Z_j, \quad a_{ij} \in \mathbb{F}_{4^n} \tag{2}$$

where $Z_1, \ldots, Z_r$ are uniformly and independently distributed over $\mathbb{F}_4$ and X has support of size greater than $4^{n/2}$. Let

$$A = \left\{ a_{ij} \mid (i,j) \in \binom{[r]}{2} \right\} \tag{3}$$

denote the set of coefficients appearing in (2). In light of Claim 2 it suffices to construct E_0 such that $E_0(X)$, when written as a polynomial over $Z_1, \ldots, Z_r$, has a set of coefficients that spans $\mathbb{F}_{4^n}$ over $\mathbb{F}_4$. (Then we "project" this polynomial onto, say, the first coordinate and get a non-constant function mapping into $\mathbb{F}_4$, i.e., a non-boolean disperser.)

To do this we take the approach of DeVos and Gabizon [9] which uses the theorem of Hou, Leung and Xiang [15]. Assuming n is prime, this theorem implies that if $A, B \subset \mathbb{F}_{q^n}$ are sets spanning spaces of respective dimensions d_1, d_2 over $\mathbb{F}_q$, then the set of products

$$A \cdot B \triangleq \{a \cdot b \mid a \in A, b \in B\}$$

spans a subspace of $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ of dimension at least $\min\{n, d_1 + d_2 - 1\}$. Returning to our case and taking A as in (3), our first observation is that $\dim(\text{span}(A)) > n/2$ because X is contained in $\text{span}(A)$. So the theorem of [15] mentioned above implies that $\text{span}(A \cdot A) = \mathbb{F}_{4^n}$. Consider what would happen if we could sample *twice* from X independently and take the product of the

two samples in $\mathbb{F}_{4^n}$. Using $X', Z'_1, \ldots, Z'_r$ to express the second sample we write this product as

$$X \cdot X' = \left(\sum_{(i,j) \in \binom{r}{2}} a_{ij} Z_i Z_j \right) \cdot \left(\sum_{(i',j') \in \binom{r}{2}} a_{i'j'} Z'_i Z'_j \right).$$

Opening the right-hand-side as a polynomial in $Z_1, \ldots, Z_r, Z'_1, \ldots, Z'_r$ we see that its set of coefficients is $A \cdot A$ which spans $\mathbb{F}_{4^n}$ over $\mathbb{F}_4$, as desired[2].

Unfortunately we only have access to a *single* sample of X and have to make use of it. We use the fact that $\mathbb{F}_4$ is a degree 2 extension of a smaller field ($\mathbb{F}_2$) and hence has two distinct Frobenius automorphisms. And here comes our second observation: Taking the product of 2 distinct Frobenius automorphisms of a *single* sample of X has a similar effect to that of taking two independent samples of X! Indeed, take the product of $\sigma_0(X)$ and $\sigma_1(X)$ and, using the linearity of Frobenius mapping, expand as

$$\begin{aligned} X \cdot X^2 &= \left(\sum_{(i,j) \in \binom{r}{2}} a_{ij} Z_i Z_j \right) \cdot \left(\sum_{(i',j') \in \binom{r}{2}} a_{ij}^2 Z_i^2 Z_j^2 \right) \\ &= \sum_{(i,j),(i',j') \in \binom{r}{2}} a_{ij} a_{i'j'}^2 Z_i Z_j Z_{i'}^2 Z_{j'}^2. \end{aligned}$$

The main point is that every element in the set of products of A and $A^2 \triangleq \{a^2 \mid a \in A\}$ appears as the coefficient of a monomial in the polynomial above and these monomials are distinct over $\mathbb{F}_4$. And the dimension-preservation of σ_1 implies that $\dim(\mathrm{span}(A^2)) = \dim(\mathrm{span}(A)) > n/2$. Consequently, the theorem of [15] implies that $A \cdot A^2$ spans $\mathbb{F}_{4^n}$ over $\mathbb{F}_4$, so by Claim 2 the function $E_0(X)$, which outputs the first coordinate of $X \cdot X^2$, is non-constant for X and this completes the sketch of our non-boolean disperser for the special case of homogenous, quadratic, multilinear polynomials over $\mathbb{F}_4$.

To extend this argument to general polynomial sources of individual degree $\leq d$ we carefully select a set of t distinct Frobenius automorphisms $\sigma_{i_0}, \ldots, \sigma_{i_{t-1}}$ (assuming $\mathbb{F}_q$ is an extension-field of degree at least t) such that the mapping $f : (\mathbb{F}_q^{\leq d}[Z_1, \ldots, Z_r])^t \to \mathbb{F}_q[Z_1, \ldots, Z_r]$ given by

$$f(M_0, \ldots, M_{t-1}) = \prod_{j=0}^{t-1} \sigma_{i_j}(M_j) \mod (Z_1^q - Z_1, \ldots, Z_r^q - Z_r)$$

is injective. Then we argue, just as in the case above, that the function $g(X) \triangleq \prod_{j=0}^{t-1} \sigma_{i_j}(X)$ expands to a sum of distinct monomials with coefficients ranging over the product set $\hat{A} = \sigma_{i_0}(A) \cdots \sigma_{i_{t-1}}(A)$ where $\sigma(A) = \{\sigma(a) \mid a \in A\}$. The

[2] The same argument would work as well over the two-element field $\mathbb{F}_2$. The extension field is needed to deal with the case of a single source as explained next.

theorem of [15] is applied t times to conclude that $\hat{A}$ spans $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. Now we apply Claim 2 and get that the first coordinate of $g(X)$ (viewing $g(X)$ as a tuple of n polynomials over $\mathbb{F}_q$) is a non-constant function. Details are provided in Section 4.

From dispersers to extractors. This part is based on the work of Gabizon and Raz [13] and uses an important theorem of Weil [21]. This theorem implies the following. Suppose we evaluate a polynomial $g \in \mathbb{F}_q[Z_1, \ldots, Z_r]$ of small-enough degree $\deg(g) < \sqrt{q}$ on a uniformly random sample in $\mathbb{F}_q^r$ and then take the first bit of this evaluation (when viewing it as a vector over $\mathbb{F}_2$). Then, this bit will either be constant — we then say g is "degenerate" — or close to the uniform distribution. Assuming our source is low-degree and the field size q is sufficiently large we can argue that $\deg(E_0(X)) < \sqrt{q}$ because X is low-degree by assumption and E_0 is low-degree by construction. So to apply Weil's Theorem and get an extractor we only need to ensure that we have in hand a non-degenerate polynomial. Alas, we have relatively little control over the polynomial source so need to transform it somehow into a non-degenerate one in a black-box manner. Here we apply another observation, its proof is due to Swastik Kopparty, which says that $(E_0(X))^v$ is non-degenerate for odd[3] $v > 2$. This part is explained in the full version. So we take $E_1(Y)$ to be the first[4] bit of Y^3 and using this observation and Weil's Theorem conclude that $E_1(E_0(X))$ is close to uniform. Analysis of the resulting extractor is given in the full version.

3 Preliminaries

Notation: When we discuss identities between polynomials we only mean identities as *formal polynomials*. We will frequently alternate between viewing $\mathbf{x} \in \mathbb{F}_q^n$ as an element of either $\mathbb{F}_q^n$ or the field $\mathbb{F}_{q^n}$. When we do this we assume it is using an implicit bijective map $\phi : \mathbb{F}_q^n \mapsto \mathbb{F}_{q^n}$ that is an isomorphism of vector spaces. That is, $\phi(t_1 \cdot a_1 + t_2 \cdot a_2) = t_1 \cdot \phi(a_1) + t_2 \cdot \phi(a_2)$ for any $t_1, t_2 \in \mathbb{F}_q$ and $a_1, a_2 \in \mathbb{F}_q^n$. Such ϕ is efficiently computable using standard representations of $\mathbb{F}_{q^n}$. (For details see for example the book of Lidl and Niederreiter [17].) For a set Ω we denote by U_Ω the uniform distribution on Ω.

3.1 Dimension Expansion of Products

Recall that $\mathbb{F}_{q^n}$ is a vector space over $\mathbb{F}_q$ isomorphic to $\mathbb{F}_q^n$. For a set $A \subseteq \mathbb{F}_{q^n}$ we denote by $\dim(A)$ the dimension of the $\mathbb{F}_q$-span of A. For sets $A, B \subseteq \mathbb{F}_{q^n}$ let $A \cdot B \triangleq \{a \cdot b \mid a \in A, b \in B\}$. Hou, Leung and Xiang [15] show that such products expand in dimension. The following theorem is a corollary of Theorem 2.4 of [15].

[3] For characteristic $p > 2$ the criteria for v is a bit different: we need $p \nmid v$.
[4] In fact, we can output several bits. See the full version for details.

Theorem 3 (Dimension expansion of products). *Let $\mathbb{F}_q$ be any field, and let n be prime.*[5] *Let A and B be non-empty subsets of $\mathbb{F}_{q^n}$ such that $A, B \neq \{0\}$. Then*

$$\dim(A \cdot B) \geq \min\{n, \dim(A) + \dim(B) - 1\}$$

In particular, if $A_1, \ldots, A_m$ are non-empty subsets of $\mathbb{F}_{q^n}$ such that for all $1 \leq i \leq m$, $dim(A_i) \geq k$ for some $k \geq 1$. Then

$$\dim(A_1 \cdots A_m) \geq \min\{n, k \cdot m - (m-1)\}.$$

Remark 2. The definition of $A \cdot B$ is somewhat different from that in [15] where it is defined only for subspaces, and as the *span* of all possible products. The definition above will be more convenient for us. It is easy to see that Theorem 2.4 of [15] implies the theorem above with our definition. For clarity, we give a self-contained proof in the full version.[6]

3.2 Frobenius Automorphisms of $\mathbb{F}_q$

Let $q = p^\ell$ for prime p and let $i \geq 0$ be an integer. Raising to power p^i in $\mathbb{F}_q$ is known as a Frobenius automorphism of $\mathbb{F}_q$ over $\mathbb{F}_p$ and will play an important role. We record two useful and well-known properties of this automorphism that will be used in our proofs.

- **Linearity:** $\forall a, b \in \mathbb{F}_q$, $(a+b)^{p^i} = a^{p^i} + b^{p^i}$.
- **Bijection:** The map $x \mapsto x^{p^i}$ over $\mathbb{F}_q$ is bijective. In particular, for $c \in \mathbb{F}_q$, c^{1/p^i} is always (uniquely) defined.

A useful fact following from these properties is that 'taking the p'th power' of a set does not change its dimension.

Claim (Dimension preservation). Let $q = p^\ell$ from prime p and an integer ℓ. For an integer $i \geq 1$ and a set $A \subseteq \mathbb{F}_{q^n}$ let $A^{p^i} \triangleq \{a^{p^i} \mid a \in A\}$. Then $\dim(A) = \dim(A^{p^i})$.

See the full version for a proof of the claim.

4 The Main Construction

As before, we use r to denote the number of inputs of $f(Z_1, \ldots, Z_r) \in \mathcal{M}[n, d, k]$. We denote by $\mathcal{D}$ the product set $\{0, \ldots, d\}^r$. We use bold letters to denote variables that are vectors in $\mathbb{F}_q^r$. For example, $\mathbf{Z} = (Z_1, \ldots, Z_r)$. For an element $S = (s_1, \ldots, s_r) \in \mathcal{D}$ we use the notation

$$\mathbf{Z}^S \triangleq Z_1^{s_1} \cdots Z_r^{s_r}.$$

[5] The theorem of [15] works also for non-prime n in which case the inequality involves the size of a certain subfield of $\mathbb{F}_{q^n}$.

[6] Also, see Section 3.2 of [9] for a self-contained proof using the definition of [15].

Fix $f = (f_1(\mathbf{Z}), \ldots, f_n(\mathbf{Z})) \in \mathcal{M}[n, d, k]$. For $1 \leq j \leq n$, we write

$$f_j(\mathbf{Z}) = \sum_{S \in \mathcal{D}} a_{j,S} \cdot \mathbf{Z}^S.$$

With the notation above, for $S \in \mathcal{D}$ let $a_S \triangleq (a_{1,S}, \ldots, a_{n,S}) \in \mathbb{F}_q^n$. Using the isomorphism of the vectors spaces $\mathbb{F}_q^n$ and $\mathbb{F}_{q^n}$, we can view a_S as an element of $\mathbb{F}_{q^n}$ and write

$$f(\mathbf{Z}) = \sum_{S \in \mathcal{D}} a_S \cdot \mathbf{Z}^S. \tag{4}$$

That is, we view f as a multivariate polynomial with coefficients in $\mathbb{F}_{q^n}$. A crucial observation is that when f has large support the coefficients of f have large dimension.

Lemma 1 (Large support implies large span). *Let $f \in \mathcal{M}[n, d, k]$. As in (4), write $f(\boldsymbol{Z}) = \sum_{S \in \mathcal{D}} a_S \cdot \boldsymbol{Z}^S$ where $a_S \in \mathbb{F}_{q^n}$. Then $\dim\{a_S\}_{S \in \mathcal{D} \setminus \{\mathbf{0}\}} \geq k$.*

Proof. The range of f over inputs in $\mathbb{F}_q^r$ is contained in an affine shift of the $\mathbb{F}_q$-linear span of $\{a_S\}_{S \in \mathcal{D} \setminus \{\mathbf{0}\}}$. Since this range is of size at least q^k, we must have $\dim\{a_S\}_{S \in \mathcal{D} \setminus \{\mathbf{0}\}} \geq k$.

A simple but crucial observation from [9] is that a polynomial with coefficients in $\mathbb{F}_{q^n}$ whose non-constant coefficients span $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$ can be 'projected' to a non-constant polynomial with coefficients in $\mathbb{F}_q$. We formalize this in the definition and lemma below.

Definition 2 (Full-span polynomial). *We say that a polynomial $G \in \mathbb{F}_{q^n}[\boldsymbol{Z}] = \mathbb{F}_{q^n}[Z_1, \ldots, Z_r]$ has* full span *if the coefficients of the non-constant monomials of G span $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$.*

Lemma 2 (Disperser for full-span polynomials). *Suppose $G \in \mathbb{F}_{q^n}[\boldsymbol{Z}]$ has full span. Let $T : \mathbb{F}_{q^n} \mapsto \mathbb{F}_q$ be a non-trivial $\mathbb{F}_q$-linear mapping. Then $T(G(\boldsymbol{Z}))$, as a function from $\mathbb{F}_q^r$ to $\mathbb{F}_q$, is a non-constant polynomial in $\mathbb{F}_q[\boldsymbol{Z}]$ whose total and individual degrees are at most those of G.*

See the full version of a proof of the lemma.

The previous lemma implies that to construct a disperser for polynomial sources it suffices to produce a function that increases the span of low-degree polynomials, which is what we do in the next theorem which is of paramount importance in this paper.

Theorem 4 (Product of Frobenius automorphisms increases span). *Fix a prime power $q = p^\ell$. Fix integers $k \leq n$ and $d < s$ such that n is prime and s is a power of p. (In particular, raising to power s^i is a Frobenius automorphism of $\mathbb{F}_q$ over $\mathbb{F}_p$.) Let $u = \lceil (n-k)/(k-1) \rceil$. Then for any $f(Z_1, \ldots, Z_r) \in \mathcal{M}[n, k, d]$, the polynomial*

$$f^{1+s+s^2+\ldots+s^u}(Z_1, \ldots, Z_r) = f(Z_1, \ldots, Z_r) \cdot f^s(Z_1, \ldots, Z_r) \cdots f^{s^u}(Z_1, \ldots, Z_r)$$

has full span.

Proof. Fix $f \in \mathcal{M}[n,k,d]$. As in (4), write $f(\mathbf{Z}) = \sum_{S\in\mathcal{D}} a_S \cdot \mathbf{Z}^S$ with $a_S \in \mathbb{F}_{q^n}$.

$$f^{1+s+s^2+\ldots+s^u}(\mathbf{Z}) = \left(\sum_{S\in\mathcal{D}} a_S \cdot \mathbf{Z}^S\right)^{1+s+s^2+\ldots+s^u} = \prod_{i=0}^{u}\left(\sum_{S\in\mathcal{D}} a_S \cdot \mathbf{Z}^S\right)^{s^i}$$

In what follows we use the notation $S_i = (S_{i,1},\ldots,S_{i,r})$ and $S_i \cdot s^i = (S_{i,1}\cdot s^i,\ldots,S_{i,r}\cdot s^i)$. Using the linearity of Frobenius automorphisms we continue the derivation and get

$$= \prod_{i=0}^{u}\left(\sum_{S\in\mathcal{D}} a_S^{s^i} \cdot \mathbf{Z}^{S\cdot s^i}\right) = \sum_{S_0,\ldots,S_u\in\mathcal{D}} \prod_{i=0}^{u} a_{S_i}^{s^i} \cdot \prod_{i=0}^{u} \mathbf{Z}^{S_i\cdot s^i}$$

$$= \sum_{S_0,\ldots,S_u\in\mathcal{D}} \prod_{i=0}^{u} a_{S_i}^{s^i} \cdot \prod_{i=0}^{u}\prod_{j=1}^{r} Z_j^{S_{i,j}\cdot s^i} = \sum_{S_0,\ldots,S_u\in\mathcal{D}} A_{S_0,\ldots,S_u} \cdot M_{S_0,\ldots,S_u}(\mathbf{Z}),$$

where $A_{S_0,\ldots,S_u} = \prod_{i=0}^{u} a_{S_i}^{s^i}$ and $M_{S_0,\ldots,S_u}(\mathbf{Z}) = \prod_{i=0}^{u}\prod_{j=1}^{r} Z_j^{S_{i,j}\cdot s^i}$. The crucial observation is that if $(S_0,\ldots,S_u)$ and $(S'_0,\ldots,S'_u)$ are two distinct tupples of elements of $\mathcal{D}$ then the monomials $M_{S_0,\ldots,S_u}(\mathbf{Z})$ and $M_{S'_0,\ldots,S'_u}(\mathbf{Z})$ are distinct as well: Consider $j \in \{1,\ldots,r\}$ such that $S_{i,j} \neq S'_{i,j}$ for some $0 \le i \le u$. Then Z_j is raised to power $\sum_{i=0}^{u} S_{i,j}\cdot s^i$ in $M_{S_0,\ldots,S_u}(\mathbf{Z})$ and to power $\sum_{i=0}^{u} S'_{i,j}\cdot s^i$ in $M_{S'_0,\ldots,S'_u}(\mathbf{Z})$. These powers are different as for all $0 \le i \le u$, $S_{i,j}, S'_{i,j} \le d < s$; And there is only one way to write an integer in base s with 'coefficients' smaller than s.

Define $A \triangleq \{A_{S_0,\ldots,S_u} \mid S_0,\ldots,S_u \in \mathcal{D}\setminus\{\mathbf{0}\}\}$. For $0 \le i \le u$, define $B^{s^i} \triangleq \{a_S^{s^i} \mid S \in \mathcal{D}\setminus\{\mathbf{0}\}\}$. Note that $A = B^{s^0}\cdots B^{s^u}$. For all $0 \le i \le u$, by Lemma 1 and Claim 3.2 we have $\dim(B^{s^i}) \ge k$. Therefore, by Theorem 3 we get

$$\dim(A) \ge \min\{n, k\cdot(u+1) - u\} = n.$$

Our theorem follows by noticing that the coefficients of the non-constant monomials in $f^{1+s+s^2+\ldots+s^u}$ contain the set A, hence $f^{1+s+\ldots+s^u}$ has full span.

In the full version Theorem 4 is used together with a version of Weil's Theorem to obtain our main results.

Acknowledgements. We thank Swastik Kopparty for the proof of a version of Weil's Theorem appearing in the full version. We thank Swastik Kopparty and Shubhangi Saraf for helpful discussions. We thank Zeev Dvir for reading a previous version of this paper. The first author thanks Emanuele Viola for raising the question addressed in this paper. We thank the anonymous reviewers for helpful comments.

References

1. Ben-Sasson, E., Hoory, S., Rozenman, E., Vadhan, S., Wigderson, A.: Extractors for affine sources (2001) (unpublished Manuscript)
2. Ben-Sasson, E., Kopparty, S.: Affine dispersers from subspace polynomials. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing, pp. 65–74 (2009)
3. Ben-Sasson, E., Zewi, N.: From affine to two-source extractors via approximate duality. In: Fortnow, L., Vadhan, S.P. (eds.) STOC, pp. 177–186. ACM (2011)
4. Blum, N.: A boolean function requiring 3n network size. Theor. Comput. Sci. 28, 337–345 (1984)
5. Bourgain, J.: On the construction of affine extractors. Geometric & Functional Analysis 17(1), 33–57 (2007)
6. Chor, B., Goldreich, O.: Unbiased bits from sources of weak randomness and probabilistic communication complexity. SIAM Journal on Computing 17(2), 230–261 (1988); Special issue on cryptography
7. De, A., Watson, T.: Extractors and Lower Bounds for Locally Samplable Sources. In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) APPROX/RANDOM 2011. LNCS, vol. 6845, pp. 483–494. Springer, Heidelberg (2011)
8. Demenkov, E., Kulikov, A.S.: An Elementary Proof of a $3n - o(n)$ Lower Bound on the Circuit Complexity of Affine Dispersers. In: Murlak, F., Sankowski, P. (eds.) MFCS 2011. LNCS, vol. 6907, pp. 256–265. Springer, Heidelberg (2011)
9. DeVos, M., Gabizon, A.: Simple affine extractors using dimension expansion. In: Proceedings of the 25th Annual IEEE Conference on Computational Complexity, p. 63 (2010)
10. Dvir, Z.: Extractors for varieties (2009)
11. Dvir, Z., Gabizon, A., Wigderson, A.: Extractors and rank extractors for polynomial sources. Computational Complexity 18(1), 1–58 (2009)
12. Dvir, Z., Lovett, S.: Subspace evasive sets. Electronic Colloquium on Computational Complexity (ECCC) 18, 139 (2011)
13. Gabizon, A., Raz, R.: Deterministic extractors for affine sources over large fields. Combinatorica 28(4), 415–440 (2008)
14. Guruswami, V.: Linear-algebraic list decoding of folded reed-solomon codes. In: IEEE Conference on Computational Complexity, pp. 77–85. IEEE Computer Society (2011)
15. Hou, X., Leung, K.H., Xiang, Q.: A generalization of an addition theorem of kneser. Journal of Number Theory 97, 1–9 (2002)
16. Li, X.: A new approach to affine extractors and dispersers (2011)
17. Lidl, R., Niederreiter, H.: Introduction to finite fields and their applications. Cambridge University Press, Cambridge (1994)
18. Shaltiel, R.: Dispersers for affine sources with sub-polynomial entropy. In: Ostrovsky, R. (ed.) FOCS, pp. 247–256. IEEE (2011)
19. Viola, E.: Extractors for circuit sources. Electronic Colloquium on Computational Complexity (ECCC) 18, 56 (2011)
20. von Neumann, J.: Various techniques used in connection with random digits. Applied Math Series 12, 36–38 (1951)
21. Weil, A.: On some exponential sums. Proc. Nat. Acad. Sci. USA 34, 204–207 (1948)
22. Yehudayoff, A.: Affine extractors over prime fields (2009) (manuscript)

Multiple-Choice Balanced Allocation in (Almost) Parallel*

Petra Berenbrink[1], Artur Czumaj[2], Matthias Englert[2], Tom Friedetzky[3], and Lars Nagel[4]

[1] School of Computing Science, Simon Fraser University, Burnaby, B.C., Canada
petra@sfu.ca
[2] DIMAP and Department of Computer Science, University of Warwick, UK
{A.Czumaj,M.Englert}@warwick.ac.uk
[3] School of Engineering and Computing Sciences, Durham University, Durham, UK
tom.friedetzky@dur.ac.uk
[4] Zentrum für Datenverarbeitung, Johannes Gutenberg Universität Mainz, Germany
nagell@uni-mainz.de

Abstract. We consider the problem of resource allocation in a parallel environment where new incoming resources are arriving online in groups or *batches*.

We study this scenario in an abstract framework of allocating balls into bins. We revisit the allocation algorithm GREEDY[2] due to Azar, Broder, Karlin, and Upfal (*SIAM J. Comput.* 1999), in which, for sequentially arriving balls, each ball chooses two bins at random, and gets placed into one of those two bins with minimum load. The maximum load of any bin after the last ball is allocated by GREEDY[2] is well understood, as is, indeed, the entire load distribution, for a wide range of settings. The main goal of our paper is to study balls and bins allocation processes in a parallel environment with the balls arriving in *batches*. In our model, m balls arrive in batches of size n each (with n being also equal to the number of bins), and the balls in each batch are to be distributed among the bins simultaneously. In this setting, we consider an algorithm that uses GREEDY[2] for all balls within a given batch, the answers to those balls' load queries are with respect to the bin loads at the end of the previous batch, and do not in any way depend on decisions made by other balls from the same batch.

Our main contribution is a tight analysis of the new process allocating balls in batches: we show that after the allocation of *any* number of batches, the gap between maximum and minimum load is $O(\log n)$ with high probability, and is therefore independent of the number of batches used.

1 Introduction

One of the central challenges in modern distributed systems is to cope with the problem of allocating their resources effectively in a balanced way. In this paper we consider a general scenario of resource allocation in the case when new incoming resources are arriving as a *stream of batches*, and the resources from each incoming batch are to be

* Research supported by the Centre for Discrete Mathematics and its Applications (DIMAP), and by EPSRC awards EP/D063191/1, EP/G069034/1, EP/F043333/1 and EP/F043333/1.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 411–422, 2012.
© Springer-Verlag Berlin Heidelberg 2012

allocated instantly. Although any analysis of the resource allocation protocols depends heavily on various properties of the underlying system, such as, for instance, the underlying network, service times and processing times, our focus is to study resource allocation schemes in an abstract framework of balls and bins, which is known to be able to provide important insights into more complex systems.

The framework of balls and bins is a powerful model with various applications in computer science, e.g., the analysis of hashing, the modeling of load balancing strategies, or the analysis of distributed processes. The typical aim is to find strategies that balance the balls evenly among the bins and produce small maximum loads. While traditionally mostly processes allocating balls into random bins (the single-choice scheme) have been studied (cf. [8,10]), more recently the main focus has been on the analysis of extensions of processes that let each ball choose multiple random bins instead of one, and then allocate itself to one of the chosen bins by considering their loads (see, e.g., [3,4,7,9,14,15]). As many of these papers have demonstrated, multiple-choice protocols maintain the simplicity of the original single-choice scheme, while at the same they have superior performance in many natural settings. For example, if one allocates sequentially n balls into n bins by choosing $d \geq 2$ random bins for each ball and then places the ball into the lesser loaded of the chosen bins (such a scheme will be denoted by GREEDY$[d]$), then no bin will have load greater than $\frac{\ln \ln n}{\ln d} + O(1)$ with high probability (w.h.p.[1]) [3], which compares favorably to the allocation of the balls performed i.u.r. (*independently and uniformly at random*), where the maximum load is $\Theta(\frac{\ln n}{\ln \ln n})$, w.h.p.

One major disadvantage of the multiple-choice strategies described above is that they unfold their full potential only in a sequential setting. For example, to prove the bounds for the standard multiple-choice schemes [3,4,15], it is assumed that the m balls are allocated online, one after another, and that the information about bin loads is immediately updated after allocation of each ball. These assumptions are unrealistic in various load balancing applications, e.g., where the balls model the jobs in some parallel or distributed setting and the choices of the balls must be performed independently and in parallel, or in scenarios where the balls cannot easily access the current load of the bins, for example, because of the delay in receiving this information. To cope with this, various multiple-choice strategies have been developed for parallel environments to deal with concurrent requests [1,2,11,13] and communication delays [6,7,12]. They base their decisions on the number of parallel requests, allow extra rounds of communication, and in some cases let balls re-choose.

We investigate how multi-choice schemes perform in a semi-parallel environment. In our model, a stream of m balls arrives in batches of size n each (with n being equal to the number of bins), and the loads of the bins are updated only between batches, that is, any decisions made by balls belonging to the same batch are strictly concurrent. In this setting, the algorithm uses GREEDY$[d]$, but for all balls within a given batch, the answers to those balls' load queries are with respect to the bin loads at the end of the previous batch, and do not in any way depend on decisions made by balls belonging to the same batch. We show that, for $d = 2$, after the allocation of the m^{th} ball, the gap

[1] Event $\mathcal{E}$ holds *with high probability* if $P(\mathcal{E}) \geq 1 - n^{-c}$ for any constant $c > 0$; it is important to notice that throughout the paper this bound is independent of m, the number of balls.

between the maximum and the minimum load is $O(\log n)$ with high probability, with probability at least $1 - 1/n^{O(1)}$, and is therefore independent of the number of batches.

1.1 Our Model

In this paper we investigate how the bare GREEDY$[2]$ protocol performs in a semi-parallel environment in which $m \geq n$ balls are allocated into n bins. Concurrent requests to the same bin are answered with the same current load (load here means the number of balls allocated to the bin) and no additional information, like the number of new requests. We model this by updating the bins only after every n^{th} ball and show that the gap between maximum and minimum load is independent of the number of balls (and batches). With high probability, the gap is $O(\log n)$, similar to the bounds in the sequential setting [4]. Our process follows GREEDY$[2]$ [3,4], but we introduce explicit *batches* of size n, and will assume that all *balls within one batch will be allocated concurrently*. Our protocol (which we shall refer to as BGREEDY$[2]$, short for *batch greedy*) is, therefore, in some sense a mix between sequential and parallel.

When a new batch of n balls arrives, each ball has two random choices and goes into a bin of lower load. If both loads are equal, the bin with smaller ID is selected. (The use of the bins' IDs is only to break the ties; there are no restrictions for the IDs other than that they are all distinct and there is a total order defined on them.) Note that due to the batch structure of our model, bin loads are updated only after having allocated all n balls belonging to a batch.

BGREEDY$[d]$**:**

- Repeat $\frac{m}{n}$ times:
  - ▷ for each of the n new balls in a new batch, *independently in parallel* do the following:
    - ◇ choose d bins i.u.r.
    - ◇ allocate the ball into the chosen bin with the minimum load[a]; in case of tie, allocate the ball into the chosen minimum-load bin with the smallest ID

[a] The load of each bin remains unchanged for the allocation of all balls from the same batch.

Our goal is to show that after throwing m balls into n bins, the gap between the maximum load and the minimum load is at most $O(\log n)$, with high probability, independent of the number of balls. We restrict our analysis to the case $d = 2$. Indeed, experiments with larger values of d suggest that the resulting load distribution does not improve but gets slightly worse, though still being $O(\log n)$ for constant d.

1.2 Related Work

There is a vast amount of literature studying the resource allocation problem modeled using the balls into bins framework. The classical processes allocating balls into random bins (the single-choice schemes) have been surveyed, e.g., in [8,10], and used in many

areas of mathematics, computer science, and engineering. The multiple-choice schemes have been used in these areas and in various settings, e.g., in adaptive load sharing [7], PRAM simulations [9], load balancing [3], and numerous follow-up papers, e.g., [1,4,5,14].

Although the multiple-choice schemes have been originally studied in the context of sequential allocation, there has also been a significant interest in its use in a parallel setting, see, e.g., [1,2,11,13]. Most known strategies involve additional rounds of communication, some are also adaptive and allow for re-choosing bins. In a typical parallel multiple-choice scheme, one aims at allocating n balls into n bins by allocating the balls with very limited coordination and using as few as possible extra communication rounds. For example, Lenzen and Wattenhofer [11] show that one can attain a maximum load of 2 using $\log^* n + O(1)$ rounds of communication, w.h.p.

The main difference between the parallel multiple-choice schemes and our model is that in our setting, the allocation of the balls from a single batch must be done instantly, without any coordination between the allocation of balls in the same batch.

Our model shares some similarities with the *bulletin board model with periodic updates*, as proposed by Mitzenmacher [12], to deal with systems with "outdated information." The model deals with the continuous process of allocating balls into bins: the balls are arriving as a Poisson stream of rate λn, $\lambda < 1$, and each bin "serves" (removes) its balls with exponential distribution with mean 1. The novel feature of the model is the access to the information about the load of the bins, which is available through a *bulletin board*, and which can contain outdated information about the load of the bins. The main variant of the model proposed by Mitzenmacher [12], *the bulletin board model with periodic updates*, assumes that the information about the load of each bin is updated periodically every T seconds, that is, for every $k \in \mathbb{N}$, to allocate the balls arriving in time interval $[kT, (k+1)T)$, the process will use the load of the bins at time kT. Mitzenmacher [12] considers three allocation mechanisms in this setting: (i) each ball chooses a bin i.u.r., (ii) each ball chooses a bin with the smallest load in the bulletin board, and (iii) each ball chooses d bins i.u.r. and is then allocated to the chosen bin with the smallest load in the bulletin board. Mitzenmacher [12] provided an analytical study for this model for the limiting case as $n \to \infty$ and supported the analytical results by simulations. The third model studied by Mitzenmacher [12] is very related to the model considered in our paper, though with several key differences. Firstly, it assumes stochastic arrivals of the balls and stochastic ball removals. Secondly, the paper only provides an analytical study in the limiting case which is supported by simulations, whereas our paper gives a rigorous probabilistic analysis.

1.3 Contributions of This Paper

We analyze BGREEDY$[2]$ in which the balls are allocated in batches of size n. We consider the scenario in which m balls are allocated into n bins, and we assume that the bins are initially empty. The allocation at *time* t is described by the load vector directly after the t^{th} batch. Our main goal is to understand the load of the bins after allocating m balls in $\frac{m}{n}$ batches for arbitrary values of m.

The main result of the paper, Theorem 3, is that after the last batch has been allocated, the load of any bin is $\frac{m}{n} \pm O(\log n)$ w.h.p. (with probability at least $1 - n^{-c}$ for any constant c). This follows from our two main technical results, Theorems 1 and 2.

We begin with Theorem 1 which studies the process under the assumption that the number of allocated balls is (relatively) small, at most polynomially large in n.

Theorem 1. *Let $\delta \geq 1$ be an arbitrary constant. Suppose that we run* BGREEDY$[2]$ *for $\tau \leq n^{\delta-1}$ batches, allocating $m \leq n^{\delta}$ many balls.*

1. For all $i \geq 0$ simultaneously, the number of bins with load at least $\frac{m}{n} + i + \gamma$ is upper bounded by ne^{-i}, w.h.p., where $\gamma = \gamma(\delta)$ denotes a suitable constant.

2. No bin has fewer than $\frac{m}{n} - O(\log n)$ balls, w.h.p.

Theorem 1 directly implies Corollary 1.

Corollary 1. *For any constant $\delta \geq 1$, if $m \leq n^{\delta}$ then the maximum load is $\frac{m}{n} + O(\log n)$ w.h.p. and the minimum load is $\frac{m}{n} - O(\log n)$ w.h.p.*

Our proof of Theorem 1 crucially relies on the assumption that m is at most polynomial in n. To deal with arbitrarily large values of m we prove Theorem 2 which removes the restriction of having to have only polynomially many balls, and reduces the problem to the case $m = \text{poly}(n)$.

Theorem 2. *Let c be a sufficiently large constant. Suppose that we run* BGREEDY$[2]$ *for $\tau \geq n^{c}$ batches. Further suppose that the maximum load is at most* MAX *and that the minimum load is at least* MIN *with probability at least $\mathfrak{p}$. Then, for any positive constant δ and any $\tau^* > \tau$, the process after running τ^* batches will have maximum load at most* MAX *and minimum load at least* MIN *with probability at least $\mathfrak{p} - n^{-\delta}$.*

By combining Theorem 2 with Corollary 1 we immediately obtain the following main theorem, which holds for any number m of allocated balls.

Theorem 3 (Main). *Fix n and m to be arbitrary integers and let c be any constant. If one allocates m balls into n bins using* BGREEDY$[2]$ *then with probability at least $1 - n^{-c}$ the maximum load is $\frac{m}{n} + O(\log n)$ and the minimum load is $\frac{m}{n} - O(\log n)$.*

Remark 1. Let us emphasize that Theorem 3 ensures that the gap between the maximum and the minimum load is $O(\log n)$ w.h.p. *at the end of the process*. It is easy to see that for large enough m no such bound can be ensured after every single batch.

The Approach. On a high level, our analysis follows the approach proposed by Berenbrink *et al.* [4] (see also [14]), but there are differences when applying the line of attack from [4] to the parallel setting considered in this paper. Our analysis uses new ideas and needs to be significantly tighter in several places.

The first part of our analysis (Theorem 1 and Corollary 1, proven in Section 2) deals with the process after allocating a polynomial number of balls in the system, or equivalently, after a polynomial number of batches. That part forms the basic block of this paper, as the analysis of the general case can be reduced to it. Many ideas from [4] do not work any more once decisions have to be made based upon outdated information.

We split this (batch-wise) analysis into two sub-parts: The first provides bounds on the distribution of the underloaded bins (with load below the average), the second bounds on the distribution of the overloaded bins (with load above the average). Whereas the analysis of the underloaded bins follows the one in [4] rather closely, the analysis of the overloaded bins requires several new ideas. The basic approach used in [4], the layered induction, cannot (easily) be applied because of the large number of new balls allocated in parallel in each single round. Instead, using the fact that the probability for a bin to receive a ball does not change within a batch, we base our analysis on an appropriate bound on the expected number of new balls for each bin.

The second part of the analysis is related to the infinite process (the number of batches is arbitrarily large) and is formalized by the so-called Short Memory Theorem. It states that, informally, if we run the process for a long time, then the behavior of the load of the bins is essentially determined only by a small number of the most recent batches. With that, one can reduce the analysis for an arbitrary number of batches to the case in the first part, that is, to the case when the number of batches is only polynomially small. The proof of the Short Memory Theorem uses similar coupling arguments as the approach initiated in [4] (cf. also [14]), but the need to cope with parallel allocations for the same batch makes the arguments more involved.

Further Discussion. Our analysis shows that even in a parallel environment, where the tasks from the same batch are to be allocated concurrently, the idea of using multiple-choices for the allocation leads to a significant improvement in the performance of system. Indeed, if we used BGREEDY$[1]$ instead of BGREEDY$[2]$, that is, if all balls were allocated at random, then it is a folklore result that for $m \geq n \log n$ the gap between the maximum and the minimum load is $\Theta(\sqrt{m \log n/n})$, w.h.p. Thus, our result shows that despite the lack of any coordination between the allocation of balls in a single batch, the use of a multiple-choice allocation scheme can improve the performance of system as compared to the naive approach of fully random allocations (BGREEDY$[1]$).

Our result in Theorem 3 provides further evidence that even in systems with outdated information, by carefully choosing the allocation rules (multiple-choice allocation scheme), one can obtain a very balanced load allocation.

Let us also mention that our analysis is tight in the sense that for large enough m the gap between the maximum and the minimum load in BGREEDY$[2]$ is $\Omega(\log n)$, w.h.p.

2 Polynomially Many Balls (Theorem 1)

Our analysis for the case of polynomially many balls follows the outline of the proof of [4]. We will show two invariants, one for the underloaded and one for the overloaded bins. The underloaded bins are analyzed in Section 2.2, the overloaded bins in Section 2.3. Together the invariants shown in both sections imply Theorem 1.

2.1 Preliminaries

The *load* of a bin is the number of balls it contains. Assuming that balls are allocated sequentially, a ball's *height*, or *level*, is the load of the selected bin right after the allocation. Thus, one can picture the bin as a stack of balls and every new ball is simply

pushed on top of the stack. If balls arrive at the same time, then we nevertheless assume that they are added to the stack one after the other (in an arbitrary order) so that each ball has a unique height.

Fix a time step t and let m be the number of balls allocated until time step t (that is, in t batches of size n each). The average number of balls per bin at time t is $\frac{m}{n} = t$.

We call bins with fewer than t balls *underloaded* and bins with more than t balls *overloaded*. We will frequently refer to *holes* in the distribution. For a given bin, the number of holes is defined to be the number of balls it is short of the average load at that point of time.

Key invariants. Our analysis relies on the following invariants that we will prove to hold w.h.p. (for $t \leq \text{poly}(n)$):

- $L(t)$: At time t, there are at most $0.7 \cdot n$ holes.
- $H(t)$: At time t, there are at most $0.47 \cdot n$ balls of height at least $t + 5$.

Observe that since the total number of holes equals the total number of balls with height above average, invariant $L(t)$ immediately implies that there are at most $0.7 \cdot n$ balls with height $t + 1$ or larger at time t.

We will use induction on t to prove the invariants $L(t)$ and $H(t)$: we will show that if $L(0), \ldots, L(t-1)$ and $H(0), \ldots, H(t-1)$ hold, then $L(t)$ and $H(t)$ are fulfilled w.h.p. (Observe that unlike in [4], we do not need $L(t)$ to prove $H(t)$.) We will analyze the underloaded and overloaded bins separately; the corresponding analyses communicate only through the two invariants above. We will finally use invariant $H(t)$ to derive Theorem 1. Throughout the analysis, we use the following notation:

Definition 1. *For i, $t \geq 0$, we let $\alpha_i^{(t)}$ denote the fraction of bins with load at most $t - i$ at time t, and $\beta_i^{(t)}$ denote the fraction of bins with load at least $t + i$ at the same time t.*

2.2 Analysis of Underloaded Bins

We begin with the analysis of the load in the underloaded bins, that is, in the bins with the load below the average load. Our goal is to prove that for any t, if the invariants $L(0), \ldots, L(t-1)$ and $H(0), \ldots, H(t-1)$ hold, then $L(t)$ is fulfilled, that is, there are at most $0.7n$ holes at time t w.h.p. Our analysis follows the analysis for the underloaded bins from [4]. The details are omitted here.

Let c_1 and c_2 be suitable constants with $c_1 \leq c_2$. The idea is to prove the following two invariants (implying $L(t)$) for time $t \in [0, \text{poly}(n)]$:

- $L_1(t)$: For $1 \leq i \leq c_1 \cdot \ln n$, we have $\alpha_i^{(t)} \leq 1.6 \cdot 0.3^i$.
- $L_2(t)$: For $i \geq c_2 \cdot \ln n$, we have $\alpha_i^{(t)} = 0$.

The proofs of $L_1(t)$ and $L_2(t)$ use an "outer" induction on t and an "inner" (layered) induction on i. Note that the second invariant establishes the bound on the minimum load of Theorem 1.

2.3 Analysis of Overloaded Bins

In this section, we analyze the load in overloaded bins and we will prove invariant $H(t)$: there are not more than $0.47 \cdot n$ balls with height at least $t+5$ w.h.p. The proof assumes that invariant $L(t-1)$ holds, and hence that at time $t-1$ there are at most $0.7 \cdot n$ balls above the average $t-1$. Unlike our analysis in Section 2.2, this section is new and the analysis requires many new ideas compared to [4].

We will analyze invariants $H_1(t)$ and $H_2(t)$ that imply both $H(t)$ and Theorem 1. To formulate the invariants $H_1(t)$ and $H_2(t)$, we first define two auxiliary functions h and f:

Definition 2. *For any $i \geq 0$, define $h(i) = 67 \cdot 0.34^i$.*

Let ℓ denote the smallest integer i such that $h(i) \leq n^{-0.9}$ and let $\sigma \geq 1$ denote a suitable constant (that will be specified later). For $i \geq 4$, we define:

$$f(i) = \begin{cases} h(i) & \text{for } 4 \leq i < \ell \ , \\ \max\{h(i), \frac{1}{3} \cdot n^{-0.9}\} & \text{for } i = \ell \ , \\ \sigma \cdot n^{-1} & \text{for } i = \ell + 1 \ . \end{cases}$$

We use Definition 2 to set up our main invariants, $H_1(t)$ and $H_2(t)$. (Let us recall that $\beta_i^{(t)}$ denotes the fraction of bins with load at least $t+i$ at time t; see Definition 1.)

- $H_1(t)$: For $5 \leq i \leq \ell$, we have $\beta_i^{(t)} \leq f(i)$,
- $H_2(t)$: $\sum_{i>\ell} \beta_i^{(t)} \leq \sigma \cdot n^{-1}$.

$H_1(t)$ tells us that the number of balls decrease exponentially with each level. On level ℓ the fraction of balls is upper-bounded by $n^{-0.9}$. The number of balls above level ℓ can be bounded by a constant σ. The proof of the following observation follows easily from the properties of the function f.

Observation 1. *$H_1(t)$ and $H_2(t)$ imply $H(t)$.*

Observation 2. *If $L(t)$, $H_1(t)$ and $H_2(t)$ hold w.h.p. for all t, then Theorem 1 holds.*

Proof. First we show that the number of bins with load at least $\frac{m}{n} + i + 5$ is upper bounded by $n \cdot e^{-i}$: using Definition 2 and basic properties of functions f and h, we can show that for $i \geq 5$, the fraction β_i of balls on level i is upper-bounded by $h(i)$. Thus, it suffices to show that $e^{-k} \geq h(k+4)$ for $k \geq 1$:

$$1.08^k \geq 0.9 \Rightarrow e^{-k} \cdot 0.34^{-k} \geq 67 \cdot 0.34^4 \Leftrightarrow e^{-k} \geq h(k+4) = 67 \cdot 0.34^{k+4} \ .$$

It remains to prove that this upper bound holds w.h.p. for all $t \leq \frac{n^\delta}{n} = n^{\delta-1}$. This follows directly from the statement that $L(t)$, $H_1(t)$ and $H_2(t)$ hold w.h.p. for all t. □

Further details are omitted here, but the invariants H_1 and H_2 are proven by induction on t. Our induction assumptions are $H_1(0), \ldots, H_1(t-1), H_2(t-1)$ and $L(t-1)$. These assumptions provide a distribution of the balls over the bins at time $t-1$. The induction step is proven by bounding the number of additional balls for each bin w.h.p. Counting the number of additional balls is somewhat simplified by the fact that the probability for a bin to receive a ball from batch t does not depend on how many balls of batch t have been allocated before. This is because the protocol defines that the allocation of a ball depends only on the loads of the bins (immediately) before batch t.

3 Reducing to Polynomially Many Batches (Theorem 2)

In this section we sketch the arguments used to prove Theorem 2, which shows that in order to analyze the maximum and/or minimum load after allocating m balls it is sufficient to consider the scenario when the number of balls is polynomial with respect to the number of bins, that is, $m = \text{poly}(n)$.

The proof of Theorem 2 follows the approach proposed by [4] (see also [14]). The main idea (stated formally in Theorem 4 in Section 3.3) is to prove that in BGREEDY[2], if we start the process with K balls already allocated in the bins, and we then allocate another $K \cdot \text{poly}(n)$ batches using BGREEDY[2], the obtained load distribution will be (in a stochastic sense) almost independent of the initial allocation of the K balls in the system. Therefore, without loss of generality, we could assume that the initial allocation started with the same number of balls in every bin, in which case the process would be identical to the one which ignored the initial K balls. This allows us to reduce the analysis of BGREEDY[2] with m balls to the analysis of BGREEDY[2] with $m' \ll m$ balls, and by applying this recursively, we can reduce the analysis of BGREEDY[2] to the case when m is not too big, namely $m = \text{poly}(n)$.

3.1 Basic Definitions and Notation

We use the standard notation $[M] = \{1, 2, \ldots, M\}$ for any natural number M.

Load vectors and normalized load vectors. We model the allocation of balls in the bins using *load vectors*. A load vector $\mathbf{x} = (x_1, \ldots, x_n)$ specifies that the *load* of the i^{th} bin is x_i. We will consider *normalized* load vectors; a load vector $\mathbf{x}$ is *normalized* if the entries in $\mathbf{x}$ are sorted in non-increasing order, that is, $x_i \geq x_{i+1}$ for every $1 \leq i < n$. In that case, x_i denotes the number of balls in the i^{th} fullest bin. We observe that since in our analysis the order among the bins is irrelevant (apart from tie breaking according to bin IDs, which themselves are essentially arbitrary), we can restrict the state space to normalized load vectors.

Let us mention an important feature of our analysis: while the normalized load vectors are n-vectors with integer values, BGREEDY[2] resolves the ties in the load of the two chosen bin by taking the one with the smallest ID, and so the outcome of BGREEDY[2] depends on more than just the vector. However, one can always see any normalized load vector as the one in which we order the bins of the same load according to their IDs, from the largest ID to the smallest one. In view of that, the process of selecting two bins to allocate a ball according to BGREEDY[2] for a normalized load vector $\mathbf{x} = (x_1, \ldots, x_n)$ is equivalent to one of choosing two indices i^t, j^t i.u.r. and then allocating the ball into the bin corresponding to $x_{\max\{i^t, j^t\}}$.

3.2 Allocation Process and Markov Chains

We will model the allocation process (one step of BGREEDY[2]) by a Markov chain: if $\mathbf{X}_t$ denotes the (normalized) load vector at time t (after inserting t batches) then the stochastic process $(\mathbf{X}_t)_{t \in \mathbb{N}}$ corresponds to a Markov chain $\mathbb{MC} = (\mathbf{X}_t)_{t \in \mathbb{N}}$ whose transition probabilities are defined by our allocation process. In particular, $\mathbf{X}_t$ is a random

variable obeying a probability distribution $\mathcal{L}(\mathbf{X}_t)$ defined by t steps of BGREEDY[2]. (Throughout the paper we use the standard notation to denote the probability distribution of a random variable U by $\mathcal{L}(U)$.)

Measuring similarity of distributions. We use a standard measure of discrepancy between two probability distributions ϑ and ν on a space Ω, the *variation distance*, defined as $\|\vartheta - \nu\| = \frac{1}{2}\sum_{\omega\in\Omega} |\vartheta(\omega) - \nu(\omega)| = \max_{A\subseteq\Omega}(\vartheta(A) - \nu(A))$.

3.3 Short Memory Theorem

Now we are ready to state our key result: Short Memory Theorem 4. Let us begin with some further useful terminology. For any $n, K \in \mathbb{N}$, let $\Psi_{n,K}$ be the set of all normalized load vectors $\mathbf{x} = (x_1, \ldots, x_n)$ with $\sum_{i=1}^{n} x_i = K$. That is, $\Psi_{n,K}$ is the set of all normalized load vectors that describe the system with K balls allocated to n bins.

Theorem 4 (Short Memory Theorem). *Let $K \in \mathbb{N}$ and let $\mathbf{x}$ and $\mathbf{y}$ be any two normalized load vectors in $\Psi_{n,K}$. For any t, let $\mathbf{X}_t$ ($\mathbf{Y}_t$) be the random variable describing the normalized load vector after allocating t further batches on top of $\mathbf{x}$ ($\mathbf{y}$, respectively) using* BGREEDY[2].

Then, for any $\varepsilon > 0$ there is some $\tau = O(K \cdot n + n^6 \cdot \log^2(Kn/\varepsilon))$, such that for every $T \geq \tau$, $\|\mathcal{L}(\mathbf{X}_T) - \mathcal{L}(\mathbf{Y}_T)\| \leq \varepsilon$.

One should read the claim in Theorem 4 so that if we start with any two arbitrary allocations of K balls into n bins, then after adding $T = K(n\log(1/\varepsilon))^{O(1)}$ batches to each of them, the normalized load vectors of these two systems are almost indistinguishable; they will be stochastically identical with probability at least $1-\varepsilon$, for an arbitrary small, positive ε.

Sketch of the Proof of Theorem 4. The proof of Theorem 4 uses the neighboring coupling approach initiated in [4]. We consider two normalized load vectors after allocating τ batches, $\mathbf{x}^\tau = (x_1^\tau, \ldots, x_n^\tau)$ and $\mathbf{y}^\tau = (y_1^\tau, \ldots, y_n^\tau)$ that differ by a single ball, that is, $\mathbf{x}^\tau = \mathbf{y}^\tau + \mathbf{e}_i - \mathbf{e}_j$ for $i \neq j$. (Here, for any $s \in [n]$, $\mathbf{e}_s$ will denote an n-vector consisting of a single element 1 in the coordinate s and of 0 in all other coordinates. With this notation, if $\mathbf{x}^\tau = \mathbf{y}^\tau + \mathbf{e}_i - \mathbf{e}_j$ for $i \neq j$ then $x_i = y_i + 1$, $x_j = y_j - 1$, and $x_s = y_s$ for all $s \in [n] \setminus \{i, j\}$.) For any two normalized load vectors $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ with $\mathbf{x} = \mathbf{y} + \mathbf{e}_i - \mathbf{e}_j$ for any i, j, define $\mathbf{\Delta}(\mathbf{x}, \mathbf{y}) = \max\{|x_i - x_j|, |y_i - y_j|\}$. Note that if $\mathbf{x} = \mathbf{y}$ then $i = j$ and $\mathbf{\Delta}(\mathbf{x}, \mathbf{y}) = 0$.

We will analyze a *coupling* for the Markov chains $(\mathbf{X}_t)$ and $(\mathbf{Y}_t)$ starting with $\mathbf{x}^0$ and $\mathbf{y}^0$ differing by a single ball, where all random choices performed by $\mathbf{x}^\tau$ are identical to those performed by $\mathbf{y}^\tau$. More formally, let us first consider state $\mathbf{x}^\tau$. For each ball in a given batch, we first choose two random numbers $i, j \in [n]$ i.u.r., then take the larger of them, say i, and then allocate the ball to the i^{th} bin in the vector $\mathbf{x}^\tau = (x_1^\tau, \ldots, x_n^\tau)$. Then, the same choice of i is used for the same ball for the vector $\mathbf{y}^\tau$. Observe that this construction uses the fact that in the normalized vector, the bins with the same load are sorted in the decreasing order of their IDs.

It is not difficult to see that *(i)* the coupling $(\mathbf{x}^\tau, \mathbf{y}^\tau) \mapsto (\mathbf{x}^{\tau+1}, \mathbf{y}^{\tau+1})$ is a proper coupling (i.e., transitions $\mathbf{x}^\tau \mapsto \mathbf{x}^{\tau+1}$ and $\mathbf{y}^\tau \mapsto \mathbf{y}^{\tau+1}$ are faithful copies of one step of BGREEDY[2]), *(ii)* if $\mathbf{x}^\tau$ and $\mathbf{y}^\tau$ differ by a single ball then either $\mathbf{x}^{\tau+1}$ and $\mathbf{y}^{\tau+1}$ differ by a single ball or $\mathbf{x}^{\tau+1} = \mathbf{y}^{\tau+1}$, and *(iii)* if $\mathbf{x}^\tau = \mathbf{y}^\tau$ then our coupling ensures that $\mathbf{x}^{\tau+1} = \mathbf{y}^{\tau+1}$. In view of these properties, our interest is in the analysis of the number of steps required until $\mathbf{x}^\tau = \mathbf{y}^\tau$. Our central result about the coupling is as follows.

Lemma 1. *Let t be any time step of the process with $\mathbf{\Delta}(\mathbf{x}^t, \mathbf{y}^t) > 0$. Then, either*

- *for some constant $c > 0$: $\mathbf{Pr}\left[\, \mathbf{\Delta}(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) = 0 \mid \mathbf{x}^t, \mathbf{y}^t \,\right] \geq \frac{c}{n^3}$, or*
- $\mathbf{E}\left[\mathbf{\Delta}(\mathbf{x}^{t+1}, \mathbf{y}^{t+1}) \mid \mathbf{x}^t, \mathbf{y}^t, \mathbf{x}^t \neq \mathbf{y}^t\right] \leq \mathbf{\Delta}(\mathbf{x}^t, \mathbf{y}^t) - \frac{1}{n}$.

By combining Lemma 1 with some basic analysis of random walks on a line, we can prove the following.

Lemma 2. *Let ε be any positive real. If $\mathbf{\Delta}(\mathbf{x}^0, \mathbf{y}^0) = \Delta$ then the coupling satisfies $\mathbf{Pr}\left[\, \mathbf{x}^\tau = \mathbf{y}^\tau \mid \mathbf{x}^0, \mathbf{y}^0 \,\right] \geq 1 - \varepsilon$ for some $\tau = O(\Delta \cdot n + n^6 \cdot \log^2(n/\varepsilon))$.*

As the final step, we can combine Lemma 2 with the neighboring coupling approach from [4] to conclude the proof of Theorem 4.

3.4 Using Short Memory Theorem 4 to Prove Theorem 2

We are now ready to prove our key result, Theorem 2. Our approach follows the approach used in [4, Section 4] (see the discussion in [4, Remark 2, p. 1376]), and below we will briefly present the main ideas of the reduction.

Suppose that we have m batches to be allocated into n bins. We first allocate a smaller number of batches, say $m' \ll m$ batches with $m' \cdot n$ balls. Then, suppose we can show that the maximum load in any bin is at most $m' + \vartheta$ and the minimum load in any bin is at least $m' - \vartheta$, with sufficiently high probability $1 - p$, and for an appropriate value ϑ (majorization by the process of allocating all balls in random gives $\vartheta = O(\sqrt{m' \cdot \log n/p})$, see, e.g., [4]). Since the difference between the maximum and minimum load is at most 2ϑ, the distance between the load vector after allocating $m' \cdot n$ balls and the load vector in which every bin has identical load m' is at most $2\vartheta n$. Therefore, if we apply the Short Memory Theorem 4, after allocating a further ϑn^c batches for an appropriate constant c, we will have a system with $m' \cdot n + \vartheta n^{c+1}$ balls for which the distributions of the bins loads in these two processes are almost indistinguishable (w.h.p.). Hence, instead of analyzing the original process, it is sufficient to analyze the process in which we first allocate m' balls to each bin, and then allocate a further ϑn^c batches using BGREEDY[2] – but this process can completely ignore the first m' batches, because they are allocated deterministically. Therefore, we have shown that in order to analyze the process for $m = m' + \vartheta n^c$ batches, it is sufficient to analyze the same process for a smaller number of batches, for $m^* = \vartheta n^c$. As it has been shown in detail in [4], by applying the reduction recursively with an appropriate choice of parameters, the arguments above can be easily formalized to prove Theorem 2.

References

1. Adler, M., Berenbrink, P., Schröder, K.: Analyzing an Infinite Parallel Job Allocation Process. In: Bilardi, G., Pietracaprina, A., Italiano, G.F., Pucci, G. (eds.) ESA 1998. LNCS, vol. 1461, pp. 417–428. Springer, Heidelberg (1998)
2. Adler, M., Chakrabarti, S., Mitzenmacher, M., Rasmussen, L.: Parallel randomized load balancing. In: Proceedings of the 27th Annual ACM Symposium on Theory of Computing (STOC), USA, pp. 238–247 (1995)
3. Azar, Y., Broder, A.Z., Karlin, A.R., Upfal, E.: Balanced allocations. SIAM Journal on Computing 29(1), 180–200 (1999)
4. Berenbrink, P., Czumaj, A., Steger, A., Vöcking, B.: Balanced allocations: The heavily loaded case. SIAM Journal on Computing 35(6), 1350–1385 (2006)
5. Czumaj, A., Stemann, V.: Randomized allocation processes. In: Proceedings of the 38th IEEE Symposium on Foundations of Computer Science (FOCS), pp. 194–203 (1997)
6. Dahlin, M.: Interpreting stale load information. IEEE Transactions on Parallel and Distributed Systems 11(10), 1033–1047 (2000)
7. Eager, D.L., Lazowska, E.D., Zahorjan, J.: Adaptive load sharing in homogeneous distributed systems. IEEE Transactions on Software Engineering 12, 662–675 (1986)
8. Johnson, N.L., Kotz, S.: Urn Models and Their Application: An Approach to Modern Discrete Probability Theory. John Wiley & Sons, New York (1977)
9. Karp, R.M., Luby, M., Meyer auf der Heide, F.: Efficient PRAM simulation on a distributed memory machine. In: Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC), pp. 318–326 (1992)
10. Kolchin, V.F., Sevast'yanov, B.A., Chistyakov, V.P.: Random Allocations. V. H. Winston and Sons, Washington, D.C. (1978)
11. Lenzen, C., Wattenhofer, R.: Tight bounds for parallel randomized load balancing. In: Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC), pp. 11–20 (2011)
12. Mitzenmacher, M.: How useful is old information? IEEE Transactions on Parallel and Distributed Systems 11(1), 6–20 (2000)
13. Stemann, V.: Parallel balanced allocations. In: Proceedings of the 8th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), pp. 261–269 (1996)
14. Talwar, K., Wieder, U.: Balanced allocations: The weighted case. In: Proceedings of the 39th Annual ACM Symposium on Theory of Computing (STOC), pp. 256–265 (2007)
15. Vöcking, B.: How asymmetry helps load balancing. Journal of the ACM 50(4), 568–589 (2003)

Optimal Hitting Sets for Combinatorial Shapes*

Aditya Bhaskara[1], Devendra Desai[2], and Srikanth Srinivasan[3]

[1] Department of Computer Science, Princeton University
bhaskara@cs.princeton.edu
[2] Department of Computer Science, Rutgers University
devdesai@cs.rutgers.edu
[3] DIMACS, Rutgers University
srikanth@dimacs.rutgers.edu

Abstract. We consider the problem of constructing explicit Hitting sets for Combinatorial Shapes, a class of statistical tests first studied by Gopalan, Meka, Reingold, and Zuckerman (STOC 2011). These generalize many well-studied classes of tests, including symmetric functions and combinatorial rectangles. Generalizing results of Linial, Luby, Saks, and Zuckerman (Combinatorica 1997) and Rabani and Shpilka (SICOMP 2010), we construct hitting sets for Combinatorial Shapes of size polynomial in the alphabet, dimension, and the inverse of the error parameter. This is optimal up to polynomial factors. The best previous hitting sets came from the Pseudorandom Generator construction of Gopalan et al., and in particular had size that was quasipolynomial in the inverse of the error parameter.

Our construction builds on natural variants of the constructions of Linial et al. and Rabani and Shpilka. In the process, we construct fractional perfect hash families and hitting sets for combinatorial rectangles with stronger guarantees. These might be of independent interest.

1 Introduction

Randomness is a tool of great importance in Computer Science and combinatorics. The probabilistic method is highly effective both in the design of simple and efficient algorithms and in demonstrating the existence of combinatorial objects with interesting properties. But the use of randomness also comes with some disadvantages. In the setting of algorithms, introducing randomness adds to the number of resource requirements of the algorithm, since truly random bits are hard to come by. For combinatorial constructions, 'explicit' versions of these objects often turn out to have more structure, which yields advantages beyond the mere fact of their existence (e.g., we know of explicit error-correcting codes that can be efficiently encoded and decoded, but we don't know if random codes can [5]). Thus, it makes sense to ask exactly how powerful probabilistic algorithms and arguments are. Can they be 'derandomized', i.e., replaced by determinstic algorithms/arguments of comparable efficiency?[1] There is a long line of research that has addressed this question in various forms [19,11,18,23,16].

* Many proofs are missing from this extended abstract. The full version is on the arxiv.
[1] A 'deterministic argument' for the existence of a combinatorial object is one that yields an efficient deterministic algorithm for its construction.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 423–434, 2012.
© Springer-Verlag Berlin Heidelberg 2012

An important line of research into this subject is the question of derandomizing randomized space-bounded algorithms. In 1979, Aleliunas et al. [1] demonstrated the power of these algorithms by showing that undirected s-t connectivity can be solved by randomized algorithms in just $O(\log n)$ space. In order to show that any randomized logspace computation could be derandomized within the same space requirements, researchers considered the problem of constructing an efficient *ε-Pseudorandom Generator* (ε-PRG) that would stretch a short random seed to a long pseudorandom string that would be indistinguishable (up to error ε) to any logspace algorithm.[2] In particular, an ε-PRG (for small constant $\varepsilon > 0$) with seedlength $O(\log n)$ would allow efficient deterministic simulations of logspace randomized algorithms since a deterministic algorithm could run over all possible random seeds.

A breakthrough work of Nisan [18] took a massive step towards this goal by giving an explicit ε-PRG for $\varepsilon = 1/\mathrm{poly}(n)$ that stretches $O(\log^2 n)$ truly random bits to an n-bit pseudorandom string for logspace computations. In the two decades since, however, Nisan's result has not been improved upon at this level of generality. However, many interesting subcases of this class of functions have been considered as avenues for progress [20,12,14,13,15].

The class of functions we consider are the very natural class of *Combinatorial Shapes*. A boolean function f is a combinatorial shape if it takes n inputs $x_1, \ldots, x_n \in [m]$ and computes a symmetric function of boolean inputs that depend on the membership of the inputs x_i in sets $A_i \subseteq [m]$ associated with f. (A function of boolean bits $y_1, \ldots, y_n$ is symmetric if its output depends only on their sum.) In particular, ANDs, ORs, Modular sums and Majorities of subsets of the input alphabet all belong to this class. Until recently, Nisan's result gave the best known seedlength for any explicit ε-PRG for this class, even when ε was a constant. In 2011, however, Gopalan et al. [9] gave an explicit ε-PRG for this class with seedlength $O(\log(mn) + \log^2(1/\varepsilon))$. This seedlength is optimal as a function of m and n but suboptimal as a function of ε, and for the very interesting case of $\varepsilon = 1/n^{O(1)}$, this result does not improve upon Nisan's work.

Is the setting of small error important? We think the answer is yes, for many reasons. The first deals with the class of combinatorial shapes: many tests from this class accept a random input only with inverse polynomial probability (e.g., the alphabet is $\{0, 1\}$ and the test accepts iff the Hamming weight of its n input bits is $n/2$); for such tests, the guarantee that a $1/n^{o(1)}$-PRG gives us is unsatisfactory. Secondly, while designing PRGs for some class of statistical tests with (say) constant error, it often is the case that one needs PRGs with much smaller error — e.g., one natural way of constructing almost-$\log n$ wise independent spaces uses PRGs that fool parity tests [17] to within inverse polynomial error. Thirdly, the reason to improve the dependence on the error is simply because we know that such PRGs exist. Indeed, a randomly chosen function that expands $O(\log n)$ bits to an n-bit string is, w.h.p., an ε-PRG for $\varepsilon = 1/\mathrm{poly}(n)$. Derandomizing this existence proof is yet another challenge in understanding

[2] As a function of its random bits, the logspace algorithm is *read-once*: it scans its input once from left to right.

how to eliminate randomness from existence proofs, and the tools we gain in solving this problem might help us in solving others of a similar flavor.

Our result: While we are unable to obtain optimal PRGs for the class of combinatorial shapes, we make progress on a standard relaxation of this problem: the construction of an *ε-Hitting Set* (ε-HS). An ε-HS for the class of combinatorial shapes has the property that any combinatorial shape that accepts at least an ε fraction of truly random strings accepts at least one of the strings in the hitting set. This is clearly a weaker guarantee than what an ε-PRG gives us. Nevertheless, in many cases, this problem turns out to be very interesting and non-trivial: in particular, an ε-HS for the class of space-bounded computations would solve the long-standing open question of whether RL = L. Our main result is an explicit ε-HS of size $\text{poly}(mn/\varepsilon)$ for the class of combinatorial shapes, which is *optimal*, to within polynomial factors, for all errors.

Theorem 1 (Main Result (informal)). *For any $m, n \in \mathbb{N}, \varepsilon > 0$, there is an explicit ε-HS for the class of combinatorial shapes of size* $\text{poly}(mn/\varepsilon)$.

Related work: There has been a substantial amount of research into both PRGs and hitting sets for many interesting subclasses of the class of combinatorial shapes, and also some generalizations. Naor and Naor [17] constructed PRGs for parity tests of bits (alphabet size 2); these results were extended by Lovett et al. [13] and Meka and Zuckerman [15] to modular sums (with coefficients). Combinatorial rectangles, a subclass of combinatorial shapes, have also been the subject of much attention. A series of works [6,4,14] constructed ε-PRGs for this class of functions: the best such PRG, due to Lu [14], has seedlength $O(\log n + \log^{3/2}(1/\varepsilon))$. Linial et al. [12] constructed optimal hitting sets for this class of tests. We build on many ideas from this work.

We also mention two more recent results that are very pertinent to our work. The first is to do with Linear Threshold functions which are weighted generalizations of threshold symmetric functions of input bits. For this class, Rabani and Shpilka [21] construct an explicit ε-HS of optimal size $\text{poly}(n/\varepsilon)$. They use a bucketing and expander walk construction to build their hitting set. Our construction uses similar ideas.

The final result that we use is the PRG for combinatorial shapes by Gopalan et al. [9] that was mentioned in the introduction. This work directly motivates our results and moreover, we use their PRG as a black-box within our construction.

2 Notation and Preliminaries

Definition 1 (Combinatorial Shapes, Rectangles, Thresholds). *A function f is an* (m, n)-Combinatorial Shape *if there exist sets $A_1, \ldots, A_n \subseteq [m]$ and a symmetric function $h : \{0,1\}^n \to \{0,1\}$ such that $f(x_1, \ldots, x_n) = h(1_{A_1}(x_1), \ldots, 1_{A_n}(x_n))$*[3]*. If h is the AND function, we call f an* (m, n)-Combinatorial Rectangle. *If h is an unweighted threshold function (i.e. h accepts based on the sign of*

[3] 1_A is the indicator function of the set A.

$\sum_i 1_{A_i}(x_i) - \theta$ *for some* $\theta \in \mathbb{N}$*), then* f *is said to be an* (m, n)-Combinatorial Threshold. *We denote by* $\mathsf{CShape}(m, n)$, $\mathsf{CRect}(m, n)$, *and* $\mathsf{CThr}(m, n)$ *the class of* (m, n)*-Combinatorial Shapes, Rectangles, and Thresholds respectively.*

Notation. For $i \in [n]$, let $X_i = 1_{A_i}(x_i)$, $p_i = |A_i|/m$, $q_i = 1 - p_i$ and $w_i = p_i q_i$. Define the weight of a shape f as $w(f) = \sum_i w_i$. For $\theta \in \mathbb{N}$, let T^-_θ (resp. T^+_θ) be the function that accepts iff $\sum \mathbf{1}_{A_i}(X_i)$ is at most (resp. at least) θ.

Definition 2 (Pseudorandom Generators and Hitting Sets). *Let* $\mathcal{F} \subseteq \{0,1\}^A$ *denote a boolean function family for some input domain* A*. A function* $G : \{0,1\}^s \to A$ *is an* ε*-pseudorandom generator (*ε*-PRG) with seedlength* s *for a class of functions* $\mathcal{F}$ *if for all* $f \in \mathcal{F}$, $|\mathbb{P}_{x \in_u \{0,1\}^s}[f(G(x)) = 1] - \mathbb{P}_{y \in_u A}[f(y) = 1]| \leq \varepsilon$.*. An* ε*-hitting set (*ε*-HS) for* $\mathcal{F}$ *is a subset* $H \subseteq A$ *s.t. for any* $f \in \mathcal{F}$*, if* $\mathbb{P}_{x \in_u A}[f(x) = 1] \geq \varepsilon$*, then* $\exists x \in H$ *s.t.* $f(x) = 1$.

Remark 1. Whenever we say that there exist *explicit* families of combinatorial objects of some kind, we mean that the object can be constructed by a deterministic algorithm in time polynomial in the description of the object.

We will need the following previous results in our constructions.

Theorem 2 (ε**-PRGs for** $\mathsf{CShape}(m, n)$ **[9]).** *For every* $\varepsilon > 0$*, there exists an explicit* ε*-PRG for* $\mathsf{CShape}(m, n)$ *with seed-length* $O(\log(mn) + \log^2(1/\varepsilon))$.

Theorem 3 (ε**-HS for** $\mathsf{CRect}(m, n)$ **[12]).** *For every* $\varepsilon > 0$*, there exists an explicit* ε*-hitting set* $\mathcal{S}^{m,n,\varepsilon}_{LLSZ}$ *for* $\mathsf{CRect}(m, n)$ *of size* $\mathrm{poly}(m(\log n)/\varepsilon)$.

Recall that a distribution μ over $[m]^n$ is k-wise independent for $k \in \mathbb{N}$ if for any $S \subseteq [n]$ s.t. $|S| \leq k$, the marginal $\mu|_S$ is uniform over $[m]^{|S|}$. Also, $\mathcal{G} : \{0,1\}^s \to [m]^n$ is a *k-wise independent probability space over* $[m]^n$ if for uniformly randomly chosen $z \in \{0,1\}^s$, the distribution of $\mathcal{G}(z)$ is k-wise independent.

Fact 4 (Explicit k**-wise independent spaces)** *For any* $k, m, n \in \mathbb{N}$*, there is an explicit* k*-wise independent probability space* $\mathcal{G}^{m,n}_{k\text{-}wise} : \{0,1\}^s \to [m]^n$ *with* $s = O(k \log(mn))$.

Expanders. Recall that a degree-D graph $G = (V, E)$ on N vertices is an (N, D, λ)-expander if the second largest (in absolute value) eigenvalue of its normalized adjacency matrix is at most λ. We will use explicit expanders as a basic building block. We refer the reader to the excellent survey of Hoory, Linial, and Wigderson [10] for various related results.

Fact 5 (Explicit Expanders [10]) *Given any* $\lambda > 0$ *and* $N \in \mathbb{N}$*, there is an explicit* (N, D, λ)*-expander where* $D = (1/\lambda)^{O(1)}$.

Expanders have found numerous applications in derandomization. A central theme in these applications is to analyze random walks on a sequence of expander graphs. Let $G_1, \ldots, G_\ell$ be a sequence of (possibly different) graphs on

the *same* vertex set V. Assume G_i ($i \in [\ell]$) is an (N, D_i, λ_i)-expander. Fix any $u \in V$ and $y_1, \ldots, y_\ell \in \mathbb{N}$ s.t. $y_i \in [D_i]$ for each $i \in [\ell]$. Note that $(u, y_1, \ldots, y_\ell)$ naturally defines a 'walk' $(v_1, \ldots, v_\ell) \in V^\ell$ as follows: v_1 is the y_1th neighbour of u in G_1 and for each $i > 1$, v_i is the y_ith neighbour of v_{i-1} in G_i. We denote by $\mathcal{W}(G_1, \ldots, G_\ell)$ the set of all tuples $(u, y_1, \ldots, y_\ell)$ as defined above. Moreover, given $w = (u, y_1, \ldots, y_\ell) \in \mathcal{W}(G_1, \ldots, G_\ell)$, we define $v_i(w)$ to be the vertex v_i defined above (we will simply use v_i if the walk w is clear from the context).

We need a variant of a result due to Alon et al. [2] and a corollary that follows from it. The lemma as it is stated below is slightly more general than the one given in [2] but it can be obtained by using essentially the same proof and setting the parameters appropriately.

Lemma 1. *There is an absolute constant $c_{walk} > 0$ s.t. the following holds. Let $G_1, \ldots, G_\ell$ be a sequence of graphs defined on the same vertex set V of size N. Assume that G_i is an (N, D_i, λ_i)-expander. Let $V_1, \ldots, V_\ell \subseteq V$ s.t. $|V_i| \geq p_i N > 0$ for each $i \in [\ell]$. Then, $\mathbb{P}_{w \in \mathcal{W}(G_1, \ldots, G_\ell)}[\forall i \in [\ell], v_i(w) \in V_i] \geq \frac{1}{2^{c_{walk}\ell}} \prod_{i \in [\ell]} p_i$ as long as for each $i \in [\ell]$, $\lambda_i \leq (p_i p_{i-1})/10$.*

Corollary 1. *Let V be a set of N elements, and let $0 < p_i < 1$ for $1 \leq i \leq s$ be given. There exists an explicit set of walks $\mathcal{W}$, each of length s, s.t. for any subsets $V_1, V_2, \ldots, V_s$ of V, with $|V_i| \geq p_i N$, there exists a walk $w = w_1 w_2 \ldots w_s \in \mathcal{W}$ such that $w_i \in V_i$ for all i. Furthermore, there exist such $\mathcal{W}$ satisfying $|\mathcal{W}| \leq poly(N, \prod_{i=1}^{s} \frac{1}{p_i})$.*

Hashing. Hashing plays a vital role in all our constructions. Thus, we need explicit hash families which have several "good" properties. These are obtained by extending constructions due to Rabani and Shpilka [21], Schmidt and Siegel [22], and Fredman, Komlos, and Szemeredi [8]. The second lemma is a *fractional* version of the first. Proofs are omitted for lack of space.

Lemma 2 (Perfect Hash Families). *There is an absolute constant $c_{perf} > 0$ so that the following holds. For any $n, t \in \mathbb{N}$, there is an explicit family of hash functions $\mathcal{H}^{n,t}_{perf} \subseteq [t]^{[n]}$ of size $2^{O(t)}\text{poly}(n)$ s.t. for any $S \subseteq [n]$ with $|S| = t$, we have $\mathbb{P}_{h \in \mathcal{H}^{n,t}_{perf}}[h \text{ is 1-1 on } S] \geq \frac{1}{2^{c_{perf} t}}$.*

Lemma 3 (Fractional Perfect Hash families). *For an absolute constant c_{frac} and any $n, t \in \mathbb{N}$, there is an explicit family of hash functions $\mathcal{H}^{n,t}_{frac} \subseteq [t]^{[n]}$ of size $2^{O(t)} n^{O(1)}$ s.t. for any $z \in [0,1]^n$ with $\sum_{j \in [n]} z_j \geq 10t$, we have $\mathbb{P}_{h \in \mathcal{H}^{n,t}_{frac}}\left[\forall i \in [t], \sum_{j \in h^{-1}(i)} z_j \in [0.01M, 10M]\right] \geq \frac{1}{2^{c_{frac} t}}$, where $M = \frac{\sum_{j \in [n]} z_j}{t}$.*

3 Outline of the Construction

We first make a standard simplifying observation that we can throughout assume that $m, n, 1/\varepsilon$ can be $n^{O(1)}$. Thus, we only need to construct hitting sets of size $n^{O(1)}$ in this case. The proof is omitted. From now on, we assume $m, 1/\varepsilon = n^{O(1)}$.

Lemma 4. *Assume that for every constant $c \geq 1$, and $m \leq n^c$, there is an explicit $1/n^c$-HS for $\mathsf{CShape}(m,n)$ of size $n^{O_c(1)}$. Then, for any $m, n, \in \mathbb{N}$ and $\varepsilon > 0$, there is an explicit ε-HS for $\mathsf{CShape}(m,n)$ of size $\mathrm{poly}(mn/\varepsilon)$.*

Next, we prove a crucial lemma which shows how to obtain hitting sets for $\mathsf{CShape}(m,n)$ starting with hitting sets for $\mathsf{CThr}(m,n)$. This reduction crucially uses the fact that CShape does only 'symmetric' tests – it fails to hold, for instance, for natural "weighted" generalizations of CShape.

Lemma 5. *Suppose that for every $\varepsilon > 0$, there exist ε-HS for $\mathsf{CThr}(m,n)$ of size $F(m,n,1/\varepsilon)$. Then there exists an ε-HS for $\mathsf{CShape}(m,n)$ of size $(n+1) \cdot F^2(m,n,n/\varepsilon)$.*

Proof. Suppose we can construct hitting sets for $\mathsf{CThr}(m,n)$ and parameter ε' of size $F(m,n,1/\varepsilon')$, for all $\varepsilon' > 0$. Now consider some $f \in \mathsf{CShape}(m,n)$, defined using sets A_i and symmetric function h. Since h is symmetric, it depends only on the *number* of 1's in its input. In particular, there is a $W \subseteq [n] \cup \{0\}$ s.t. for $a \in \{0,1\}^n$ we have $h(a) = 1$ iff $|a| \in W$. Now if $\mathbb{P}_x[f(x) = 1] \geq \varepsilon$, there must exist a $w \in W$ s.t. the probability that $\mathbb{P}_x[|\{i \in [n] \mid 1_{A_i}(x_i) = 1\}| = w] \geq \varepsilon/|W| \geq \varepsilon/n$. Thus if we consider functions in $\mathsf{CThr}(m,n)$ defined by the same A_i, and thresholds T_w^+ and T_w^- respectively, we have that *both* have accepting probability at least ε/n, and thus an ε/n-HS $\mathcal{S}$ for $\mathsf{CThr}(m,n)$ must have 'accepting' elements $y, z \in [m]^n$ for T_w^- and T_w^+ respectively.

The key idea is now the following. Suppose we started with the string y and moved to string z by flipping the coordinates one at a time – i.e., the sequence of strings would be: $(y_1\ y_2\ \ldots\ y_n)$, $(z_1\ y_2\ \ldots\ y_n)$, $(z_1\ z_2\ \ldots\ y_n), \ldots, (z_1\ z_2\ \ldots\ z_n)$.

In this sequence the number of "accepted" indices (i.e., i for which $1_{A_i}(x_i) = 1$) changes by at most one in each 'step'. To start with, since y was accepting for T_w^-, the number of accepting indices was at most w, and in the end, the number is at least w (since z is accepting for T_w^+), and hence one of the strings must have precisely w accepting indices, and this string would be accepting for f!

Thus we can consider every pair of strings in a hitting set for $\mathsf{CThr}(m,n)$ and error ε/n, and consider the $(n+1)$ "intermediate" strings as a hitting set for $\mathsf{CShape}(m,n)$ of error ε. It is easy to check that it has size $(n+1) \cdot F^2(m,n,n/\varepsilon)$.

Overview of the Constructions. In what follows, we focus on constructing hitting sets for $\mathsf{CThr}(m,n)$. We will describe the construction of two families of hitting sets: the first is for the "high weight" case – $w(f) := \sum_i w_i > C \log n$ for some large constant C, and the second for the case $w(f) < C \log n$. The final hitting set is a union of the ones for the two cases. The high-weight case is conceptually simpler, and illustrates the important tools. A main tool in both cases is a "fractional" version of the perfect hashing lemma, which, though a consequence of folklore techniques, does not seem to be known in this generality (Lemma 3).

The proof of the low-weight case is technically more involved, and for lack of space, we only present the solution in the special case when all the sets A_i are "small", i.e., we have $p_i \leq 1/2$ for all i. This case illustrates the main tool we use for the low-weight case, which is the perfect hashing lemma (which appears, for

instance in derandomization of "color coding" – a trick introduced in [3], which our proof in fact bears a resemblance to).

4 Hitting Sets for Combinatorial Thresholds

As described above, we first consider the high-weight case (i.e., $w(f) \geq C \log n$ for some large absolute constant C). Next, we consider the low-weight case, with an additional restriction that each of the accepting probabilities $p_i \leq 1/2$. This serves as a good starting point to explain the *general* low-weight case, which we get to in section 4.2. The theorem we finally prove in the section is as follows

Theorem 6. *Suppose $m, 1/\varepsilon = n^{O(1)}$. For the class of functions $\mathsf{CThr}(m, n)$, there exists an explicit ε-hitting set of polynomial size.*

4.1 High Weight Case

Theorem 7. *For any $c > 0$, there is a $C > 0$ s.t. for $m, 1/\varepsilon \leq n^c$, there is an explicit ε-HS of size $n^{O_c(1)}$ for the class of functions in $\mathsf{CThr}(m, n)$ of weight at least $C \log n$.*

As discussed earlier, we wish to construct hitting sets for T^+_θ and T^-_θ, for θ s.t. the probability of the event for independent, perfectly random x_i is at least $1/n^c$. For convenience, define $\mu := p_1 + p_2 + \cdots + p_n$, and $W := w_1 + w_2 + \ldots w_n$. We have $W > C \log n$ for a large constant C (it needs to be *large* compared to c, as seen below). First, we have the following by Chernoff bounds.

Claim. If $\mathbb{P}_x[T^+_\theta(x) = 1] > \varepsilon$ $(\geq 1/n^c)$, we have $\theta \leq \mu + 2\sqrt{cW \log n}$.

Outline. Let us concentrate on hitting sets for the event T^+_θ (the case T^- follows verbatim). The main idea is the following: we first divide the indices $[n]$ into $\log n$ buckets using a hash function (from a *fractional perfect hash family*, see Lemma 3). This is to ensure that the w_i get distributed uniformly. Second, we aim to obtain an *advantage* of roughly $2\sqrt{\frac{cW}{\log n}}$ in each of the buckets (advantage is w.r.t. the mean in each bucket). Third, we ensure that the advantages add up, giving a total advantage of $2\sqrt{cW \log n}$ over the mean, which is what we intended to obtain. In the second step (i.e., in one bucket), we can prove that the desired advantage occurs with *constant* probability, and thus we can use a result of Gopalan et al. [9]. Finally, in the third step, we cannot afford to use different hash functions in different buckets (this would result in a seed length of $\Theta(\log^2 n)$) – thus we need to use expander walks to save on randomness. Let us now describe the three steps in detail. We note that these steps parallel the results of Rabani and Shpilka [21].

The first step is straightforward: we pick a hash function from a perfect fractional hash family $\mathcal{H}^{n,\log n}_{\text{frac}}$. From Lemma 3, we obtain

Claim. For every set of weights w, there exists an $h \in \mathcal{H}^{n,\log n}_{\text{frac}}$ s.t. for all $1 \leq i \leq \log n$, we have $\frac{W}{100 \log n} \leq \sum_{j \in h^{-1}(i)} w_j \leq \frac{100W}{\log n}$.

The rest of the construction is done starting with each $h \in \mathcal{H}_{\text{frac}}^{n,\log n}$. Thus for analysis, suppose that we are working with an h satisfying the inequality from the above claim. For the second step, we first prove that for independent random $x_i \in [m]$, we have a constant probability of getting an *advantage* of $2\sqrt{\frac{cW}{\log n}}$.

Lemma 6. *Let S be the sum of k independent random variables X_i, with $\mathbb{P}[X_i = 1] = p_i$, let c' be a constant, and let $\sum_i p_i(1-p_i) \geq \sigma^2$, for some σ satisfying $\sigma \geq 4e^{c'^2}$. Define $\mu := \sum_i p_i$. Then $\mathbb{P}[S > \mu + c'\sigma] \geq \alpha$, and $\mathbb{P}[S < \mu - c'\sigma] \geq \alpha$, for some constant α depending on c'.*

The proof is straightforward, but it is instructive to note that in general, a random variable (in this case, S) need not deviate "much more" (in this case, a c' factor more) than its standard deviation: we have to use the fact that S is the sum of independent r.v.s. This is done by an application of the Berry-Esséen theorem [7]. We refer to the full version of the paper for details.

Now, note that since α is a *constant*, we can appeal to the result of Gopalan et al. [9] and use the output of a pseudorandom generator instead of independent x_i (in each bucket), and succeed w.p. at least $\alpha/2$.

Thus we are left with the third step: here for each bucket, we would like to have (independent) PRGs which generate the corresponding x_i (and each of these PRGs has a seed length of $O(\log n)$). Since we cannot afford $O(\log^2 n)$ total seed length, we instead do the following: consider a PRG for combinatorial thresholds defined on *all* the n indices (we can obtain assignments to a subset of indices by restriction). This is done for error parameter $\alpha/2$ (a constant), thus the seed length is only $O(\log n)$. Let $\mathcal{S}$ be such a PRG (viewed as a collection of strings: $\mathcal{S} \subseteq [m]^n$). From the above, we have that for the ith bucket, the probability $x \sim \mathcal{S}$ exceeds the threshold on indices in bucket i is at least $\alpha/2$. Now there are $\log n$ buckets, and in each bucket, the probability of 'success' is at least $\alpha/2$. We can thus appeal to the 'expander walk' lemma of Alon et al. [2] (see preliminaries, Lemma 1 and the corollary following it).

This means the following: we consider an explicitly constructed expander on a graph with vertices being the elements of $\mathcal{S}$, and the degree being a constant depending on α). We then perform a random walk of length $\log n$ (the number of buckets). Let $s_1, s_2, \ldots, s_{\log n}$ be the strings (from $\mathcal{S}$) we see in the walk. We form a new string in $[m]^n$ by picking values for indices in bucket i, from the string s_i. By the Expander walk lemma (1), with non-zero probability, this will succeed for *all* $1 \leq i \leq \log n$, and this gives the desired advantage.

The seed length for generating the walk is $O(\log |\mathcal{S}|) + O(1) \cdot \log n = O(\log n)$. Combining (or in some sense, *composing*) this with the hashing earlier completes the construction.

4.2 Thresholds with Small Weight (and Small Sized Sets)

We now prove Theorem 6 for the case of shapes f satisfying $w(f) = O(\log n)$.

Theorem 8. *Fix any $c \geq 1$. For any $m = n^c$, there exists an explicit $1/n^c$-HS $\mathcal{S}_{low}^{n,c} \subseteq [m]^n$ of size $n^{O_c(1)}$ for functions $f \in \mathsf{CThr}(m,n)$ s.t. $w(f) \leq c \log n$.*

We will prove this theorem in the rest of this sub-section in the *special case that the underlying subsets of f, $A_1, \ldots, A_n \subseteq [m]$ are small*: $p_i \leq 1/2$ for each i. To begin, we note that hitting sets for the symmetric function T_θ^- are very easy to come up with in this case. In particular, since T_0^- accepts iff $\sum X_i = 0$, it can also be interpreted as a combinatorial rectangle on $\overline{A_1}, \ldots, \overline{A_n}$. The probability of this event over uniformly chosen inputs is at least $\prod_i (1 - p_i) \geq e^{-2\sum_i p_i} \geq e^{-4\sum_i p_i(1-p_i)} \geq n^{-4c}$. Thus the existence of these follows from Linial et al.. [12]. Further, by definition, a hitting set for T_0^- is also a hitting set for T_θ^- for $\theta > 0$.

Let us now fix a T_θ^+ that accepts with good probability, $\mathbb{P}_x[T_\theta^+(x) = 1] \geq \varepsilon$. Since $w(T_\theta^+) \leq c \log n$, it follows that $\mu \leq 2c \log n$. Thus by a Chernoff bound and the fact that $\varepsilon = 1/n^c$, we have that $\theta \leq c' \log n$ for some $c' = O_c(1)$.

Outline. The idea is to use a *perfect hash family* (not a fractional one) mapping $[n] \mapsto [\theta]$. The aim will now be to obtain a contribution of 1 from each bucket[4]. In order to do this, we require $\prod_i \mu_i$ be large, where μ_i is the sum of p_j for j in bucket B_i. By a reason similar to color coding (see [3]), it will turn out that this quantity is large when we bucket using a perfect hash family. We then prove that using a pairwise independent space in each bucket B_i "nearly" gives probability μ_i of succeeding. As before, since we cannot use independent hashes in each bucket, we take a hash function over $[n]$, and do an expander walk. The final twist is that in the expander walk, we cannot use a constant degree expander: we will have to use a sequence of expanders on the same vertex set with appropriate degrees (some of which can be super-constant, but the product will be small). This will complete the proof. We note that the last trick was implicitly used in the work of [12].

Construction. Let us formally describe a hitting set for T_θ^+ for a fixed θ. (The final set $\mathcal{S}_{\text{low}}^{n,c}$ will be a union of these for $\theta \leq c' \log n$ along with the hitting set of [12]).

Step 1: Let $\mathcal{H}_{perf}^{n,\theta} = \{h : [n] \to [\theta]\}$ be a perfect hash family as in Lemma 2. The size of the hash family is $2^{O(\theta)}\text{poly}(n) = n^{O_{c'}(1)} = n^{O_c(1)}$. For each hash function $h \in \mathcal{H}_{perf}^{n,\theta}$ divide $[n]$ into θ buckets $B_1, \ldots, B_\theta$ (so $B_i = h^{-1}(i)$).

Step 2: We will plug in a pairwise independent space in each bucket. Let $\mathcal{G}_{2-wise}^{m,n} : \{0,1\}^s \to [m]^n$ denote the generator of a pairwise independent space. Note that the seed-length for any bucket is $s = O(\log n)$[5].

Step 3: The seed for the first bucket is chosen uniformly at random and seeds for the subsequent buckets are chosen by a walk on expanders with varying degrees. For each $i \in [\theta]$ we choose every possible η_i' such that $1/\eta_i'$ is a power of 2 and $\prod_i \eta_i' \geq 1/n^c$. There are at most $\text{poly}(n)$ such choices for all η_i''s in total. We then take an $(2^s, d_i, \lambda_i)$-expander H_i on vertices $\{0,1\}^s$ with degree $d_i = \text{poly}(1/(\eta_i'\eta_{i-1}'))$ and $\lambda_i = \eta_i'\eta_{i-1}'/100$. Now for any $j \in [\theta]$, $u \in \{0,1\}^s$, $\{y_i \in [d_i]\}_{i=1}^j$, let $(u, y_1, \ldots, y_j) \in \mathcal{W}(H_1, \ldots, H_j)$ be a j-step walk. For all

[4] This differs from the high-weight case, where we looked at advantage over the mean.

[5] We do not use generators with different output lengths, instead we take the n-bit output of one generator and restrict to the entries in each bucket.

starting seeds $z_0 \in \{0,1\}^s$ and all possible $y_i \in [d_i]$ for all $i \in [\theta]$, construct the input for the i-th bucket as $x|_{B_i} = G(v_i(z_0, y_1, \ldots, y_i))$.

Size. We have $|\mathcal{S}^{n,c}_{\text{low}}| = c' \log n \cdot n^{O_c(1)} \cdot \prod_i d_i$, where the $c' \log n$ factor is due to the choice of θ, the $n^{O_c(1)}$ factor is due to the size of the perfect hash family, the number of choices of $(\eta'_1, \ldots, \eta'_\theta)$, and the choice of the first seed, and the $\prod_i d_i$ factor is the number of expander walks. Simplifying, $|\mathcal{S}^{n,c}_{\text{low}}| = n^{O_c(1)} \prod d_i = n^{O_c(1)} \prod_i (\eta'_i)^{-O(1)} \leq n^{O_c(1)}$, where the last inequality is due to the choice of η'_i's.

Analysis. We follow the outline. First, we can upper bound $\mathbb{P}_{x \sim [m]^n}[T^+_\theta(x) = 1]$ by $\sum_{|S|=\theta} \prod_{i \in S} p_i \geq \varepsilon$ (this is like a Union bound). Second, if we hash the indices $[n]$ into θ buckets at random and consider one S with $|S| = \theta$, the probability that the indices in S are 'uniformly spread' (one into each bucket) is $1/2^{O(\theta)}$. This property is also true if we pick h from the perfect hash family. Formally, given an $h \in \mathcal{H}^{n,\theta}_{perf}$, define $\alpha_h = \prod_{i \in \theta} \sum_{j \in h^{-1}(i)} p_j$. Over a uniform choice of h from the family $\mathcal{H}^{n,\theta}_{perf}$, we can conclude that $\mathbb{E}_h \alpha_h \geq \sum_{|S|=\theta} \prod_{i \in S} p_i \, \mathbb{P}_h[h \text{ is 1-1 on } S] \geq \varepsilon / 2^{O(\theta)} \geq 1/n^{O_c(1)}$. Thus there must exist an h that satisfies $\alpha_h \geq 1/n^{O_c(1)}$. We fix such an h.

For a bucket B_i, define $\mu_i = \sum_{j \in B_i} p_j$ and $\eta_i = \mathbb{P}[\sum_{j \in B_i} X_j \geq 1]$ where the probability is taken over inputs generated by $\mathcal{G}^{m,n}_{2-wise}$. Further, call a bucket B_i as being *good* if $\mu_i \leq 1/10$, otherwise call the bucket *bad*. The following claim gives lower bounds on the probability that in any given bucket, at least one of the input coordinates x_j satisfies $x_j \in A_j$. The claim is easily proved using pairwise independence and inclusion-exclusion. We omit the proof.

Claim. For any *good* bucket B_i, $\eta_i \geq \mu_i/2$ and for any *bad* bucket B_i, $\eta_i \geq 1/20$. Moreover, the good buckets collectively satisfy $\prod_{B_i \text{ good}} \mu_i \geq 1/n^{O_c(1)}$.

For a moment, let us analyze the construction assuming independent pairwise independent spaces in each bucket. Then, using the above claim, the *success* probability, namely the probability that *every* bucket i has a non-zero $\sum_{j \in h^{-1}(i)} X_j$ is equal to $\prod_i \eta_i \geq (\prod_{B_i \text{ bad}} 1/20)(\prod_{B_i \text{ good}} \mu_i/2) = (1/2^{O(\theta)}) \prod_{B_i \text{good}} \mu_i \geq 1/(2^{O(\theta)} n^{O_c(1)}) \geq 1/n^{O_c(1)}$.(∗)

If now the seeds for $\mathcal{G}^{m,n}_{2-wise}$ in each bucket are chosen according to the expander walk corresponding to the probability vector $(\eta_1, \ldots, \eta_\theta)$, then by Lemma 1 the success probability becomes at least $(1/2^{O(\theta)}) \prod_i \eta_i \geq 1/n^{O_c(1)}$, using (∗) for the final inequality.

But we are not done yet. We cannot guess the correct probability vector exactly. Instead, we get a closest guess $(\eta'_1, \ldots, \eta'_\theta)$ such that for all $i \in [\theta]$, $\eta'_i \geq \eta_i/2$. Again, by Lemma 1 the success probability becomes at least $(1/2^{O(\theta)}) \prod_i \eta'_i \geq (1/2^{O(\theta)})^2 \prod_i \eta_i \geq 1/n^{O_c(1)}$, using (∗) for the final inequality.

The general low-weight case: The general case (where p_i are arbitrary) is more technical: here we need to do a "two level" hashing. The top level is by dividing into buckets, and in each bucket we get the desired "advantage" using a generalization of hitting sets for combinatorial rectangles (which itself uses

hashing) from [12]. For lack of space, we are unable to prove the general low-weight case here, but we state this generalization of (a special case of) [12] and outline its proof below. As pointed to in the introduction, [12] give ε-hitting set constructions for combinatorial rectangles, even for $\varepsilon = 1/n^2$. However in our applications, we require something slightly stronger – in particular, we need a set $\mathcal{S}$ s.t. $\mathbb{P}_{x\sim\mathcal{S}}(x \text{ in the rectangle}) \geq \varepsilon$ (roughly speaking). We however need to fool only special kinds of rectangles, given by the two conditions in Theorem 9.

Theorem 9. *For all constants $c > 0$, $m = n^c$, and $\rho \leq c \log n$, for any $\mathcal{R} \in \mathsf{CRect}(m, n)$ which satisfies the properties: 1. $\mathcal{R}$ is defined by A_i, and the rejecting probabilities q_i which satisfy $\sum_i q_i \leq \rho$ and 2. $\mathbb{P}_{x\sim[m]^n}[\mathcal{R}(x) = 1] \geq p$ $(\geq 1/n^c)$, there is an explicit set $\mathcal{S}^{n,c,\rho}_{rect}$ of size poly(n) that satisfies $\mathbb{P}_{x\sim\mathcal{S}^{n,c,\rho}_{rect}}[\mathcal{R}(x) = 1] \geq p/2^{c_{rect}\rho}$, for some c_{rect} depending on c.*

Proof sketch. The outline of the construction is as follows:

1. We guess an integer $r \leq \rho/10$ (supposed to be an estimate for $\sum_i q_i/10$).
2. Then we use a fractional hash family $\mathcal{H}^{n,r}_{frac}$ to map the indices into r buckets. This ensures that each bucket has roughly a constant weight.
3. In each bucket, we show that taking $O(1)$-wise independent spaces (Fact 4) would ensure a success probability depending on the weight of the bucket.
4. We then combine the distributions for different buckets using expander walks (this step has to be done with more care now, since the probabilities are different across buckets).

Proof of Theorem 6. The theorem follows easily from Theorems 7 and 8. Fix constant $c \geq 1$ s.t. $m, 1/\varepsilon \leq n^c$. For $C > 0$ a constant depending on c, we obtain hitting sets for thresholds of weight at least $C \log n$ from Theorem 7 and for thresholds of weight at most $C \log n$ from Theorem 8. Their union is an ε-HS for all of $\mathsf{CThr}(m, n)$.

Open Problems. It would be nice to extend our methods to weighted variants of combinatorial shapes: functions which accept an input x iff $\sum_i \alpha_i \mathbf{1}_{A_i}(x_i) = S$ where $\alpha_i \in \mathbb{R}_{\geq 0}$. The difficulty here is that having hitting sets for this sum being $\geq S$ and $\leq S$ do not imply a hitting set for '$= S$'. The simplest open case here is $m = 2$ and all A_i being $\{1\}$. However, it would also be interesting to prove formally that such weighted versions can capture much stronger computational classes.

References

1. Aleliunas, R., Karp, R.M., Lipton, R.J., Lovász, L., Rackoff, C.: Random walks, universal traversal sequences, and the complexity of maze problems. In: 20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, October 29-31, pp. 29–31. IEEE (1979)
2. Alon, N., Feige, U., Wigderson, A., Zuckerman, D.: Derandomized graph products. Computational Complexity 5, 60–75 (1995), doi:10.1007/BF01277956

3. Alon, N., Yuster, R., Zwick, U.: Color-coding. J. ACM 42(4), 844–856 (1995)
4. Armoni, R., Saks, M., Wigderson, A., Zhou, S.: Discrepancy sets and pseudorandom generators for combinatorial rectangles. In: 37th Annual Symposium on Foundations of Computer Science, Burlington, VT, pp. 412–421. IEEE Comput. Soc. Press, Los Alamitos (1996)
5. Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. J. ACM 50(4), 506–519 (2003)
6. Even, G., Goldreich, O., Luby, M., Nisan, N., Veličković, B.: Efficient approximation of product distributions. Random Structures Algorithms 13(1), 1–16 (1998)
7. Feller, W.: An Introduction to Probability Theory and its Applications, vol. 2. Wiley (1971)
8. Fredman, M.L., Komlós, J., Szemerédi, E.: Storing a sparse table with 0(1) worst case access time. J. ACM 31(3), 538–544 (1984)
9. Gopalan, P., Meka, R., Reingold, O., Zuckerman, D.: Pseudorandom generators for combinatorial shapes. In: STOC, pp. 253–262 (2011)
10. Hoory, S., Linial, N., Wigderson, A.: Expander graphs and their applications. Bulletin of the AMS 43(4), 439–561 (2006)
11. Impagliazzo, R., Wigderson, A.: $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In: Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, El Paso, Texas, May 4-6, pp. 220–229 (1997)
12. Linial, N., Luby, M., Saks, M., Zuckerman, D.: Efficient construction of a small hitting set for combinatorial rectangles in high dimension. Combinatorica 17, 215–234 (1997), doi:10.1007/BF01200907
13. Lovett, S., Reingold, O., Trevisan, L., Vadhan, S.: Pseudorandom Bit Generators That Fool Modular Sums. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 615–630. Springer, Heidelberg (2009)
14. Lu, C.-J.: Hyper-encryption against space-bounded adversaries from on-line strong extractors, pp. 257–271
15. Meka, R., Zuckerman, D.: Small-bias spaces for group products. These proceedings (2009)
16. Moser, R.A., Tardos, G.: A constructive proof of the general lovász local lemma. J. ACM 57(2) (2010)
17. Naor, J., Naor, M.: Small-bias probability spaces: Efficient constructions and applications. SIAM Journal on Computing 22(4), 838–856 (1993)
18. Nisan, N.: Pseudorandom generators for space-bounded computation. Combinatorica 12(4), 449–461 (1992)
19. Nisan, N., Wigderson, A.: Hardness vs. randomness. J. Comput. Syst. Sci. 49(2), 149–167 (1994)
20. Nisan, N., Zuckerman, D.: Randomness is linear in space. Journal of Computer and System Sciences 52(1), 43–52 (1996)
21. Rabani, Y., Shpilka, A.: Explicit construction of a small epsilon-net for linear threshold functions. SIAM J. Comput. 39(8), 3501–3520 (2010)
22. Schmidt, J.P., Siegel, A.: The analysis of closed hashing under limited randomness (extended abstract). In: STOC, pp. 224–234 (1990)
23. Shaltiel, R., Umans, C.: Pseudorandomness for approximate counting and sampling. Computational Complexity 15(4), 298–341 (2006)

Tight Bounds for Testing k-Linearity

Eric Blais[1] and Daniel Kane[2]

[1] School of Computer Science, Carnegie Mellon University
eblais@cs.cmu.edu
[2] Department of Mathematics, Stanford University
dankane@math.stanford.edu

Abstract. The function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is *k-linear* if it returns the sum (over $\mathbb{F}_2$) of exactly k coordinates of its input. We introduce strong lower bounds on the query complexity for testing whether a function is k-linear. We show that for any $k \leq \frac{n}{2}$, at least $k - o(k)$ queries are required to test k-linearity, and we show that when $k \approx \frac{n}{2}$, this lower bound is nearly tight since $\frac{4}{3}k + o(k)$ queries are sufficient to test k-linearity. We also show that non-adaptive testers require $2k - O(1)$ queries to test k-linearity.

We obtain our results by reducing the k-linearity testing problem to a purely geometric problem on the boolean hypercube. That geometric problem is then solved with Fourier analysis and the manipulation of Krawtchouk polynomials.

1 Introduction

What global properties of functions can we test with only a partial, local view of an unknown object? Property testing, a model introduced by Rubinfeld and Sudan [20], formalizes this question. In this model, a *property* of functions $\mathbb{F}_2^n \to \mathbb{F}_2$ is simply a subset of these functions. A function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is *ϵ-far* from a property $\mathcal{P}$ if for every $g \in \mathcal{P}$, the inequality $f(x) \neq g(x)$ holds for at least an ϵ fraction of the inputs $x \in \mathbb{F}_2^n$. A *q-query ϵ-tester* for $\mathcal{P}$ is a randomized algorithm that queries a function f on at most q inputs and distinguishes with probability at least $\frac{2}{3}$ between the cases where $f \in \mathcal{P}$ and where f is ϵ-far from $\mathcal{P}$. The aim of property testing is to identify the minimum number of queries required to test various properties. For more details on property testing, we recommend the recent surveys [17,18,19] and the collection [13].

Linearity testing is one of the earliest success stories in property testing. The function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is *linear* if it is of the form $f(x) = \sum_{i \in S} x_i$ for some set $S \subseteq [n]$, where the sum is taken over $\mathbb{F}_2$. Equivalently, f is linear if every pair $x, y \in \mathbb{F}_2^n$ satisfies the identity $f(x) + f(y) = f(x + y)$. Blum, Luby, and Rubinfeld [7] showed that, remarkably, linearity can be ϵ-tested with only $O(1/\epsilon)$ queries. The exact query complexity of this problem has since been studied extensively [2,3,1,15] and is well understood.

In this work, we study a closely related property: k-linearity. The function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ is *k-linear* if it is of the form $f(x) = \sum_{i \in S} x_i$ for some set $S \subseteq [n]$ of size $|S| = k$. The k-linearity property plays a fundamental role in testing properties of boolean functions. Notably, the query complexity of the k-linearity testing problem provides a lower bound for the query complexity for testing juntas [11], testing

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 435–446, 2012.
© Springer-Verlag Berlin Heidelberg 2012

low Fourier degree [9], testing computability by small-depth decision trees [9], and testing a number of other basic properties of boolean functions.

Our goal is to determine the *exact* query complexity of the k-linearity testing problem. As an initial observation, we note that for any $0 \leq k \leq n$, the query complexity for the k-linearity and $(n-k)$-linearity testing problems are identical. (See the full version of the article for the easy proof of this fact.) This observation lets us concentrate on the range $0 \leq k \leq \frac{n}{2}$ from now on; all our results also apply to the range $\frac{n}{2} < k \leq n$ by applying this identity.

Previous Work. The connection between property testing and learning theory, first established by Goldreich, Goldwasser, and Ron [14], yields a simple and non-adaptive k-linearity tester with query complexity $n + O(1/\epsilon)$. For $i = 1, 2 \ldots, n$, define $e_i \in \mathbb{F}_2^n$ to be the vector with value 1 in the ith coordinate and value 0 elsewhere. The tester queries the function on the inputs $e_1, e_2, \ldots, e_n \in \mathbb{F}_2^n$. If $f(e_i) = 1$ for exactly k indices $i \in [n]$, then f is consistent with exactly one k-linear function h. We can query the function f on $O(1/\epsilon)$ additional inputs chosen uniformly and independently at random from $\mathbb{F}_2^n$ to verify that the rest of the function f is also consistent with h. This test always accepts k-linear functions, while the functions that are ϵ-far from k-linear functions fail at least one of the two steps of the test with high probability. We call this algorithm the *learning tester* for k-linearity.

Fischer, Kindler, Ron, Safra, and Samorodnitsky [11] introduced an algorithm for testing k-linearity with roughly $O(k^2)$ queries. They also showed that for $k = o(\sqrt{n})$, non-adaptive testers—that is, testers that must fix all their queries before observing the value of the function on any of those queries—require roughly $\Omega(\sqrt{k})$ queries to test k-linearity. This implies a lower bound of $\Omega(\log k)$ queries for general (i.e., possibly adaptive) k-linearity testers for the same range of values of k.

The upper bound on the query complexity for testing k-linearity was improved implicitly by the introduction of a new algorithm for testing k-juntas—that is, testing whether a function depends on at most k variables—with only $O(k \log k)$ queries [4]. By combining this junta tester with the BLR linearity test [7], one can test k-linearity with roughly $O(k \log k)$ queries.

The first lower bound for testing k-linearity that applied to all values of k was discovered by Blais and O'Donnell [6], who, as a special case of a more general theorem on testing function isomorphism, showed that non-adaptive testers need at least $\Omega(\log k)$ queries to test k-linearity.

A much stronger bound was obtained by Goldreich [12], who showed that $\Omega(k)$ queries are required to test k-linearity non-adaptively, and that general testers require at least $\Omega(\sqrt{k})$ queries for the same task. He conjectured that this last bound could be strengthened to show that $\Omega(k)$ queries are required to test k-linearity for all $1 \leq k \leq \frac{n}{2}$.[1] Goldreich's conjecture was recently verified by Blais,

[1] Goldreich's results and conjecture are stated in terms of the slightly different problem of testing $\leq k$-linearity—the property of being a function that returns the sum over $\mathbb{F}_2^n$ of *at most* k variables. The $\leq k$-linearity and k-linearity problems are largely equivalent; see [5,12] for more details.

Brody, and Matulef [5], who proved the desired lower bound by establishing a new connection between communication complexity and property testing.

Our Results. Continuing on the line of work described above, we pose the following question: can we obtain *exact* bounds on the query complexity of the k-linearity testing problem? The results presented in this paper make significant progress on this question. Our main results are new lower bounds for general as well as for non-adaptive testing algorithms.

Theorem 1.1. *Fix $1 \leq k \leq \frac{n}{2}$. At least $k - O(k^{2/3})$ queries are required to test k-linearity.*

Theorem 1.2. *Fix $1 \leq k \leq \frac{n}{2}$. Non-adaptive testers for k-linearity need at least $2k - O(1)$ queries.*

A particularly interesting case for k-linearity testing is when $k = \frac{n}{2}$. The learning tester for $\frac{n}{2}$-linearity requires n queries, so the lower bound in Theorem 1.2 shows that no non-adaptive tester can reduce this query complexity by more than an additive constant. It is reasonable to ask whether Theorem 1.1 can be strengthened to obtain the same conclusion for adaptive testers as well. It cannot: our next result shows that there is an adaptive $\frac{n}{2}$-linearity tester that makes much fewer than n queries.

Theorem 1.3. *It is possible to test $\frac{n}{2}$-linearity with $\frac{2}{3}n + O(\sqrt{n})$ queries.*

This theorem is a special case of a more general upper bound on the query complexity for testing k-linearity for values of k that are close to $\frac{n}{2}$. The details and the proof of this more general upper bound are presented in the full version of the article.

The lower bounds in Theorems 1.1 and 1.2, as well as all previous lower bounds on the query complexity for testing $\frac{n}{2}$-linearity, proceed by establishing a lower bound on the number of queries required to distinguish $\frac{n}{2}$-linear and $(\frac{n}{2}+2)$-linear functions. Our final result shows that for this promise problem our lower bound is optimal up to the lower order error term.

Theorem 1.4. *We can distinguish $\frac{n}{2}$-linear and $(\frac{n}{2}+2)$-linear functions with $\lceil \frac{n}{2} \rceil + 1$ queries. More generally, for $\ell \geq 1$, let b be the smallest positive integer for which 2^b does not divide ℓ. It is possible to distinguish $\frac{n}{2}$-linear and $(\frac{n}{2}+2\ell)$-linear functions with $\frac{2}{3}(1-2^{-2b})n + o(n)$ queries.*

Implications. The k-linearity testing problem plays a fundamental role in the study of property testing on boolean functions. In particular, lower bounds on the query complexity of this problem imply lower bounds for the query complexity of a number of other property testing problems. Our lower bounds carry over directly to all these other problems. As a result, Theorem 1.1 sharpens several previous results. In this section, we only provide a short description of these results; the details are found in the full version of the article.

Corollary 1.5. *Fix $1 \leq k \leq \frac{n}{2}$. At least $k - O(k^{2/3})$ queries are required to test (1) k-juntas, (2) k-sparse $\mathbb{F}_2$-polynomials, (3) functions of Fourier degree at most k, (4) functions computable by depth-k decision trees, and (5) isomorphism to the function $f : x \mapsto x_1 + \cdots + x_k$.*

A property of linear functions is called *symmetric* if it is invariant under relabeling of its variables. A symmetric property $\mathcal{P}$ of linear functions is completely characterized by the function $h_{\mathcal{P}} : \{0, 1, \ldots, n\} \to \{0, 1\}$ where $h_{\mathcal{P}}(k) = 1$ iff k-linear functions are included in $\mathcal{P}$. Define $\Gamma_{\mathcal{P}}$ to be the minimum value of $\ell \in \{0, 1, \ldots, \lfloor \frac{n}{2} \rfloor\}$ for which every value of k in the range $\ell \leq k \leq n - \ell$ satisfies $h_{\mathcal{P}}(k) = h_{\mathcal{P}}(k + 2)$. This measure is closely related to the Paturi complexity of symmetric functions [16]. It also provides a lower bound on the query complexity for testing $\mathcal{P}$.

Corollary 1.6. *Let $\mathcal{P}$ be a symmetric property of linear functions. Then at least $\Gamma_{\mathcal{P}} - O(\Gamma_{\mathcal{P}}^{2/3})$ queries are required to test $\mathcal{P}$.*

Discussion of Our Results. Rare are the questions in theoretical computer science for which we can obtain exact (as opposed to asymptotic) answers. The results in this paper shows that the query complexity of the k-linearity testing problem is one of those special questions. Yet, while the fundamental nature of the k-linearity testing problem causes the determination of its exact query complexity to be of intrinsic interest, two other reasons form the main motivation for the research described in this article.

First, one main reason for studying the k-linearity testing problem is to gain a better understanding of the structure of linear functions. All the previous works on this problem yielded new insights into this structure. However, the insights into the structure of linear functions have yet to be exhausted by the current line of research. Indeed, as we will discuss below, our research uncovered new connections between the problem of testing k-linearity and the geometry of the boolean hypercube.

Second, the asymptotic bounds on query complexity hide some important questions. For example, consider the following rephrasing of our main question: what is the *difference* between the query complexities of the best $\frac{n}{2}$-linearity tester and the (naïve) learning tester? An asymptotic lower bound on the query complexity of $\frac{n}{2}$-linearity testers is too weak to shed any light on this question. In stark contrast, Theorem 1.2 shows that if we restrict our attention to non-adaptive testers, the difference is at most *constant*. Furthermore, Theorem 1.3 shows that for adaptive testers the difference is *linear* in n.

Our Techniques. We reduce the problem of testing k-linear functions to a purely geometric problem on the Hamming cube. Namely, we obtain our testing lower bound by showing that affine subspaces of large dimension intersect roughly the same fraction of the middle layers of the cube. More precisely, let $W_k \subseteq \mathbb{F}_2^n$ denote the set of vectors $x \in \mathbb{F}_2^n$ of Hamming weight k. Our main technical contribution is the following result.

Lemma 1.7. *There is a constant $c > 0$ such that for any affine subspace $V \subseteq \{0,1\}^n$ of codimension $d \leq \frac{n}{2} - cn^{2/3}$,*

$$\left| \frac{|V \cap W_{\frac{n}{2}-1}|}{|W_{\frac{n}{2}-1}|} - \frac{|V \cap W_{\frac{n}{2}+1}|}{|W_{\frac{n}{2}+1}|} \right| \leq \tfrac{1}{36} 2^{-d}.$$

We prove the lemma with Fourier analysis and with the manipulation of Krawtchouk polynomials.

The proof of our lower bound for non-adaptive testers proceeds via a similar reduction to a geometric problem on the Hamming cube. See Section 4 for the details.

2 Preliminaries

Fourier Analysis. For a finite dimensional vector space V over $\mathbb{F}_2$, the *inner product* of two functions $f, g : V \to \mathbb{R}$ is $\langle f, g \rangle = \mathbf{E}_{x \in V}[f(x) \cdot g(x)]$, where the expectation is over the uniform distribution on V. The L_2 norm of f is $\|f\|_2 := \sqrt{\langle f, f \rangle}$. A *character* of V is a group homomorphism $\chi : V \to \{1,-1\}^*$. Equivalently a character is a function $\chi : V \to \{1,-1\}$ so that for any $x, y \in V$, $\chi(x+y) = \chi(x)\chi(y)$. Define $\hat{V}$ to be the set of characters of V.

For a function $f : V \to \mathbb{R}$, the *Fourier transform* of f is the function $\hat{f} : \hat{V} \to \mathbb{R}$ given by $\hat{f}(\chi) := \langle f, \chi \rangle$. The *Fourier decomposition* of f is $f(x) = \sum_{\chi \in \hat{V}} \hat{f}(\chi)\chi(x)$. A fundamental property of the Fourier transform is that it preserves the squared L_2 norm.

Fact 2.1 (Parseval's Identity). *For any $f : V \to \mathbb{R}$, $\|f\|_2^2 = \sum_{\chi \in \hat{V}} \hat{f}(\chi)^2$.*

The *pushforward* of the function $f : V \to \mathbb{R}$ by the linear function $g : V \to W$ is defined by $(g_*(f))(x) := \frac{1}{|V|} \sum_{y \in g^{-1}(x)} [f(y)]$.

Fact 2.2. *For any linear function $g : V \to W$ and any function $f : V \to \mathbb{R}$, $\widehat{g_*(f)}(\chi) = \frac{1}{|W|} \hat{f}(\chi \circ g)$.*

Krawtchouk Polynomials. For $n > 0$ and $k = 0, 1, \ldots, n$, the (binary) *Krawtchouk polynomial* $K_k^n : \{0, 1, \ldots, n\} \to \mathbb{Z}$ is defined by

$$K_k^n(m) = \sum_{j=0}^{k} (-1)^j \binom{m}{j} \binom{n-m}{k-j}.$$

The generating function representation of the Krawtchouk polynomial $K_k^n(m)$ is $K_k^n(m) = [x^k]\,(1-x)^m(1+x)^{n-m}$. Krawtchouk polynomials satisfy a number of useful properties. In particular, we use the following identities in our proofs.

Fact 2.3. *Fix $n > 0$. Then*

i. For every $2 \leq k \leq n$, $K_k^n(m) - K_{k-2}^n(m) = K_k^{n+2}(m+1)$.
ii. $\sum_{k=0}^{n} K_k^n(m)^2 = (-1)^m K_n^{2n}(2m)$.

iii. *For every* $0 \leq d \leq \frac{n}{2}$, $\sum_{j=0}^{d} \binom{d}{j}(-1)^j K^n_{\frac{n}{2}}(2j+2) = 2^{2d} K^{n-2d}_{\frac{n}{2}-d}(2)$.

iv. $K^{2n}_n(2m+1) = 0$ *and* $(-1)^m K^{2n}_n(2m)$ *is positive and decreasing in* $\min\{m, n-m\}$.

Fact 2.4. *Fix* $n > 0$ *and* $-\frac{n}{2} \leq k \leq \frac{n}{2}$. *Then*

$$K^n_{\frac{n}{2}+k}(m) = \frac{2^{n-1} i^m}{\pi} \int_0^{2\pi} \sin^m \theta \cos^{n-m} \theta e^{i2k\theta} \, \mathrm{d}\theta.$$

Krawtchouk polynomials are widely used in coding theory [22] and appear in our proofs because of their close connection with the Fourier coefficients of the (Hamming weight indicator) function $I_{W_k} : \mathbb{F}_2^n \to \{0,1\}$ defined by $I_{W_k}(x) = \mathbf{1}_{|x|=k}$. With the Hamming weight of the vector $\alpha = (\alpha_1, \ldots, \alpha_n) \in \{0,1\}^n$ defined as $|\alpha| = \sum_{i=1}^n \alpha_i$, the connection is formulated as follows.

Fact 2.5. *Fix* $0 \leq k \leq n$, *and* $\alpha \in \{0,1\}^n$. *Then* $\widehat{I_{W_k}}(\alpha) = 2^{-n} K^n_k(|\alpha|)$.

For a more thorough introduction to Krawtchouk polynomials and for the proofs of these facts, see [21,22] and the full version of this article.

Property Testing. The proof of Theorem 1.1 uses the following standard property testing lemma.

Lemma 2.6. *Let* $\mathcal{D}_{\text{yes}}$ *and* $\mathcal{D}_{\text{no}}$ *be any two distributions over functions* $\mathbb{F}_2^n \to \mathbb{F}_2$. *If for every set* $X \subseteq \mathbb{F}_2^n$ *of size* $|X| = q$ *and any vector* $r \in \mathbb{F}_2^q$ *we have that* $\left|\Pr_{f \sim \mathcal{D}_{\text{yes}}}[f(X) = r] - \Pr_{f \sim \mathcal{D}_{\text{no}}}[f(X) = r]\right| < \frac{1}{36} 2^{-q}$, *then any algorithm that distinguishes functions drawn from* $\mathcal{D}_{\text{yes}}$ *from those drawn from* $\mathcal{D}_{\text{no}}$ *with probability at least* $\frac{2}{3}$ *makes at least* $q+1$ *queries.*

Lemma 2.6 follows from Yao's Minimax Principle [23]. The proof of this result can be found in [10,8].

3 Proof of the General Lower Bound

Proof (of Theorem 1.1). We first prove the special case where $k = \frac{n}{2} - 1$. There is a natural bijection between linear functions $\mathbb{F}_2^n \to \mathbb{F}_2$ and vectors in $\mathbb{F}_2^n$: associate $f(x) = \sum_{i \in S} x_i$ with the vector $\alpha \in \mathbb{F}_2^n$ whose coordinates satisfy $\alpha_i = 1$ iff $i \in S$. Note that $f(x) = \alpha \cdot x$.

For $0 \leq \ell \leq n$, let $W_\ell \subseteq \mathbb{F}_2^n$ denote the set of elements of Hamming weight ℓ. Fix any set $X \subseteq \mathbb{F}_2^n$ of $q < \frac{n}{2} - O(n^{2/3})$ queries and any response vector $r \in \mathbb{F}_2^q$. The set of linear functions that return the response vector r to the queries in X corresponds in our bijection to an affine subspace $V \subseteq \mathbb{F}_2^n$ of codimension q. This is because for each $x \in X$, the requirement that $f(x) = r_i$ imposes an affine linear relation on f. By Lemma 1.7, this subspace satisfies the inequality

$$\left| \frac{|V \cap W_{\frac{n}{2}-1}|}{|W_{\frac{n}{2}-1}|} - \frac{|V \cap W_{\frac{n}{2}+1}|}{|W_{\frac{n}{2}+1}|} \right| \leq \tfrac{1}{36} 2^{-q}. \tag{1}$$

Define $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ to be the uniform distributions over $(\frac{n}{2}-1)$-linear and $(\frac{n}{2}+1)$-linear functions, respectively. By our bijection, $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ correspond to the uniform distributions over $W_{\frac{n}{2}-1}$ and $W_{\frac{n}{2}+1}$. As a result, the probability that a function drawn from $\mathcal{D}_{\text{yes}}$ or from $\mathcal{D}_{\text{no}}$ returns the response r to the set of queries X is

$$\Pr_{f\sim\mathcal{D}_{\text{yes}}}[f(X)=r]=\frac{|V\cap W_{\frac{n}{2}-1}|}{|W_{\frac{n}{2}-1}|}\quad\text{and}\quad\Pr_{f\sim\mathcal{D}_{\text{no}}}[f(X)=r]=\frac{|V\cap W_{\frac{n}{2}+1}|}{|W_{\frac{n}{2}+1}|}.$$

So (1) and Lemma 2.6 imply that at least $\frac{n}{2}-O(n^{2/3})$ queries are required to distinguish $(\frac{n}{2}-1)$-linear and $(\frac{n}{2}+1)$-linear functions. All $(\frac{n}{2}+1)$-linear functions are $\frac{1}{2}$-far from $(\frac{n}{2}-1)$-linear functions, so this completes the proof of the theorem for $k=\frac{n}{2}-1$.

For other values of k, we apply a simple padding argument. When $k<\frac{n}{2}-1$, modify $\mathcal{D}_{\text{yes}}$ and $\mathcal{D}_{\text{no}}$ to be uniform distributions over k-linear and $(k+2)$-linear functions, respectively, under the restriction that all coordinates in the sum taken from the set $[2k+2]$. This modification with $k=\frac{n}{2}-2$ shows that $\frac{n}{2}-O(n^{2/3})$ queries are required to distinguish $(\frac{n}{2}-2)$- and $\frac{n}{2}$-linear functions; this implies the lower bound in the theorem for the case $k=\frac{n}{2}$. □

Proof (of Lemma 1.7). For any set $A\subseteq\mathbb{F}_2^n$, define $I_A:\mathbb{F}_2^n\to\{0,1\}$ to be the indicator function for A. For a given function $f:\mathbb{F}_2^n\to\{0,1\}$, let us write $\mathbf{E}[f]$ as shorthand for $\mathbf{E}_x[f(x)]$ where the expectation is over the uniform distribution of $x\in\mathbb{F}_2^n$. Similarly, for two functions f,g, we write $\mathbf{E}[f\cdot g]$ as short-hand for $\mathbf{E}_x[f(x)\cdot g(x)]$.

For any subsets $A,B\subseteq\mathbb{F}_2^n$, $|A\cap B|=2^n\cdot\mathbf{E}[I_A\cdot I_B]$. Since $|W_{\frac{n}{2}-1}|=|W_{\frac{n}{2}+1}|=\binom{n}{\frac{n}{2}-1}$,

$$\left|\frac{|V\cap W_{\frac{n}{2}-1}|}{|W_{\frac{n}{2}-1}|}-\frac{|V\cap W_{\frac{n}{2}+1}|}{|W_{\frac{n}{2}+1}|}\right|=\frac{2^n}{\binom{n}{\frac{n}{2}-1}}\cdot\mathbf{E}\big[I_V\cdot(I_{W_{\frac{n}{2}-1}}-I_{W_{\frac{n}{2}+1}})\big].$$

The subspace V can be defined by a set $S\subseteq[n]$ of size $|S|=d$ and an affine-linear function $f:\{0,1\}^{n-d}\to\{0,1\}^d$, where $x\in V$ iff $x_S=f(x_{\bar{S}})$. Define I_m^S and $I_m^{\bar{S}}$ to be indicator functions for $|x_S|=m$ and $|x_{\bar{S}}|=m$, respectively. Then

$$\mathbf{E}[I_V\cdot(I_{W_{\frac{n}{2}-1}}-I_{W_{\frac{n}{2}+1}})]=\sum_{m=0}^{d}\mathbf{E}\left[I_V\cdot I_m^S\cdot(I^{\bar{S}}_{\frac{n}{2}-m-1}-I^{\bar{S}}_{\frac{n}{2}-m+1})\right].$$

Let $U\subseteq\{0,1\}^S$ be the image of f. Let $d'=\dim(U)$. Define $h_m:\{0,1\}^S\to[-1,1]$ by setting $h_m(u)=\mathbf{E}_{x\in\{0,1\}^{\bar{S}}}[I_V(x,u)\cdot(I^{\bar{S}}_{\frac{n}{2}-m-1}(x)-I^{\bar{S}}_{\frac{n}{2}-m+1}(x))]$. Note that $h_m=f_*\left(I^{\bar{S}}_{\frac{n}{2}-m-1}-I^{\bar{S}}_{\frac{n}{2}-m+1}\right)$. Notice also that h_m is supported on U. We have

$$\mathbf{E}[I_V\cdot(I_{W_{\frac{n}{2}-1}}-I_{W_{\frac{n}{2}+1}})]=\sum_{m=0}^{d}\mathbf{E}\left[I_m^S\cdot h_m\right]=\sum_{m=0}^{d}\mathbf{E}\left[I_m^S\cdot\mathbf{1}_U\cdot h_m\right].\qquad(2)$$

Two applications of the Cauchy-Schwarz inequality yield

$$\sum_{m=0}^{d} \mathbf{E}\left[I_m^S \cdot \mathbf{1}_U \cdot h_m\right] \leq \sum_{m=0}^{d} \|I_m^S \cdot \mathbf{1}_U\|_2 \cdot \|h_m\|_2 \leq \sqrt{\sum_{m=0}^{d} \|I_m^S \cdot \mathbf{1}_U\|_2^2} \cdot \sqrt{\sum_{m=0}^{d} \|h_m\|_2^2}. \tag{3}$$

We bound the two terms on the right-hand side. The first term satisfies

$$\sum_{m=0}^{d} \|I_m^S \cdot \mathbf{1}_U\|_2^2 = \sum_{m} \mathbf{E}_x[I_m^S(x)^2 \cdot \mathbf{1}_U] = \mathbf{E}_x\left[\mathbf{1}_U \sum_{m} I_m^S(x)^2\right] = 2^{d'-d}, \tag{4}$$

where the last equality follows from the fact that for every $x \in \{0,1\}^n$, there is exactly one m for which $I_m^S(x) = 1$.

We now examine the second term. By Parseval's Identity, we have that $\|h_m\|_2^2 = \sum_{\alpha \in \{0,1\}^S} \hat{h}_m(\chi_\alpha)^2$. Suppose that the image of f has dimension $d' \leq d$. Then, since h_m is a pushforward,

$$\hat{h}_m(\chi) = 2^{-d}\left(\widehat{I^{\bar{S}}_{\frac{n}{2}-m-1}}(\chi \circ f) - \widehat{I^{\bar{S}}_{\frac{n}{2}-m+1}}(\chi \circ f)\right).$$

The characters $\chi \circ f$ depend only on the restriction of χ to $f(\{0,1\}^{\bar{S}})$. Thus these characters all lie in some subspace $W \subseteq \widehat{\{0,1\}^{\bar{S}}}$ of dimension d', with each character appearing $2^{d-d'}$ times. Thus, we have that

$$\|h_m\|_2^2 = 2^{-d-d'} \sum_{\chi \in W} \left(\widehat{I^{\bar{S}}_{\frac{n}{2}-m-1}}(\chi) - \widehat{I^{\bar{S}}_{\frac{n}{2}-m+1}}(\chi)\right)^2.$$

For any set $\chi \subseteq \bar{S}$, we can apply Facts 2.5 and 2.3(i) to obtain

$$\widehat{I^{\bar{S}}_{\frac{n}{2}-m+1}}(\chi) - \widehat{I^{\bar{S}}_{\frac{n}{2}-m-1}}(\chi) = 2^{-(n-d)} K^{n-d+2}_{\frac{n}{2}-m+1}(|\chi|+1).$$

Therefore, $\sum_{m=0}^{d} \|h_m\|_2^2 = 2^{-2n+d-d'} \sum_m \sum_{\chi \in W} K^{n-d+2}_{\frac{n}{2}-m+1}(|\chi|+1)^2$ and by Fact 2.3(ii),

$$\sum_{m=0}^{d} \|h_m\|_2^2 \leq 2^{-2n+d-d'} \sum_{\chi \in W} (-1)^{|\chi|+1} K^{2(n-d+1)}_{n-d+1}(2|\chi|+2). \tag{5}$$

There exist some d' coordinates such that the projection of W onto those coordinates is surjective. Therefore the number of elements of W with weight at most ℓ is at most $\sum_{j=1}^{\ell} \binom{d'}{j}$. We also have a similar bound on the number of elements of W of size at least $n-d-\ell$. Therefore, since by Fact 2.3(iv) the summand in (5) is decreasing in $\min(|\chi|, n-d-|\chi|)$, we have

$$\sum_{m=0}^{d} \|h_m\|_2^2 \leq 2^{-2n+d-d'+1} \sum_{j=0}^{d'} \binom{d'}{j} (-1)^{j+1} K^{2(n-d+1)}_{n-d+1}(2j+2).$$

By Fact 2.3(iii), the sum on the right-hand side evaluates to $-K_{n-d-d'+1}^{2(n-d-d'+1)}(2)$. We can then apply the generating function representation of Krawtchouk polynomials to obtain

$$\begin{aligned}\sum_{m=0}^{d} \|h_m\|_2^2 &\le -2^{-2n+d+d'+1}[x^{n-d-d'+1}](1-x)^2(1+x)^{2(n-d-d')} \\ &= 2^{-2n+d+d'+2}\left(\binom{2(n-d-d')}{n-d-d'} - \binom{2(n-d-d')}{n-d-d'-1}\right) \\ &= 2^{-d-d'}\Theta(n-d-d')^{-3/2} = 2^{-d-d'}O\left((n-2d)^{-3/2}\right).\end{aligned}$$

Thus we have that

$$\begin{aligned}\mathbf{E}[I_V \cdot (I_{W_{\frac{n}{2}+1}} - I_{W_{\frac{n}{2}-1}})] &\le \sqrt{2^{d'-d}}\sqrt{2^{-d-d'}O\left((n-2d)^{-3/2}\right)} \\ &= 2^{-d}O\left((n-2d)^{-3/4}\right).\end{aligned}$$

When $d = \frac{n}{2} - cn^{2/3}$ for some large enough constant $c > 0$, we therefore have $\mathbf{E}[I_V \cdot (I_{W_{\frac{n}{2}+1}} - I_{W_{\frac{n}{2}-1}})] < \frac{1}{36}\binom{n}{\frac{n}{2}-1}2^{-n-d}$ and the lemma follows. □

4 Non-adaptive Lower Bound

The strategy for the proof of Theorem 1.2 is similar to that of the proof of the general lower bound in the last section. Once again, we reduce the problem to a geometric problem on the Hamming cube. The main difference is that in this case we prove the following lemma.

Lemma 4.1. *There is a constant $d_0 > 0$ such that for any linear subspace $V \subseteq \{0,1\}^n$ of codimension $d \le n - d_0$,*

$$\sum_{x \in \{0,1\}^n/V} \left(\frac{|(V+x) \cap W_{\frac{n}{2}-1}|}{|W_{\frac{n}{2}-1}|} - \frac{|(V+x) \cap W_{\frac{n}{2}+1}|}{|W_{\frac{n}{2}+1}|}\right)^2 \le \tfrac{1}{3}2^{-d}.$$

Proof (sketch). As in the last section, define $I_A : \{0,1\}^n \to \{0,1\}$ to be the indicator function for the set $A \subseteq \{0,1\}^n$. To prove Lemma 4.1, we want to show that

$$\sum_{x \in \{0,1\}^n/V} \left(\frac{\mathbf{E}[I_{V+x} \cdot I_{W_{\frac{n}{2}-1}}]}{\mathbf{E}[I_{W_{\frac{n}{2}-1}}]} - \frac{\mathbf{E}[I_{V+x} \cdot I_{W_{\frac{n}{2}+1}}]}{\mathbf{E}[I_{W_{\frac{n}{2}+1}}]}\right)^2 \le \tfrac{1}{3}2^{-d}.$$

Let $D_{\frac{n}{2}} = I_{W_{\frac{n}{2}-1}} - I_{W_{\frac{n}{2}+1}}$, and note that $\mathbf{E}[I_{W_{\frac{n}{2}-1}}] = \mathbf{E}[I_{W_{\frac{n}{2}+1}}] = \binom{n}{\frac{n}{2}-1}/2^n$. Then the above inequality is equivalent to

$$\sum_{x \in \{0,1\}^n/V} \mathbf{E}[I_{V+x} \cdot D_{\frac{n}{2}}]^2 \le \tfrac{1}{3}2^{-d} \cdot \left(\frac{\binom{n}{\frac{n}{2}-1}}{2^n}\right)^2.$$

Let $\pi : \{0,1\}^n \to \{0,1\}^n/V$ be the projection map. Notice that $\mathbf{E}[I_{V+x} \cdot D_{\frac{n}{2}}] = \pi_* D_{\frac{n}{2}}(x)$. By Parseval's Theorem and Fact 2.2,

$$\mathbf{E}_{x \in \{0,1\}^n/V}[\pi_* D_{\frac{n}{2}}(x)^2] = |\pi_* D_{\frac{n}{2}}|_2^2 = \sum_{\chi \in \widehat{\{0,1\}^n/V}} \widehat{\pi_* D_{\frac{n}{2}}}(\chi) = 2^{d-n} \sum_{\chi \in V^\perp} \widehat{D_{\frac{n}{2}}}(\chi).$$

Where above $V^\perp$ is the set of pullbacks of $\widehat{\{0,1\}^n/V}$ to $\widehat{\{0,1\}^n}$, which is the space of characters of $\{0,1\}^n$ that are trivial on V.

By Fact 2.5, $\widehat{D}_{\frac{n}{2}}(\chi) = 2^{-n} K^{n+2}_{\frac{n}{2}+1}(|\chi|+1)$. By Fact 2.3(iv), the absolute value of this is 0 for $|\chi|$ even and otherwise decreasing in $\min(|\chi|, n-|\chi|)$. Since there are at most $2\sum_{j=0}^{\ell} \binom{d}{j}$ $\chi \in V^\perp$ with $\min(|\chi|, n-|\chi|) < \ell$, the above sum is less than it would be if there were 2 $\chi \in V^\perp$ with $|\chi| = 0$, $2\left(\binom{d}{2} + \binom{d}{1}\right)$ with $|\chi| = 2$, $2\left(\binom{d}{4} + \binom{d}{3}\right)$ with $|\chi| = 4$, and so on. Hence

$$\sum_{x \in \{0,1\}^n/V} \mathbf{E}[I_{V+x} \cdot D_{\frac{n}{2}}]^2 \leq 2^{2-2n-d} \sum_{m=0}^{d} \binom{d}{m} K^{n+2}_{\frac{n}{2}+1}(m+1)^2.$$

By Fact 2.4, we can expand the sum on the right-hand side of the inequality into a double integral. Namely,

$$\sum_{m=0}^{d} \binom{d}{m} K^{n+2}_{\frac{n}{2}+1}(m+1)^2$$
$$= \frac{2^{2n}}{\pi^2} \iint \sum_{m=0}^{d} \binom{d}{m} (-1)^m \sin^{m+1}\theta \sin^{m+1}\phi \cos^{n-m+1}\theta \cos^{n-m+1}\phi \, \mathrm{d}\theta \, \mathrm{d}\phi.$$

As we show in the full version of this article, we can manipulate the trigonometric functions and apply the Cauchy-Schwarz inequality to obtain

$$\sum_{m=0}^{d} \binom{d}{m} K^{n+2}_{\frac{n}{2}+1}(m+1)^2 \leq O\big(2^{2n} d^{-\frac{1}{2}} (n-d+1)^{-\frac{3}{2}}\big). \tag{6}$$

Using this bound, we obtain

$$\sum_{x \in \{0,1\}^n/V} \mathbf{E}[I_{V+x} \cdot D_{\frac{n}{2}}]^2 \leq O\left(2^{-d} d^{-\frac{1}{2}} (n-d+1)^{-\frac{3}{2}}\right).$$

Note that $2^{-d} \cdot \left(\frac{\binom{n}{\frac{n}{2}}}{2^n}\right)^2 = \Theta(2^{-d} n^{-1/2})$. If $d < n/2$, $\sum_{x \in \{0,1\}^n/V} \mathbf{E}[I_{V+x} \cdot D_{\frac{n}{2}}]^2$ is $O(2^{-d} n^{-3/2})$, which is too small. Otherwise it is $O(2^{-d} n^{-1/2} (n-d+1)^{-3/2})$, which is too small as long as $n-d$ is bigger than a sufficiently large constant. □

For the details of the proof of Lemma 4.1 as well as the proof of Theorem 1.2 using this lemma, see the complete version of the article.

5 Upper Bounds

We provide a sketch of the proofs of Theorems 1.3 and 1.4 in this section.

Let us begin by describing the algorithm for distinguishing $\frac{n}{2}$-linear and $(\frac{n}{2}+2)$-linear functions. The starting point for this algorithm is an elementary observation: $\frac{n}{2} \not\equiv \frac{n}{2}+2 \pmod 4$. For a set $S \subseteq [n]$, let $x_S \in \mathbb{F}_2^n$ be the vector with value 1 at each coordinate in S and 0 in the remaining coordinates. Query $f(x_{\{1,2\}}), f(x_{\{3,4\}}), \ldots, f(x_{\{n-1,n\}})$. Let m denote the number of queries that returned 1. Define the set $T = \{2i : f(x_{\{2i-1,2i\}}) = 0\}$. Query $f(x_T)$; if $f(x_T) = 1$, increment m by 2. When f is k-linear, we have $m \equiv k \pmod 4$ and this algorithm completes the proof of the first claim in Theorem 1.4.

The algorithm that proves the more general claim in Theorem 1.4 is obtained by applying the same approach recursively. When $b > 0$ is the minimum integer for which $2^b \nmid \ell$ and f is k-linear, we can determine the value of k modulo 2^b in b rounds and thereby distinguish between the cases where $k = \frac{n}{2}$ and $k = \frac{n}{2} + 2\ell$.

Finally, to complete the proof of Theorem 1.3, we essentially combine the Blum–Luby–Rubinfeld (BLR) linearity test [7] with the algorithm described above. The BLR test rejects functions that are far from linear; after that, the problem of testing k-linearity is essentially equivalent to that of distinguishing k-linear from functions that are k'-linear for some $k' \neq k$. For the complete proofs of Theorems 1.3 and 1.4, see the full version of this article.

References

1. Bellare, M., Coppersmith, D., Håstad, J., Kiwi, M., Sudan, M.: Linearity testing in characteristic two. IEEE Trans. on Information Theory 42(6), 1781–1795 (1996)
2. Bellare, M., Goldwasser, S., Lund, C., Russell, A.: Efficient probabilistically checkable proofs and applications to approximations. In: Proc. of the 25th Symposium on Theory of Computing, pp. 294–304 (1993)
3. Bellare, M., Sudan, M.: Improved non-approximability results. In: Proc. of the 26th Symposium on Theory of Computing, pp. 184–193 (1994)
4. Blais, E.: Testing juntas nearly optimally. In: Proc. 41st Annual ACM Symposium on Theory of Computing (STOC), pp. 151–158 (2009)
5. Blais, E., Brody, J., Matulef, K.: Property testing lower bounds via communication complexity. In: Proc. of the 26th Conference on Computational Complexity (2011)
6. Blais, E., O'Donnell, R.: Lower bounds for testing function isomorphism. In: Proc. of the 25th Conference on Computational Complexity, pp. 235–246 (2010)
7. Blum, M., Luby, M., Rubinfeld, R.: Self-testing/correcting with applications to numerical problems. J. Comput. Syst. Sci. 47, 549–595 (1993)
8. Chakraborty, S., García-Soriano, D., Matsliah, A.: Nearly tight bounds for testing function isomorphism. In: Proc. 22nd Symposium on Discrete Algorithms, pp. 1683–1702 (2011)
9. Diakonikolas, I., Lee, H.K., Matulef, K., Onak, K., Rubinfeld, R., Servedio, R.A., Wan, A.: Testing for concise representations. In: Proc. 48th Symposium on Foundations of Computer Science, pp. 549–558 (2007)
10. Fischer, E.: The art of uninformed decisions. Bulletin of the EATCS 75, 97–126 (2001)

11. Fischer, E., Kindler, G., Ron, D., Safra, S., Samorodnitsky, A.: Testing juntas. J. Comput. Syst. Sci. 68(4), 753–787 (2004)
12. Goldreich, O.: On Testing Computability by Small Width OBDDs. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J. (eds.) APPROX and RANDOM 2010, LNCS, vol. 6302, pp. 574–587. Springer, Heidelberg (2010)
13. Goldreich, O. (ed.): Property Testing. LNCS, vol. 6390. Springer, Heidelberg (2010)
14. Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. J. of the ACM 45(4), 653–750 (1998)
15. Kaufman, T., Litsyn, S., Xie, N.: Breaking the ϵ-soundness bound of the linearity test over GF(2). SIAM J. on Computing 39, 1988–2003 (2010)
16. Paturi, R.: On the degree of polynomials that approximate symmetric boolean functions (preliminary version). In: Proc. STOC 1992, pp. 468–474 (1992)
17. Ron, D.: Property testing: A learning theory perspective. Found. Trends Mach. Learn. 1, 307–402 (2008)
18. Ron, D.: Algorithmic and analysis techniques in property testing. Found. Trends Theor. Comput. Sci. 5, 73–205 (2010)
19. Rubinfeld, R., Shapira, A.: Sublinear time algorithms. Technical Report TR11-013, ECCC (2011)
20. Rubinfeld, R., Sudan, M.: Robust characterizations of polynomials with applications to program testing. SIAM J. Comput. 25(2), 252–271 (1996)
21. Szegő, G.: Orthogonal Polynomials, 4th edn. Colloquium Publications, vol. 23. AMS (1975)
22. Van Lint, J.H.V.: Introduction to Coding Theory, 3rd edn. Graduate Texts in Mathematics, vol. 86. Springer (1999)
23. Yao, A.C.: Probabilistic computations: towards a unified measure of complexity. In: Proc. 18th Sym. on Foundations of Comput. Sci., pp. 222–227 (1977)

Pseudorandomness for Linear Length Branching Programs and Stack Machines

Andrej Bogdanov[1,⋆], Periklis A. Papakonstantinou[2], and Andrew Wan[2,]

[1] Department of Computer Science and Engineering and ITCSC
Chinese University of Hong Kong
andrejb@cse.cuhk.edu.hk
[2] Institute for Theoretical Computer Science, IIIS⋆⋆
Tsinghua University, P.R. China
{papakons,andrew}@tsinghua.edu.cn

Abstract. We show the existence of an explicit pseudorandom generator G of linear stretch such that for every constant k, the output of G is pseudorandom against:

- *Oblivious* branching programs over alphabet $\{0,1\}$ of length kn and size $2^{O(n/\log n)}$ on inputs of size n.
- *Non-oblivious* branching programs over alphabet Σ of length kn, provided the size of Σ is a power of 2 and sufficiently large in terms of k.
- The model of logarithmic space randomized Turing Machines (over alphabet $\{0,1\}$) extended with an unbounded stack that make k passes over their randomness.

The construction of the pseudorandom generator G is the same as in our previous work (FOCS 2011). The results here rely on a stronger analysis of the construction. For the last result we give a length-efficient simulation of stack machines by non-deterministic branching programs. (over a large alphabet) whose accepting computations have a unique witness.

1 Introduction

We consider the problem of constructing an explicit pseudorandom distribution for branching programs of bounded width. A branching program with input symbols from the alphabet Σ, is a directed acyclic graph with a unique start vertex, where every non-sink vertex is labeled by one of n variables and has $|\Sigma|$ outgoing arcs, each labelled with $\sigma \in \Sigma$, and each sink vertex is labeled by an output value "accept" or "reject." The branching program computes a Boolean function over n variables in the natural way: it begins at the start vertex, reads the value of the variable at that vertex, and follows the corresponding arc to the

⋆ Work partially supported by RGC GRF grants CUHK410309 and CUHK410111.
⋆⋆ This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, and the National Natural Science Foundation of China Grant 61033001, 61061130540, 61073174, 61050110452, 61150110163.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 447–458, 2012.
© Springer-Verlag Berlin Heidelberg 2012

next vertex. When it reaches a sink vertex, it halts and outputs the corresponding label.

Fix an alphabet Σ. A family of distributions $\mathrm{P}\colon \Sigma^{s(n)} \to \Sigma^n$ is *pseudorandom* with seed length $s(n) < n$, and bias $\epsilon(n)$ for a class of functions $\mathcal{F}$ if for every $f \in \mathcal{F}$ in n inputs,

$$\left|\mathbf{E}_{\mathrm{P}}[f(\mathrm{P})] - \mathbf{E}_{\mathrm{U}}[f(\mathrm{U})]\right| \leq \epsilon(n).$$

where U is the uniform distribution on n symbols.

The problem of constructing explicit, unconditionally pseudorandom distributions for various models of computation has been met with the most success for two types of models, the first being small-depth computation [AB84, AW85, Nis91, Bra10]. The second type is space-bounded computation, for which branching programs play an important role: the computation of a randomized Turing Machine that uses n random bits and space S can be modeled as a width 2^S branching program, where the inputs to the program are the n random bits. The pseudorandom generators constructed by Nisan [Nis92] and by Impagliazzo et al. [INW94] use seed length $O(\log^2 n)$ to fool fixed input-order, $\mathrm{poly}(n)$ width, read-once branching programs.

Pseudorandom generators for space-bounded algorithms take advantage of the limited communication that can occur between parts of the computation, and are typically based on the following principle: a space-bounded algorithm records a small amount of information between stages of its computation, so randomness may be *reused* from one stage to the next without substantially altering performance.

However, in the constructions mentioned, the ability to recycle randomness relies not only on limited communication between the computation stages, but also on the nature of its access to the randomness. The random bits cannot be accessed too often and the order in which they are accessed must be known in advance. A natural goal is to construct distributions that remain pseudorandom without these access restrictions.

Recent work [BPW11] makes some progress towards removing these restrictions, giving the first pseudorandom generator (with linear stretch) for read-once branching programs under any ordering of the inputs. However, the access to the randomness is still restricted: the branching program is read-once and *oblivious*, i.e., it reads bits in an order independent of their values.

One motivation for our work comes from the problem of derandomizing log-space stack machines which make a bounded number of sequential passes over their randomness. These machines were proposed by David et al. [DNPS11] as a model of randomized polynomial time with limited access to randomness.[1] Without the random tape access restriction, randomized stack machines characterize randomized polynomial time [Coo71]. If they are allowed one pass over the randomness, however, such machines can be simulated deterministically. David et al. suggest studying what happens between these two extreme cases.

[1] They are also known as auxiliary pushdown automata, see the full version of [DP10] for terminology.

1.1 Results

In this work we show that the distribution in [BPW11] (with different parameters) is pseudorandom even for bounded-width branching programs that have linear length. In other words, the input symbols may be accessed adaptively and arbitrarily many times, provided that the total number of accesses is $O(n)$.

Theorem 1. *For every $k > 1$ there exist constants ρ, γ, λ and and an explicit pseudorandom distribution family $\mathrm{P} : \Sigma^{(1-\rho)n} \to \Sigma^n$, where $\Sigma = \{0,1\}^\lambda$ so that for every n,*

$$\left|\mathbf{E}_{\mathrm{P}}[F(\mathrm{P})] - \mathbf{E}_{\mathrm{U}}[F(\mathrm{U})]\right| = 2^{-\Omega(n)}$$

for every length kn, width $2^{\gamma n}$ branching program $F\colon \Sigma^n \to \{0,1\}$ over n inputs.

Here the constants ρ, γ, λ are inverse exponential in k; see the end of Section 3.2 for the precise dependence on k. For oblivious branching programs, we obtain a stronger form of the theorem in which $\Sigma = \{0,1\}$. This theorem is stated and proved as Theorem 2 in Section 3.1.

As an example application, consider the problem of identity testing for linear-size arithmetic formulas (see [KI04]). Let f be a linear-size arithmetic formula on inputs of length n coming from some subset S of a field $\mathbb{F}$. Such a formula can be computed[2] by a boolean oblivious branching program of linear length and width $|\mathbb{F}|^{O(\log n)}$. The Schwarz-Zippel lemma says that if f is nonzero, then $f(\mathrm{U})$ takes value zero with probability at most $\deg(f)/|S|$. By Theorem 2, $f(\mathrm{P})$ takes value zero with probability at most $\deg(f)/|S| - 2^{-\Omega(n)}$, as long as $|\mathbb{F}| \ll 2^{n/(\log n)^2}$.

Our proof of Theorem 1 immediately applies to non-deterministic branching programs with unique witnesses as well; we apply this result to fool (non-uniform) randomized Turing Machines over alphabet $\{0,1\}$ extended with an unbounded stack, henceforth called *stack machines*, which make a constant number of passes over their randomness tape. As mentioned previously, randomized log-space stack-machines characterize probabilistic polynomial time. David et al. [DNPS11] showed that pseudorandom generators that fool polynomial size circuits of depth $d(n) = \Omega(\log n)$ also fool stack machines that make $2^{O(d(n))}$ passes over their randomness. It is conceivable that one can derandomize stack machines that make a sub-polynomial (and in particular constant) number of passes over the randomness without the full derandomization of BPNC^1.

In the full version, we show that our pseudorandom distribution fools stack machines that make k sequential passes over their input. This in particular implies that we can replace the random tape of a randomized stack machine (restricted to make k passes over its randomness – and unrestricted in every other tape) with our distribution. Here k is the same constant as in Theorem 1. Previously, no nontrivial simulation was known even for $k = 2$.

[2] Lemma 1 of [BPW11] shows this for boolean formulas. The extension to larger domains is straightforward.

1.2 Techniques

Fooling Branching Programs. In order to construct pseudorandomness that can be accessed in arbitrary order, the approach in [BPW11] addresses the issue of limited communication in the following way. Consider the computation as occurring in two halves, where only a bit of information (however, it may be computed in an arbitrary fashion) is remembered from each half. The distribution P constructed in [BPW11] was shown to satisfy the following property:

> For every pair of Boolean functions $f, g : \{0,1\}^{n/2} \to \{0,1\}$ and every equipartition (I, J) of $[n]$, the joint distribution $(f(\mathrm{P}|_I), g(\mathrm{P}|_J))$ is close (in statistical distance) to the distribution $(f(U|_I), g(U|_J))$.

The distribution output by the base generator of the expander-based construction from [INW94] satisfies the above property for any fixed equipartition such as $\{1, \cdots, n/2\} \cup \{n/2+1, \cdots, n\}$ (but not all at the same time).

The distribution from [BPW11] has the advantage that it is pseudorandom for every equipartition and hence will accomodate access to the inputs under every ordering. In fact, it was observed in [BPW11], without proof, that the distribution remains pseudorandom for any f and g which depend on at most $(1 - \Omega(1))n$ of the input bits. We prove this more general lemma (Lemma 1) in Section 2. In the lemma, inputs to f and g can now be shared, so one might expect that the distribution will remain pseudorandom with multiple accesses.

Now consider an oblivious branching program of length kn. We split the computation into t stages, for some large enough t that will be set later. The result of the computation can then be stated as a sum over w^t products of t Boolean functions, each over nk/t variables. We do not argue that the outputs of these functions look independent; instead, we show in Section 3.1 that each summand can always be rewritten as a pair of functions (f, g), where f and g each depend on at most $(1 - \Omega(1))n$ bits, and then apply Lemma 1.

A more complicated argument is required if the branching program is non-oblivious; under the previous decomposition, a single stage of the computation may depend on all n input symbols. In fact, in this case we do not know how to construct a pseudorandom distribution with symbols from $\{0,1\}$. However, we can achieve this over any sufficiently large (in terms of k) alphabet Σ, where $|\Sigma|$ is a power of 2. Achieving this over $\{0,1\}$ is a very interesting open question. In Lemma 3, we show how to rearrange the paths of the branching program so that the combinatorial argument in Section 3.1 can still be used. Thus, we can express any branching program as a short sum (the size of the summation is substantially larger than in the oblivious case) over pairs of functions that fulfill the conditions of Lemma 1.

In fact, the decompositions we obtain for (oblivious) branching programs are implicit in work of Beame, et al. [BJS01]. That work gives similar decompositions for branching programs in order to prove lower bounds using communication complexity arguments. Accordingly, they decompose a branching program as a disjunction of function pairs, and the conditions on the function pairs are stronger. Our application requires the summation instead of the disjunction;

however, the proofs are essentially the same, and we include them here for completeness and simplification. We remark that further decompositions that yielded stronger lower bounds were given in subsequent work [Ajt99,BSSV03], but, to our knowledge, these are not relevant to the constructions here.

Fooling Stack Machines. We show that for every constant λ, a log-space stack machine over alphabet $\{0,1\}$ that makes $k(n)$ passes over its randomness can be simulated by a family of *nondeterministic* branching programs over alphabet $\{0,1\}^\lambda$ of size $2^{O((\log n)^2)}$ and length $nk(n)$. Moreover, the branching programs can be designed to have unique witnesses; namely, for every accepting input there is exactly one accepting computation path. We observe that our proof of Theorem 1 easily extends to nondeterministic branching programs with unique witnesses, and we conclude that our distribution P is pseudorandom for the corresponding stack machines. Due to space limitations, our reduction is given in the full version of this work.

A log-space stack machine computes a polynomial time predicate but it may run in time $2^{n^{O(1)}}$. In [DNPS11] it is shown that given a stack machine that makes $k(n)$ passes over its randomness, for a given input x, there is an advice string and a stack machine that computes the same predicate, runs in time $k(n)\cdot\mathsf{poly}(n)$, and preserves the number of passes over the random tape. Such stack machines can be simulated by small space computations [All89,BCD+89,Ruz80]. Niedermeier and Rossmanith [NR95] give a variant of this simulation that preserves the number of witnesses. However, these simulations fail to preserve the number of accesses to the input, even when the stack machines are equipped with an index tape to access the memory.

We show that with a non-trivial modification to [Ruz80], a randomized stack machine that makes $k(n)$ many passes can be simulated by a non-deterministic branching program with a unique witness that preserves the number of accesses to input bits (but not necessarily the order). More specifically, the branching program recursively verifies a kind of a "proof tree" that the computation accepts. For our purposes it is crucial to ensure that the random tape is not accessed more than $nk(n)$ times.

2 Fooling Pairs of Functions with Shared Inputs

In this section we give a distribution P over $\{0,1\}^n$ that looks pseudorandom to any pair of functions $f,g\colon \{0,1\}^n\to[-1,1]$ such that f and g depend on at most $(1-\Omega(1))n$ of their inputs. The construction is identical to the one from our previous work [BPW11], with different parameters. Note, however, that later on we will apply this theorem in two different ways, one of which regards distributions over alphabets other than $\{0,1\}$ (and this is essential for obtaining non-trivial stretch). In [BPW11] we proved that the desired pseudorandomness under the additional restriction that f and g each depend on $n/2$ bit inputs which are *disjoint*. We also remarked (without proof) that our analysis can be extended to the more general case, which is needed for the applications in this work. We now give a proof of that statement.

Our Pseudorandom Distribution. The pseudorandom distribution P has the form $\mathrm{P} = Mz + e$, where M is a fixed $n \times m$ for $m = (1-\rho) \cdot n$ matrix over $GF(2)$ such that every subspace spanned by $\alpha \cdot n$ rows has dimension $\alpha \cdot n - r$, and all operations are over $GF(2)$. Here $z \sim \{0,1\}^m$ is a uniformly random seed, and $e \in \{0,1\}^n$ is chosen independently of z from an ϵ-biased distribution. (Recall that e is ϵ-biased if for every $s \in \{0,1\}^n$, $s \neq 0$, $|\mathbf{E}_e[(-1)^{|\langle s,e\rangle|}]| \leq \epsilon$.)

The existence of an explicit matrix M with the desired properties follows from constructions of binary codes with small list size for list-decoding radius bounded away from $1/2$. We now explain this connection. Recall that a linear code C over $\{0,1\}^n$ is (δ, ℓ) *list-decodable* if for every $x \in \{0,1\}^n$, the number of codewords of C within hamming distance δn of x is at most ℓ. A *parity check matrix* M for C is a $GF(2)$ matrix such that $c^T M = 0$ if and only if c is a codeword of C.

It is easily seen (by substituting α for $1/2$) that the proof of Proposition 1 from [BPW11] yields the following more general statement:

Proposition 1. *Let C be a $(\frac{\alpha}{2}, \ell)$ list-decodable code over $\{0,1\}^n$. Let M be the parity check matrix of C. Then every subset of $\alpha \cdot n$ rows of M spans a vector space over $GF(2)$ of dimension at least $\alpha \cdot n - \log_2(2\ell)$.*

Then we have the following fact, which follows from the Johnson bound and standard constructions of asymptotically good binary linear codes; see Theorems 3.1 and 7.1 from [Gur07].

Proposition 2. *For every $\alpha > 0$ there exists $\rho > 0$ and an explicit matrix M of size $n \times (1-\rho)n$ such that every subset of $\alpha \cdot n$ rows spans a vector space over $GF(2)$ of dimension at least $\alpha \cdot n - r$, with $r = 4\log(4n/(1-\alpha))$.*

The Main Lemma Now, we prove the main lemma that powers our results in Section 3.

Lemma 1. *For every $\alpha > 0$ there exists $\rho > 0$ and an explicit matrix M of size $n \times (1-\rho) \cdot n$ so that for every pair of (possibly intersecting) ordered sets I, J with $|I|, |J| \leq \alpha n$ and for every pair of functions $f : \{0,1\}^{|I|} \to [-1,1], g : \{0,1\}^{|J|} \to [-1,1]$,*

$$\left|\mathbf{E}_{\mathrm{P}}[f(\mathrm{P}|_I) g(\mathrm{P}|_J)] - \mathbf{E}_{\mathrm{U}}[f(\mathrm{U}|_I) g(\mathrm{U}|_J)]\right| \leq 2^r \epsilon$$

where U is the uniform distribution over $\{0,1\}^n$, P is defined as above, and $x|_I, x|_J$ denote the projections of x on the sets I and J, respectively, and $r = 4 \cdot \log \frac{4n}{1-\alpha}$.

In particular, when $g = 1$, $|\mathbf{E}_{\mathrm{P}}[f(\mathrm{P}|_I)] - \mathbf{E}_{\mathrm{U}}[f(\mathrm{U}|_I)]| \leq 2^r \epsilon$, so the pseudorandom distribution also preserves the marginal probabilities of events, within $2^r \epsilon$, over all subsets of size at most $\alpha \cdot n$.

Proof (Proof of Lemma 7). Using Fourier decomposition, for any pair of subsets I, J of $[n]$ with $|I|, |J| \leq \alpha n$, we have

$$\begin{aligned}\mathbf{E}_{\mathrm{P}}[f(\mathrm{P}|_I)g(\mathrm{P}|_J)] &= \mathbf{E}_{z,e}[f((Mz+e)|_I)g((Mz+e)|_J)] \\ &= \sum_{S\subseteq I, T\subseteq J} \hat{f}(S)\hat{g}(T)\mathbf{E}_{z,e}[\chi_S(Mz|_I)\chi_S(e|_I)\chi_T(Mz|_J)\chi_T(e|_J)] \end{aligned} \tag{1}$$

We may view subsets $S \subseteq I$ and $T \subseteq J$ as subsets of $[n]$, so we write $\chi_S(Mz|_I) = \chi_S(Mz)$ and $\chi_T(Mz|_J) = \chi_T(Mz)$, and (1) becomes:

$$\sum_{S\subseteq I, T\subseteq J} \hat{f}(S)\hat{g}(T)\mathbf{E}_z[\chi_S(Mz)\chi_T(Mz)]\mathbf{E}_e[\chi_S(e)\chi_T(e)].$$

We denote by $S \Delta T$ the symmetric difference of S and T viewed as subsets of $[n]$.

We have that $\mathbf{E}_{\mathrm{U}}[f(\mathrm{U}|_I)g(\mathrm{U}|_J)] = \sum_{S\subseteq I\cap J} \hat{f}(S)\hat{g}(S)$ and $|\mathbf{E}_e[\chi_{S\Delta T}(e)]| \leq \epsilon$, therefore

$$\begin{aligned}&\left|\mathbf{E}_{\mathrm{P}}[f(\mathrm{P}|_I)g(\mathrm{P}|_J)] - \mathbf{E}_{\mathrm{U}}[f(\mathrm{U}|_I)g(\mathrm{U}|_J)]\right| \\ &\qquad = \left|\sum_{\substack{S\subseteq I, T\subseteq J \\ S\Delta T\neq\emptyset}} \hat{f}(S)\hat{g}(T)\mathbf{E}_z[\chi_{S\Delta T}(Mz)]\mathbf{E}_e[\chi_{S\Delta T}(e)]\right| \\ &\qquad \leq \sum_{\substack{S\subseteq I, T\subseteq J \\ S\Delta T\neq\emptyset}} \epsilon\cdot|\hat{f}(S)||\hat{g}(T)|\left|\mathbf{E}_z[\chi_{S\Delta T}(Mz)]\right|.\end{aligned}$$

Let G be a bipartite graph over vertices (subsets of I) $\cup$ (subsets of J), with an edge (S,T) present whenever $\mathbf{E}_z[\chi_{S\Delta T}(Mz)] \neq 0$. We will shortly argue that G has maximum degree 2^r. Assuming this, we can upper bound the last expression by

$$\begin{aligned}\epsilon\cdot\sum_{\text{edge }(S,T)}|\hat{f}(S)||\hat{g}(T)| &\leq \epsilon\cdot\sqrt{\sum_{\text{edge }(S,T)}\hat{f}(S)^2}\sqrt{\sum_{\text{edge }(S,T)}\hat{g}(T)^2} \\ &\leq \epsilon\cdot\sqrt{2^r\cdot\sum_{S\subseteq I}\hat{f}(S)^2}\sqrt{2^r\cdot\sum_{T\subseteq J}\hat{g}(T)^2} \leq \epsilon\cdot 2^r,\end{aligned}$$

where the first inequality follows from the Cauchy-Schwarz inequality, the second from the fact that G has maximum degree 2^r, and the third from Parseval's identity.

It remains to argue that G has maximum degree 2^r. We let $s \in \{0,1\}^n, t \in \{0,1\}^n$ be indicator vectors for the sets S and T, respectively, and s and t as vectors in $GF(2)^n$. Then

$$\mathbf{E}_z[\chi_{S\Delta T}(Mz)] = \mathbf{E}[(-1)^{(s+t)^T Mz}] = \begin{cases}1, & \text{if } (s+t)^T M = 0 \\ 0, & \text{otherwise.}\end{cases}$$

Now, $(s+t)^T M = 0$ if and only if $s^T M = t^T M$, where $s^T M$ is zero at least everywhere outside I and similarly for $t^T M$ and J. Since (by assumption) the matrix $M|_I$ has rank at least $\alpha n - r$, for every $t \in \{0,1\}^n$, there can be at most 2^r distinct vectors $s \in \{0,1\}^n$ such that $s^T M = t^T M$. Similarly, for every $s \in \{0,1\}^n$, there can be at most 2^r vectors $t \in \{0,1\}^n$ such that $s^T M = t^T M$.

In Section 3.2 we will apply the pseudorandom generator to strings of length n over alphabet $\Sigma = \{0,1\}^\lambda$. These can be viewed as strings in $\{0,1\}^{\lambda n}$ in the natural way.

3 Fooling Branching Programs of Linear Length

We show that essentially the same generator (modulo the setting of the parameters and the different input alphabets) fools branching programs of linear length. For oblivious branching programs we obtain this for binary $\{0,1\}$ input alphabets (Section 3.1), whereas for arbitrary branching programs we show this for branching programs over (larger) constant size alphabets (Section 3.2).

Let F be a width w, length kn, layered branching program over n inputs; we think of k as an arbitrarily large but fixed constant as n increases. We view the computation of the branching program on an input x as occurring in t stages, where each stage reads kn/t variables. Suppose first that the branching program is oblivious. Then, for every input each stage reads the *same* kn/t variables. In this case, we may write the branching program as a sum over w^t many t-tuples of Boolean functions (as was done in [BPW11] for $k = 1$ and $t = 2$).

More formally, divide the inputs into t sets of layers so that S_1 consists of inputs $\{1, \cdots, kn/t\}$, S_2 of inputs $\{kn/t + 1, \cdots, 2kn/t\}$, etc. (if variables reoccur within a set, its size might be smaller). We define functions $f_{i,p,q}(x|_{S_i}) : \{0,1\}^{|S_i|} \to \{0,1\}$ to be indicator functions for the event that the program moves from state p to q when the inputs in S_i are read from x. By definition, we have

$$F(x) = \sum_{\substack{p_1,\ldots,p_t: \\ p_t \in \text{accept}}} f_{1,s,p_1}(x|_{S_1}) f_{2,p_1,p_2}(x|_{S_2}) \cdots f_{t,p_{t-1},p_t}(x|_{S_t}) \tag{2}$$

3.1 Pseudorandomness for Oblivious Branching Programs

We will argue that each of the summands in (2) can be rewritten in terms of two functions, each over at most αn bits. Then, we apply Lemma 1 to show that the output of the generator fools each of these summands. This will give us the following theorem for oblivious branching programs.

Theorem 2. *Let $F : \{0,1\}^n \to \{1,-1\}$ be computable by a width w, length kn oblivious branching program on n inputs. Let P be the pseudorandom distribution. Then*

$$|\mathbf{E}_{\mathrm{P}}[F(\mathrm{P})] - \mathbf{E}_U[F(U)]| \leq w^t \cdot 2^r \epsilon.$$

where $t = 2^{4k}$, $r = 4 \log \frac{4n}{1-\alpha}$, and $\alpha > 1 - \frac{1}{2^{2k}}$.

The proof of Theorem 2 will use the following combinatorial lemma, which shows that we can always find a way to color each stage by one of two colors, so that neither color will contain too many variables. A slightly different version of this lemma was proven in [BJS01]; in the full version, we include a proof and argue that the parameter α below is close to optimal.

Lemma 2. *Fix any $k \in \mathbb{Z}^+$. Let $\{S_1, \cdots, S_t\}$ be a collection of subsets over $[n]$, each of size at most kn/t. Then there exists a partition $(\mathcal{C}, \overline{\mathcal{C}})$ of $\{1, \cdots, t\}$ satisfying:*

$$\left|\bigcup_{i \in \mathcal{C}} S_i\right| \leq \alpha \cdot n \quad \text{and} \quad \left|\bigcup_{i \in \overline{\mathcal{C}}} S_i\right| \leq \alpha \cdot n$$

where $\alpha \geq 1 - \frac{1}{2^k} + \frac{2k}{\sqrt{t}} + \frac{2}{\sqrt{n}}$.

Proof (Proof of Theorem 2). Now, consider the expected bias of the branching program using Equation 2; by linearity of expectation and the triangle inequality, we have:

$$|\mathbf{E}[F(U)] - \mathbf{E}[F(\mathrm{P})]| \leq \sum_{\substack{p_1, \ldots, p_t: \\ p_t \in \text{accept}}} |\mathbf{E}[f_{1,s,p_1}(\mathrm{P}|_{S_1}) \cdots f_{t,p_{t-1},p_t}(\mathrm{P}|_{S_t})] - \mathbf{E}[f_{1,s,p_1}(U|_{S_1}) \cdots f_{t,p_{t-1},p_t}(U|_{S_t})]|. \tag{3}$$

For each expectation of the summation, we can apply Lemma 2 to rewrite each product as a product of two functions, i.e.,

$$f_{1,s,p_1}(x|_{S_1}) \cdots f_{t,p_{t-1},a}(x_{S_t}) = g_1(x|_{\mathcal{S}}) g_2(x|_{\overline{\mathcal{S}}}),$$

where both $\mathcal{S} := \left|\bigcup_{i \in \mathcal{C}} S_i\right| \leq \alpha \cdot n$ and $\overline{\mathcal{S}}$ contain at most $\alpha \cdot n$ variables.

Setting $t = 2^{4k}$ in Lemma 2 and applying Lemma 1 with α from Lemma 2, we bound the magnitude of each difference by $2^r \epsilon$. Since there are w^t terms, we obtain

$$|\mathbf{E}[F(U)] - \mathbf{E}[F(\mathrm{P})]| \leq w^t 2^r \cdot \epsilon. \tag{4}$$

3.2 Arbitrary Linear Size Branching Programs over Large Alphabets

We show how to fool arbitrary branching programs with inputs over alphabet $\Sigma = \{0,1\}^\lambda$, where λ is a sufficiently large constant which depends on the multiplicative constant k in the length of the branching program.

Lemma 3. *Let $P(x) = P_1(x) \wedge \ldots \wedge P_t(x)$, where $P_1, \cdots, P_t \colon \Sigma^n \to \{0,1\}$ are branching programs of length at most kn/t each. Then, there exist collections of boolean functions $\{F_{\mathcal{C},U}\}$ and $\{G_{\mathcal{C},V}\}$, where $\mathcal{C}$ ranges over all partitions of $\{1, \ldots, t\}$ and U, V range over all subsets of $[n]$ of size αn such that*

$$P(x) = \sum_{\substack{\mathcal{C} \subseteq [t], U, V \subseteq [n] \\ |U| = |V| = \alpha n}} F_{\mathcal{C},U}(x) \cdot G_{\mathcal{C},V}(x) \tag{5}$$

and $\alpha \geq 1 - \frac{1}{2^k} + \frac{2k}{\sqrt{t}} + \frac{2}{\sqrt{n}}$.

Proof. We can express every P_i as $P_i(x) = \sum_{\ell_i \in \mathcal{L}_i} f_{i,\ell_i}(x)$ where the summation ranges over $\mathcal{L}_i$ which denotes all accepting paths ℓ_i of P_i and $f_{i,\ell_i}(x)$ is the indicator function for the event that the computation of P_i on input x takes path ℓ_i. We can write

$$P(x) = \prod_{i=1}^{t} P_i(x) = \prod_{i=1}^{t} \sum_{\ell_i \in \mathcal{L}_i} f_{i,\ell_i}(x) = \sum_{(\ell_1,\ldots,\ell_t) \in \mathcal{L}_1 \times \cdots \times \mathcal{L}_t} f_{1,\ell_1}(x) \cdot \ldots \cdot f_{t,\ell_t}(x).$$

By Lemma 2, for every collection $\ell = (\ell_1, \ldots, \ell_t)$ there exists a partition $\mathcal{C}(\ell)$ of $[t]$ and sets $U(\ell)$ and $V(\ell)$, each of size at most αn, such that when $i \in \mathcal{C}$, $f_{i,\ell_i}(x)$ depends only on inputs in $U(\ell)$ and when $i \in \overline{\mathcal{C}}$, $f_{i,\ell_i}(x)$ depends only on inputs in $V(\ell)$. Without loss of generality we will assume that the sizes of $U(\ell)$ and $V(\ell)$ are exactly αn. We set

$$F_{\mathcal{C},U}(x) = \bigvee_{\substack{\ell:\mathcal{C}(\ell)=\mathcal{C} \\ U(\ell)=U}} \bigwedge_{i \in \mathcal{C}} f_{i,\ell_i}(x) \qquad \text{and} \qquad G_{\mathcal{C},V}(x) = \bigvee_{\substack{\ell:\mathcal{C}(\ell)=\mathcal{C} \\ V(\ell)=V}} \bigwedge_{i \in \overline{\mathcal{C}}} f_{i,\ell_i}(x)$$

We now prove the identity (5). If $P(x) = 1$, then there is a unique path $\ell = (\ell_1, \ldots, \ell_t)$ such that $f_{i,\ell_i}(x) = 1$ for all i, and so $F_{\mathcal{C},U}(x)$ and $G_{\mathcal{C},V}(x)$ both take value 1 when and only when $\mathcal{C} = \mathcal{C}(\ell)$, $U = U(\ell)$, and $V = V(\ell)$. Then exactly one term on the right hand side of (5) evaluates to 1.

If $P(x) = 0$, then $P_i(x) = 0$ for some i, so $f_{i,\ell_i}(x) = 0$ for all accepting paths ℓ_i of P_i. This forces $F_{\mathcal{C},U}(x)$ to equal zero when $i \in \mathcal{C}$, and $G_{\mathcal{C},V}(x) = 0$ when $i \in \overline{\mathcal{C}}$. So all terms on the right hand side of (5) evaluate to 0.

To prove Theorem 3 below, we will use Lemma 3 to write the branching program as a sum of a limited number of pairs of functions, where each pair satisfies the desired property. We then use Lemma 1 to bound the deviation of each term in this summation when the uniform distribution is replaced by the pseudorandom one.

Theorem 3. *Let $k > 0$ be a constant, and fix an alphabet size $\lambda \geq 2$. Let $F : \Sigma^n \to \{0,1\}$ be computable by a branching program on n inputs of width w and length kn. Then*

$$|\mathbf{E}_{\mathrm{P}}[F(\mathrm{P})] - \mathbf{E}_{\mathrm{U}}[F(\mathrm{U})]| \leq \left(\frac{4\lambda n}{1-\alpha}\right)^4 \cdot w^{2^{4k}} \cdot 2^{2H(\alpha)n} \cdot \epsilon,$$

where P *is the pseudorandom distribution over $\{0,1\}^{\lambda n}$, and $\alpha \geq 1 - \frac{1}{2^{2k}}$.*

Proof. Applying the decomposition (2) we write

$$\begin{aligned} F(x) &= \sum_{\substack{p_1,\ldots,p_t: \\ p_t \in \text{accept}}} f_{1,s,p_1}(x|_{S_1}) f_{2,p_1,p_2}(x|_{S_2}) \cdots f_{t,p_{t-1},p_t}(x|_{S_t}) \\ &= \sum_{\substack{p_1,\ldots,p_t: \\ p_t \in \text{accept}}} F_{p_1,\ldots,p_t}(x). \end{aligned}$$

Here, $f_{i,p,q}$ are all branching programs of length kn/t. By Lemma 3 we have

$$F(x) = \sum_{\substack{p_1,\ldots,p_t:\\ p_t \in \text{accept}}} \sum_{\mathcal{C},U,V} F_{p_1,\ldots,p_t,\mathcal{C},U}(x) \cdot G_{p_1,\ldots,p_t,\mathcal{C},V}(x). \tag{6}$$

where $\mathcal{C}$ ranges over all partitions of $[t]$, U, V range over all subsets of $[n]$ of size αn, and $F_{p_1,\ldots,p_t,\mathcal{C},U} : \Sigma^{\alpha n} \to \{0,1\}$ and $G_{p_1,\ldots,p_t,\mathcal{C},V} : \Sigma^{\alpha n} \to \{0,1\}$ depend only on inputs coming from U and V respectively. Now, let us view $F_{p_1,\ldots,p_t,\mathcal{C},U}, G_{p_1,\ldots,p_t,\mathcal{C},V}$ as functions with domain $\{0,1\}^{\alpha\lambda n}$. Set $t = 2^{4k}$ and $r = 4\log(4\lambda n/(1-\alpha))$. By Lemma 1 for each term in the sum, the difference in expectations under the uniform and pseudorandom distributions is at most $\epsilon 2^r$ in absolute value. Since there are at most w^t choices for $(p_1, \ldots, p_t)$, 2^t choices for $\mathcal{C}$, and $\binom{n}{\alpha n}$ choices for each of U and V, by the triangle inequality we obtain that

$$|\mathbf{E}_{\mathrm{P}}[F(\mathrm{P})] - \mathbf{E}_{\mathrm{U}}[F(\mathrm{U})]| \leq \epsilon 2^r \cdot w^t \cdot 2^t \binom{n}{\alpha n}^2,$$

which yields the desired bound after substituting the values for r and t and the standard bound for binomial coefficients.

Parameters. We now set the parameters to obtain Theorem 1. We assume the availability of a family small-biased generators over $\{0,1\}^m$ for bias ϵ and seed length $\log(m/\epsilon)^K$ for some constant K constructible in time polynomial in the seed length (see e.g. [AGHP90] for a construction with $K = 2$). We instantiate this construction with parameters $m = \lambda n$ and $\epsilon = 2^{-4n}$ to obtain a seed length of $4Kn + o(n)$. Set $\alpha = 1 - 2^{-2k}$. By Lemma 1, there exists a constant 2ρ (depending on α) for which the distribution P can be generated efficiently with seed length $(1-2\rho)\lambda n + 4Kn + o(n)$. Setting $\lambda = 5K/\rho$, the seed length is upper bounded by $(1-\rho)\lambda n$ bits, i.e. $(1-\rho)n$ elements of Σ, when n is sufficiently large. To calculate the bias, we simplify the upper bound in Theorem 3 to $4\lambda^4 n^4 w^{2^{4k}} \cdot 2^{2n} \cdot \epsilon$. When $w \leq 2^{n/2^{4k}}$, this expression is upper bounded by $4\lambda^4 n^4 \cdot 2^{-n} = 2^{-\Omega(n)}$.

Acknowledgements. We are grateful to the anonymous referee that pointed out a significant flaw in the proof of a previous version of our main theorem.

References

[AB84] Ajtai, M., Ben-Or, M.: A theorem on probabilistic constant depth computations. In: Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing, STOC 1984, pp. 471–474. ACM, New York (1984)

[AGHP90] Alon, N., Goldreich, O., Håstad, J., Peralta, R.: Simple constructions of almost k-wise independent random variables. In: Proceedings of the 31st Annual Symposium on Foundations of Computer Science, pp. 544–553 (1990)

[Ajt99] Ajtai, M.: A non-linear time lower bound for boolean branching programs. In: 40th Annual Symposium on Foundations of Computer Science, pp. 60–70. IEEE (1999)

[All89] Allender, E.W.: P-uniform circuit complexity. Journal of the ACM 36(4), 912–928 (1989)

[AW85] Ajtai, M., Wigderson, A.: Deterministic simulation of probabilistic constant depth circuits. In: 26th Annual Symposium on Foundations of Computer Science, pp. 11–19. IEEE (1985)

[BCD+89] Borodin, A., Cook, S.A., Dymond, P.W., Ruzzo, W.L., Tompa, M.: Two applications of inductive counting for complementation problems. SIAM J. Comput 18(3), 559–578 (1989)

[BJS01] Beame, P., Jayram, T.S., Saks, M.: Time-space tradeoffs for branching programs. Journal of Computer and System Sciences 63(4), 542–572 (2001)

[BPW11] Bogdanov, A., Papakonstantinou, P.A., Wan, A.: Pseudorandomness for read-once formulas. In: Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science, FOCS 2011 (2011)

[Bra10] Braverman, M.: Polylogarithmic independence fools AC^0 circuits. Journal of the ACM (JACM) 57(5), 1–10 (2010)

[BSSV03] Beame, P., Saks, M., Sun, X., Vee, E.: Time-space trade-off lower bounds for randomized computation of decision problems. Journal of the ACM (JACM) 50(2), 154–195 (2003)

[Coo71] Cook, S.A.: Characterizations of pushdown machines in terms of time-bounded computers. Journal of ACM (JACM) 18(1), 4–18 (1971)

[DNPS11] David, M., Nguyen, P., Papakonstantinou, P.A., Sidiropoulos, A.: Computationally limited randomness. In: Chazelle, B. (ed.) Proceedings of Innovations in Computer Science - ICS 2010, January 7-9, pp. 522–536. Tsinghua University Press, Beijing (2011)

[DP10] David, M., Papakonstantinou, P.A.: Trade-off lower bounds for stack machines. In: IEEE Conference on Computational Complexity (CCC), Boston, USA, pp. 163–171 (2010)

[Gur07] Guruswami, V.: Algorithmic results in list decoding. Foundations and Trends® in Theoretical Computer Science 2(2), 107–195 (2007)

[INW94] Impagliazzo, R., Nisan, N., Wigderson, A.: Pseudorandomness for network algorithms. In: Proceedings of the 26th Annual ACM Symposium on Theory of Computing, STOC 1994, Montréal, Québec, Canada, May 23-25, pp. 356–364. ACM Press, New York (1994)

[KI04] Kabanets, V., Impagliazzo, R.: Derandomizing polynomial identity tests means proving circuit lower bounds. Computational Complexity 13(1-2), 1–46 (2004)

[Nis91] Nisan, N.: Pseudorandom bits for constant depth circuits. Combinatorica 11(1), 63–70 (1991)

[Nis92] Nisan, N.: Pseudorandom generators for space-bounded computation. Combinatorica 12(4), 449–461 (1992)

[NR95] Niedermeier, R., Rossmanith, P.: Unambiguous auxiliary pushdown automata and semi-unbounded fan-in circuits. Information and Computation 118(2), 227–245 (1995)

[Ruz80] Ruzzo, W.L.: Tree-size bounded alternation. Journal of Computer Systems and Sciences (JCSS) 21(2), 218–235 (1980)

A Discrepancy Lower Bound for Information Complexity

Mark Braverman[1,2,⋆] and Omri Weinstein[2]

[1] University of Toronto
mbraverm@cs.princeton.edu
[2] Princeton University
oweinste@cs.princeton.edu

Abstract. This paper provides the first general technique for proving information lower bounds on two-party unbounded-rounds communication problems. We show that the discrepancy lower bound, which applies to randomized communication complexity, also applies to information complexity. More precisely, if the discrepancy of a two-party function f with respect to a distribution μ is $Disc_\mu f$, then any two party randomized protocol computing f must reveal at least $\Omega(\log(1/Disc_\mu f))$ bits of information to the participants. As a corollary, we obtain that any two-party protocol for computing a random function on $\{0,1\}^n \times \{0,1\}^n$ must reveal $\Omega(n)$ bits of information to the participants.

In addition, we prove that the discrepancy of the Greater-Than function is $\Omega(1/\sqrt{n})$, which provides an alternative proof to the recent proof of Viola [Vio11] of the $\Omega(\log n)$ lower bound on the communication complexity of this well-studied function and, combined with our main result, proves the tight $\Omega(\log n)$ lower bound on its information complexity.

The proof of our main result develops a new simulation procedure that may be of an independent interest. In a very recent breakthrough work of Kerenidis et al. [KLL+12], this simulation procedure was a building block towards a proof that almost all known lower bound techniques for communication complexity (and not just discrepancy) apply to information complexity.

1 Introduction

The main objective of this paper is to expand the available techniques for proving information complexity lower bounds for communication problems. Let $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ be a function, and μ be a distribution on $\mathcal{X} \times \mathcal{Y}$. Informally, the information complexity of f is the least amount of *information* that Alice and Bob need to exchange on average to compute $f(x,y)$ using a randomized communication protocol if initially x is given to Alice, y is given to Bob, and $(x,y) \sim \mu$. Note that information here is measured in the Shannon sense, and the amount of information may be much smaller than the number of bits exchanged. Thus the randomized communication complexity of f is an upper bound on its information complexity, but may not be a lower bound.

⋆ Partially supported by an NSERC Discovery Grant, an Alfred P. Sloan Fellowship, and an NSF CAREER award.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 459–470, 2012.
© Springer-Verlag Berlin Heidelberg 2012

Within the context of communication complexity, information complexity has first been introduced in the context of direct sum theorems for randomized communication complexity [CSWY01, BYJKS04, BBCR10]. These techniques are also being used in the related direction of direct product theorems [KSDW04, LSS08, Jai10, Kla10]. A direct sum theorem in a computational model states that the amount of resources needed to perform k independent tasks is roughly k times the amount of resources c needed for computing a single task. A direct product theorem, which is a stronger statement, asserts that any attempt to solve k independent tasks using $o(kc)$ resources would result in an exponentially small success probability $2^{-\Omega(k)}$.

The direct sum line of work [HJMR07, JSR08, BBCR10, BR11] has eventually led to a tight connection (equality) between amortized communication complexity and information complexity. Thus proving lower bounds on the communication complexity of k copies of f for a growing k is equivalent to proving lower bounds on the information complexity of f. In particular if f satisfies $IC(f) = \Omega(CC(f))$, i.e. that its information cost is asymptotically equal to its communication complexity, then a strong direct sum theorem holds for f. In addition to the intrinsic interest of understanding the amount of information exchange that needs to be involved in computing f, direct sum theorems motivate the development of techniques for proving lower bounds on the information complexity of functions.

Another important motivation for seeking lower bounds on the information complexity of functions stems from understanding the limits of security in two-party computation. In a celebrated result Ben-Or et al. [BOGW88] (see also [AL11]) showed how a multi-party computation (with three or more parties) may be carried out in a way that reveals no information to the participants except for the computation's output. The protocol relies heavily on the use of random bits that are shared between some, but not all, parties. Such a resource can clearly not exist in the two-party setting. While it can be shown that perfect information security is unattainable by two-party protocols [CK89, BYCKO93], quantitatively it is not clear just how much information the parties must "leak" to each other to compute f. The quantitative answer depends on the model in which the leakage occurs, and whether quantum computation is allowed [Kla04]. Information complexity answers this question in the strongest possible sense for classical protocols: the parties are allowed to use private randomness to help them "hide" their information, and the information revealed is measured on average. Thus an information complexity lower bound of I on a problem implies that the *average* (as opposed to worst-case) amount of information revealed to the parties is at least I.

As mentioned above, the information complexity is always upper bounded by the communication complexity of f. The converse is not known to be true. Moreover, lower bound techniques for communication complexity do not readily translate into lower bound techniques for information complexity. The key difference is that a low-information protocol is not limited in the amount of communication it uses, and thus rectangle-based communication bounds do not

readily convert into information lower bounds. No general technique has been known to yield sharp information complexity lower bounds. A linear lower bound on the communication complexity of the disjointness function has been shown in [Raz92]. An information-theoretic proof of this lower bound [BYJKS04] can be adapted to prove a linear *information* lower bound on disjointness [Bra11]. One general technique for obtaining (weak) information complexity lower bounds was introduced in [Bra11], where it has been shown that any function that has I bits of information complexity, has communication complexity bounded by $2^{O(I)}$. This immediately implies that the information complexity of a function f is at least the log of its communication complexity ($IC(f) \geq \Omega(\log(CC(f)))$). In fact, this result easily follows from the stronger result we prove below (Theorem 2).

1.1 Our Results

In this paper we prove that the discrepancy method – a general communication complexity lower bound technique – generalizes to information complexity. The discrepancy of f with respect to a distribution μ on inputs, denoted $Disc_\mu(f)$, measures how "unbalanced" f can get on any rectangle, where the balancedness is measured with respect to μ:

$$Disc_\mu(f) \stackrel{def}{=} \max_{\text{rectangles } R} \left| \Pr_\mu[f(x,y) = 0 \wedge (x,y) \in R] - \Pr_\mu[f(x,y) = 1 \wedge (x,y) \in R] \right|.$$

A well-known lower bound (see e.g [KN97]) asserts that the distributional communication complexity of f, denoted $D^\mu_{1/2-\epsilon}(f)$, when required to predict f with advantage ε over a random guess (with respect to μ), is bounded from below by $\Omega(\log 1/Disc_\mu(f))$:

$$D^\mu_{1/2-\epsilon}(f) \geq \log(2\epsilon/Disc_\mu(f)).$$

Note that the lower bound holds even if we are merely trying to get an advantage of $\varepsilon = \sqrt{Disc_\mu(f)}$ over random guessing in computing f. We prove that the information complexity of computing f with probability $9/10$ with respect to μ is also bounded from below by $\Omega(\log(1/Disc_\mu(f)))$.

Theorem 1. *Let $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$ be a Boolean function and let μ be any probability distribution on $\mathcal{X} \times \mathcal{Y}$. Then*

$$\mathsf{IC}_\mu(f, 1/10) \geq \Omega(\log(1/Disc_\mu(f))).$$

Remark 1. The choice of $9/10$ is somewhat arbitrary. For randomized worst-case protocols, we may replace the success probability with $1/2 + \delta$ for a constant δ, since repeating the protocol constantly many times ($1/\delta^2$) would yield the aforementioned success rate, while the information cost of the repeated protocol differs only by a constant factor from the original one. In particular, using prior-free information cost [Bra11] this implies $\mathsf{IC}(f, 1/2 - \delta) \geq \Omega(\delta^2 \log(1/Disc_\mu(f)))$.

In particular, Theorem 1 implies a linear lower bound on the information complexity of the inner product function $IP(x,y) = \sum_{i=1}^n x_i y_i \mod 2$, and on a random boolean function $f_r : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, expanding the (limited) list of functions for which nontrivial information-complexity lower bounds are known:

Corollary 1. *The information complexity* $\mathsf{IC}_{uniform}(IP, 1/10)$ *of* $IP(x, y)$ *is* $\Omega(n)$*. The information complexity* $\mathsf{IC}_{uniform}(f_r, 1/10)$ *of a random function* f_r *is* $\Omega(n)$*, except with probability* $2^{-\Omega(n)}$*.*

We study the communication and information complexity of the Greater-Than function (GT_n) on numbers of length n. This is a very well-studied problem [Smi88, MNSW95, KN97]. Only very recently the tight lower bound of $\Omega(\log n)$ in the public-coins probabilistic model was given by Viola [Vio11]. We show that the discrepancy of the GT_n function is $\Omega(1/\sqrt{n})$:

Lemma 1. *There exist a distribution* μ_n *on* $\mathcal{X} \times \mathcal{Y}$ *such that the discrepancy of* GT_n *with respect to* μ_n *satisfies* $Disc_{\mu_n}(GT_n) < 20/\sqrt{n}$*.*

For the proof we refer the reader to the full version of this paper. Lemma 1 provides an alternative (arguably simpler) proof of Viola's [Vio11] lower bound on the *communication complexity* of GT_n. By Theorem 1, Lemma 1 immediately implies a lower bound on the *information complexity* of GT_n:

Corollary 2. $\mathsf{IC}_{\mu_n}(GT_n, 1/10) = \Omega(\log n)$

This settles the information complexity of the GT function, since this problem can be solved by a randomized protocol with $O(\log n)$ communication (see [KN97]). This lower bound is particularly interesting since it demonstrates the first tight information-complexity lower bound for a natural function that is not linear.

The key technical idea in the proof of Theorem 1 is a new simulation procedure that allows us to convert any protocol that has information cost I into a (two-round) protocol that has communication complexity $O(I)$ and succeeds with probability $> 1/2 + 2^{-O(I)}$, yielding a $2^{-O(I)}$ advantage over random guessing. Combined with the discrepancy lower bound for communication complexity, this proves Theorem 1.

1.2 Comparison and Connections to Prior Results

The most relevant prior work is an article by Lee, Shraibman, and Špalek [LSS08]. Improving on an earlier work of Shaltiel [Sha03], Lee et al. show a direct product theorem for discrepancy, proving that the discrepancy of $f^{\otimes k}$ — the k-wise XOR of a function f with itself — behaves as $Disc(f)^{\Omega(k)}$. This implies in particular that the communication complexity of $f^{\otimes k}$ scales at least as $\Omega(k \cdot \log Disc(f))$. Using the fact that the limit as $k \to \infty$ of the amortized communication complexity of f is equal to the information cost of f [BR10], the result of Lee et al. "almost" implies the bound of Theorem 1. Unfortunately, the amortized communication complexity in the sense of [BR10] is the amortized cost of k copies of f, where *each* copy is allowed to err with some probability (say 1/10). Generally speaking, this task is much easier than computing the XOR (which requires *all* copies to be evaluated correctly with high probability). Thus the lower bound that follows from Lee et al. applies to a more difficult problem, and does not imply the information complexity lower bound.

Another generic approach one may try to take is to use compression results such as [BBCR10] to lower bound the information cost from communication complexity lower bounds. The logic of such a proof would go as follows: "Suppose there was a information-complexity-I protocol π for f, then if one can compress it into a low-communication protocol one may get a contradiction to the communication complexity lower bound f". Unfortunately, all known compression results compress π into a protocol π' whose communication complexity depends on I but also on $CC(\pi)$. Even for external information complexity (which is always greater than the internal information complexity, the bound obtained in [BBCR10] is of the form $I_{ext}(\pi) \cdot polylog(CC(\pi))$. Thus compression results of this type cannot rule out protocols that have low information complexity but a very high (e.g. exponential) communication complexity.

Our result can be viewed as a weak compression result for protocols, where a protocol for computing f that conveys I bits of information is converted into a protocol that uses $O(I)$ bits of *communication* and giving an advantage of $2^{-O(I)}$ in computing f. This strengthens the result in [Bra11] where a compression to $2^{O(I)}$ bits of communication has been shown. We still do not know whether compression to a protocol that uses $O(I)$ bits of communication and succeeds with high probability (as opposed to getting a small advantage over random) is possible.

In a very recent breakthrough work of Kerenidis, Laplante, Lerays, Roland, and Xiao [KLL+12], our main protocol played an important role in the proof that almost all known lower bound techniques for communication complexity (and not just discrepancy) apply to information complexity. The results of [KLL+12] shed more light on the information complexity of many communication problems, and the question of whether interactive communication can be compressed.

2 Preliminaries

In an effort to make this paper as self-contained as possible, we provide some background on information theory and communication complexity, which is essential to proving our results. For further details and a more thorough treatment of these subjects see [BR10] and references therein.

Notation. We reserve capital letters for random variables and distributions, calligraphic letters for sets, and small letters for elements of sets. Throughout this paper, we often use the notation $|b$ to denote conditioning on the event $B = b$. Thus $A|b$ is shorthand for $A|B = b$.

We use the standard notion of *statistical/total variation* distance between two distributions.

Definition 1. *Let D and F be two random variables taking values in a set $\mathcal{S}$. Their* statistical distance *is* $|D - F| \stackrel{def}{=} \max_{\mathcal{T} \subseteq \mathcal{S}}(|\Pr[D \in \mathcal{T}] - \Pr[F \in \mathcal{T}]|) = \frac{1}{2}\sum_{s \in \mathcal{S}} |\Pr[D = s] - \Pr[F = s]|$

2.1 Information Theory

Definition 2 (Entropy). *The* entropy *of a random variable X is $H(X) \stackrel{def}{=} \sum_x \Pr[X = x] \log(1/\Pr[X = x])$. The* conditional entropy $H(X|Y)$ *is defined as* $\mathbf{E}_{y \in_R Y}[H(X|Y = y)]$.

Definition 3 (Mutual Information). *The* mutual information *between two random variables A, B, denoted $I(A; B)$ is defined to be the quantity $H(A) - H(A|B) = H(B) - H(B|A)$. The* conditional mutual information $I(A; B|C)$ *is* $H(A|C) - H(A|BC)$.

We also use the notion of *divergence* (also known as Kullback-Leibler distance or relative entropy), which is a different way to measure the distance between two distributions:

Definition 4 (Divergence). *The informational divergence between two distributions is*

$$\mathbf{D}(A||B) \stackrel{def}{=} \sum_x A(x) \log(A(x)/B(x)).$$

Proposition 1. *Let A, B, C be random variables in the same probability space. For every a in the support of A and c in the support of C, let B_a denote $B|A = a$ and B_{ac} denote $B|A = a, C = c$. Then $I(A; B|C) = \mathbf{E}_{a,c \in_R A,C}[\mathbf{D}(B_{ac}||B_c)]$.*

2.2 Communication Complexity

We use the standard definitions of the computational model defined in [Yao79]. For complete details see the full version of this paper [BW11].

Given a communication protocol π, $\pi(x, y)$ denotes the concatenation of the public randomness with all the messages that are sent during the execution of π. We call this the *transcript* of the protocol. When referring to the random variable denoting the transcript, rather than a specific transcript, we will use the notation $\Pi(x, y)$ — or simply Π when x and y are clear from the context, thus $\pi(x, y) \in_R \Pi(x, y)$. When x and y are random variables themselves, we will denote the transcript by $\Pi(X, Y)$, or just Π.

Definition 5 (Communication Complexity notation). *For a function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{Z}_K$, a distribution μ supported on $\mathcal{X} \times \mathcal{Y}$, and a parameter $\epsilon > 0$, $D^{\mu}_{\epsilon}(f)$ denotes the communication complexity of the cheapest deterministic protocol computing f on inputs sampled according to μ with error ϵ.*

Definition 6 (Combinatorial Rectangle). *A Rectangle* . in $\mathcal{X} \times \mathcal{Y}$ is a subset $R \subseteq \mathcal{X} \times \mathcal{Y}$ which satisfies $(x_1, y_1) \in R$ and $(x_2, y_2) \in R \Longrightarrow (x_1, y_2) \in R$

2.3 Information + Communication: The Information Cost of a Protocol

The following quantity, which is implicit in [BYJKS04] and was explicitly defined in [BBCR10], is the central notion of this paper.

Definition 7. *The (internal) information cost* of a protocol π over inputs drawn from a distribution μ on $\mathcal{X} \times \mathcal{Y}$, is given by:

$$\mathsf{IC}_\mu(\pi) := I(\Pi; X|Y) + I(\Pi; Y|X).$$

Intuitively, Definition 7 captures how much the two parties learn about each other's inputs from the execution transcript of the protocol π. The first term captures what the second player learns about X from Π – the mutual information between the input X and the transcript Π given the input Y. Similarly, the second term captures what the first player learns about Y from Π.

Note that the information of a protocol π depends on the prior distribution μ, as the mutual information between the transcript Π and the inputs depends on the prior distribution on the inputs. To give an extreme example, if μ is a singleton distribution, i.e. one with $\mu(\{(x,y)\}) = 1$ for some $(x,y) \in \mathcal{X} \times \mathcal{Y}$, then $\mathsf{IC}_\mu(\pi) = 0$ for all possible π, as no protocol can reveal anything to the players about the inputs that they do not already know *a-priori*. Similarly, $\mathsf{IC}_\mu(\pi) = 0$ if $\mathcal{X} = \mathcal{Y}$ and μ is supported on the diagonal $\{(x,x) : x \in \mathcal{X}\}$. As expected, one can show that the communication cost $\mathsf{CC}(\pi)$ of π is an upper bound on its information cost over *any* distribution μ:

Lemma 2. *[BR10] For any distribution μ, $\mathsf{IC}_\mu(\pi) \leq \mathsf{CC}(\pi)$.*

On the other hand, as noted in the introduction, the converse need not hold. In fact, by [BR10], getting a strict inequality in Lemma 2 is equivalent to violating the direct sum conjecture for randomized communication complexity.

As one might expect, the information cost of a function f with respect to μ and error ρ is the least amount of information that needs to be revealed by a protocol computing f with error $\leq \rho$:

$$\mathsf{IC}_\mu(f, \rho) := \inf_{\pi:\ \mathbf{P}_\mu[\pi(x,y) \neq f(x,y)] \leq \rho} \mathsf{IC}_\mu(\pi).$$

The (prior-free) information cost was defined in [Bra11] as the minimum amount of information that a worst-case error-ρ randomized protocol can reveal against *all* possible prior distributions.

$$\mathsf{IC}\,(f, \rho) := \inf_{\pi \text{ is a protocol with } \mathbf{P}[\pi(x,y) \neq f(x,y)] \leq \rho \text{ for all } (x,y)} \max_{\mu} \mathsf{IC}_\mu(\pi).$$

This information cost matches the amortized *randomized* communication complexity of f [Bra11]. It is clear that lower bounds on $\mathsf{IC}_\mu(f, \rho)$ *for any distribution* μ also apply to $\mathsf{IC}\,(f, \rho)$.

3 Proof of Theorem 1

To establish the correctness of Theorem 1, we prove the following theorem, which is the central result of this paper:

Theorem 2. *Suppose that* $\mathsf{IC}_\mu(f, 1/10) = I_\mu$. *Then there exist a protocol* π' *such that*

- $\mathsf{CC}(\pi') = O(I_\mu)$.
- $\mathbf{P}_{(x,y)\sim\mu}[\pi'(x,y) = f(x,y)] \geq 1/2 + 2^{-O(I_\mu)}$

We first show how Theorem 1 follows from Theorem 2:

Proof of Theorem 1. Let f, μ be as in theorem 1, and let $\mathsf{IC}_\mu(f, 1/10) = I_\mu$. By theorem 2, there exists a protocol π' computing f with error probability $1/2 - 2^{-O(I_\mu)}$ using $O(I_\mu)$ bits of communication. Applying the discrepancy lower bound for communication complexity we obtain

$$O(I_\mu) \geq D^\mu_{1/2-2^{-O(I_\mu)}}(f) \geq \log(2 \cdot 2^{-O(I_\mu)}/Disc_\mu(f)) \tag{1}$$

which after rearranging gives $I_\mu \geq \Omega(\log(1/Disc_\mu(f)))$, as desired.

We turn to prove Theorem 2. The main step is the following sampling lemma.

Lemma 3. *Let* μ *be any distribution over a universe* $\mathcal{U}$ *and let* $I \geq 0$ *be a parameter that is known to both* A *and* B. *Further, let* ν_A *and* ν_B *be two distributions over* $\mathcal{U}$ *such that* $\mathbf{D}(\mu||\nu_A) \leq I$ *and* $\mathbf{D}(\mu||\nu_B) \leq I$. *The players are each given a pair of real functions* (p_A, q_A), (p_B, q_B), $p_A, q_A, p_B, q_B : \mathcal{U} \to [0,1]$ *such that for all* $x \in \mathcal{U}$, $\mu(x) = p_A(x) \cdot p_B(x)$, $\nu_A(x) = p_A(x) \cdot q_A(x)$, *and* $\nu_B(x) = p_B(x) \cdot q_B(x)$. *Then there is a (two round) sampling protocol* $\Pi_1 = \Pi_1(p_A, p_B, q_A, q_B, I)$ *which has the following properties:*

1. *at the end of the protocol, the players either declare that the protocol "fails", or output* $x_A \in \mathcal{U}$ *and* $x_B \in \mathcal{U}$, *respectively ("success").*
2. *let* $\mathcal{S}$ *be the event that the players output "success". Then* $\mathcal{S} \Rightarrow x_A = x_B$, *and* $0.9 \cdot 2^{-50(I+1)} \leq \Pr[\mathcal{S}] \leq 2^{-50(I+1)}$.
3. *if* μ_1 *is the distribution of* x_A *conditioned on* $\mathcal{S}$, *then* $|\mu - \mu_1| < 2/9$.

Furthermore, Π_1 *can be "compressed" to a protocol* Π_2 *such that* $\mathsf{CC}(\Pi_2) = 211I + 1$, *whereas* $|\Pi_1 - \Pi_2| \leq 2^{-59I}$ *(by an abuse of notation, here we identify* Π_i *with the random variable representing its output).*

We will use the following technical fact about the divergence of distributions.

Claim (3). [Claim 5.1 in [Bra11]] Suppose that $\mathbf{D}(\mu||\nu) \leq I$. Let ε be any parameter. Then $\mu\left\{x : 2^{(I+1)/\varepsilon} \cdot \nu(x) < \mu(x)\right\} < \varepsilon$.

For completeness, we repeat the proof in the full version of this paper [BW11].

*Proof (***Proof of Lemma 3** *).* Throughout the execution of Π_1, Alice and Bob interpret their shared random tape as a source of points (x_i, α_i, β_i) uniformly distributed in $\mathcal{U} \times [0, 2^{50(I+1)}] \times [0, 2^{50(I+1)}]$. Alice and Bob consider the first

$T = |\mathcal{U}| \cdot 2^{100(I+1)} \cdot 60I$ such points. Their goal will be to discover the first index τ such that $\alpha_\tau \leq p_A(x_\tau)$ and $\beta_\tau \leq p_B(x_\tau)$ (where they wish to find it using a minimal amount of communication, even if they are most likely to fail). First, we note that the probability that an index t satisfies $\alpha_t \leq p_A(x_t)$ and $\beta_t \leq p_B(x_t)$ is exactly $1/|\mathcal{U}|2^{50(I+1)}2^{50(I+1)} = 1/|\mathcal{U}|2^{100(I+1)}$. Hence the probability that $\tau > T$ (i.e. that x_τ is not among the T points considered) is bounded by

$$\left(1 - 1/|\mathcal{U}|2^{100(I+1)}\right)^T < e^{-T/|\mathcal{U}|2^{100(I+1)}} = e^{-60I} < 2^{-60I} \tag{2}$$

Denote by $\mathcal{A}$ the following set of indices $\mathcal{A} := \{i \leq T : \ \alpha_i \leq p_A(x_i) \text{ and } \beta_i \leq 2^{50(I+1)} \cdot q_A(x_i)\}$, the set of potential candidates for τ from A's viewpoint. Similarly, denote $\mathcal{B} := \{i \leq T : \ \alpha_i \leq 2^{50(I+1)} \cdot q_B(x_i) \text{ and } \beta_i \leq p_B(x_i)\}$.

The protocol Π_1 is very simple. Alice takes her bet on the first element $a \in \mathcal{A}$ and sends it to Bob. Bob outputs a only if (it just so happens that) $\beta_\tau \leq p_B(a)$.

We turn to analyze Π_1. Denote the set of "Good" elements by

$$\mathcal{G} \stackrel{def}{=} \{x : \ 2^{50(I+1)} \cdot \nu_A(x) \geq \mu(x) \ \text{ and } \ 2^{50(I+1)} \cdot \nu_B(x) \geq \mu(x)\}\}.$$

Then by Claim 3, $\mu(\mathcal{G}) \geq 48/50 = 24/25$. The following claim asserts that if it succeeds, the output of Π_1 has the "correct" distribution on elements in $\mathcal{G}$. Due to space constraints we defer the proof to the full version of this paper.

Proposition 2. *Assume $\mathcal{A}$ is nonempty. Then for any $x_i \in \mathcal{U}$, the probability that Π_1 outputs x_i is at most $\mu(x_i) \cdot 2^{-50(I+1)}$. If $x_i \in \mathcal{G}$, then this probability is* exactly $\mu(x_i) \cdot 2^{-50(I+1)}$.

The following proposition gives an estimate of the success probability of the protocol. We defer the proof to the full version of this paper.

Proposition 3. *Let $\mathcal{S}$ denote the event that $\mathcal{A} \neq 0$ and $a \in \mathcal{B}$ (i.e, that the protocol succeeds). Then* $0.9 \cdot 2^{-50(I+1)} \leq \Pr[\mathcal{S}] \leq 2^{-50(I+1)}$.

Finally, the following claim asserts that if $\mathcal{S}$ occurs, then the distribution of a is indeed close to μ. For details see the full version of this paper.

Claim 4. Let μ_1 be the distribution of $a|\mathcal{S}$. Then $|\mu_1 - \mu| \leq 2/9$.

We turn to the "Furthermore" part of of Lemma 3. The protocol Π_1 satisfies the premises of the lemma, except it has a high communication cost. This is due to the fact that Alice explicitly sends a to Bob. To reduce the communication, Alice will instead send $O(I)$ random hash values of a, and Bob will add corresponding consistency constraints to his set of candidates. The final protocol Π_2 is given in Figure 1.

Let $\mathcal{E}$ denote the event that in step 4 of the protocol, Bob finds an element $x_i \neq a$ (that is, the probability that the protocol outputs "success" but $x_A \neq x_B$). We upper bound the probability of $\mathcal{E}$. Given $a \in \mathcal{A}$ and $x_i \in \mathcal{B}$ such that $a \neq x_i$, the probability (over possible choices of the hash functions) that $h_j(a) = h_j(x_i)$

Information-cost sampling protocol Π_2

1. Alice computes the set $\mathcal{A}$. Bob computes the set $\mathcal{B}$.
2. If $\mathcal{A} = \emptyset$, the protocol fails. Otherwise, Alice finds the first element $a \in \mathcal{A}$ and sets $x_A = a$. She then computes $d = \lceil 211I \rceil$ random hash values $h_1(a), \ldots, h_d(a)$, where the hash functions are evaluated using public randomness.
3. Alice sends the values $\{h_j(a)\}_{1 \le j \le d}$ to Bob.
4. Bob finds the first index τ such that there is a $b \in \mathcal{B}$ for which $h_j(b) = h_j(a)$ for $j = 1..d$ (if such an τ exists). Bob outputs $x_B = x_\tau$. If there is no such index, the protocol fails.
5. Bob outputs x_B ("success").
6. Alice outputs x_A.

Fig. 1. The sampling protocol Π_2 from Lemma 3

for $j = 1..d$ is $2^{-d} \le 2^{-211I}$. For any t, $\mathbf{P}[t \in \mathcal{B}] \le \frac{1}{|\mathcal{U}|} \sum_{x_i \in \mathcal{U}} p_B(x_i) q_B(x_i) \cdot 2^{50(I+1)} = \frac{1}{|\mathcal{U}|} \sum_{x_i \in \mathcal{U}} \nu_B(x_i) \cdot 2^{50(I+1)} = 2^{50(I+1)}/|\mathcal{U}|$. Thus, by a union bound we have

$$\begin{aligned} \mathbf{P}[\mathcal{E}] &\le \mathbf{P}[\exists x_i \in \mathcal{B}\ s.t\ x_i \ne a\ \wedge\ h_j(a) = h_j(x_i)\ \forall\ j = 1, \ldots, d] \le \\ &\le T \cdot 2^{50(I+1)} \cdot 2^{-d}/|\mathcal{U}| = 2^{150(I+1)} \cdot 60I \cdot 2^{-211I} \ll 2^{-60I}. \end{aligned} \tag{3}$$

By a slight abuse of notation, let Π_2 (Π_1) be the distribution of Π_2's ((Π_1's)) output. Note that if $\mathcal{E}$ does not occur, then the outcome of the execution of Π_2 is identical to the outcome of Π_1. Since $\mathbf{P}[\mathcal{E}] \le 2^{-60I}$, we have $|\Pi_2 - \Pi_1| = \Pr[\mathcal{E}] \cdot |[\Pi_2|\mathcal{E}] - [\Pi_1|\mathcal{E}]| \le 2 \cdot 2^{-60I} \ll 2^{-59I}$,which completes the proof.

Remark 2. The communication cost of the sampling protocol Π_2 can be reduced from $O(I_\mu)$ to $O(1)$ (more precisely, to only two bits) in the following way: Instead of having Alice compute the hash values privately and send them to Bob in step 2 and 3 of the protocol, the players can use their shared randomness to sample $d = O(I_\mu)$ random hash values $h_1(b_1), \ldots, h_d(b_d)$ (where the b_i's are random independent strings in $\mathcal{U}$), and Alice will only send Bob a single bit indicating whether those hash values match the hashing of her string a (i.e, $h_i(b_i) = h_i(a)$ for all $i \in [d]$). In step 4 Bob will act as before, albeit comparing the hashes of his candidate b to the random hashes $h_i(b_i)$, and output success ("1") if the hashes match. Note that this modification incurs an additional loss of $2^{-d} = 2^{-O(I_\mu)}$ in the success probability of the protocol (as this is the probability that $h_i(b_i) = h_i(a)$ for all $i \in [d]$), but since the success probability we are shooting for is already of the order $2^{-O(I_\mu)}$, we can afford this loss. This modification was discovered in [KLL+12].

Theorem 2 will now follow as a direct application of Lemma 3. The idea is that Alice and Bob will use the (weak) correlated sampling procedure guaranteed by Lemma 3, with μ taken to be the distribution of the information-optimal protocol $\pi(X, Y)$, whose information cost is I_μ, and the distributions ν_A and ν_B

taken to be its marginals π_x and π_y respectively. The premises of the lemma will guarantee that with probability $2^{-O(I_\mu)}$ the parties sample the correct transcript, using $O(I_\mu)$ bits of communication, which in turn yields the (small) advantage we are looking for in computing the value of f. For the full proof we refer the reader to the full version of this paper [BW11].

Remark 3. Using similar techniques, it was recently shown in [Bra11] that any function f whose information complexity is I has communication cost at most $2^{O(I)}$ [1], thus implying that $IC(f) \geq \Omega(\log(CC(f)))$. We note that this result can be easily derived (up to constant factors) from Theorem 2. Indeed, applying the "compressed" protocol $2^{O(I)} \log(1/\epsilon)$ independent times and taking a majority vote guarantees an error of at most ϵ (by a standard Chernoff bound[2]), with communication $O(I) \cdot 2^{O(I)} = 2^{O(I)}$. Thus, our result is strictly stronger than the former one.

Acknowledgments. We thank Ankit Garg and several anonymous reviewers for their useful comments and helpful discussions.

References

[AL11] Asharov, G., Lindell, Y.: A full proof of the BGW protocol for perfectly-secure multiparty computation. Electronic Colloquium on Computational Complexity (ECCC) 18, 36 (2011), http://dblp.uni-trier.de

[BBCR10] Barak, B., Braverman, M., Chen, X., Rao, A.: How to compress interactive communication. In: Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (2010)

[BOGW88] Ben-Or, M., Goldwasser, S., Wigderson, A.: Completeness theorems for non-cryptographic fault-tolerant distributed computation. In: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, pp. 1–10. ACM (1988)

[BR10] Braverman, M., Rao, A.: Information equals amortized communication. CoRR, abs/1106.3595 (2010)

[BR11] Braverman, M., Rao, A.: Information equals amortized communication. Arxiv preprint arXiv:1106.3595 (2011)

[Bra11] Braverman, M.: Interactive information complexity. Electronic Colloquium on Computational Complexity (ECCC) 18, 123 (2011)

[BW11] Braverman, M., Weinstein, O.: A discrepancy lower bound for information complexity. CoRR, abs/1112.2000 (2011)

[BYCKO93] Bar-Yehuda, R., Chor, B., Kushilevitz, E., Orlitsky, A.: Privacy, additional information and communication. IEEE Transactions on Information Theory 39(6), 1930–1943 (1993)

[BYJKS04] Bar-Yossef, Z., Jayram, T.S., Kumar, R., Sivakumar, D.: An information statistics approach to data stream and communication complexity. Journal of Computer and System Sciences 68(4), 702–732 (2004)

[1] More precisely, it shows that for any distribution μ, $D^{\mu}_{\varepsilon+\delta}(f) = 2^{O(1+\mathsf{IC}_\mu(f,\varepsilon)/\delta^2)}$.

[2] See N.Alon and J. Spencer, "The Probabilistic Method" (Third Edition) ,Corollary A.1.14, p.312.

[CK89] Chor, B., Kushilevitz, E.: A zero-one law for boolean privacy. In: Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, pp. 62–72. ACM (1989)

[CSWY01] Chakrabarti, A., Shi, Y., Wirth, A., Yao, A.: Informational complexity and the direct sum problem for simultaneous message complexity. In: Werner, B. (ed.) Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science, October 14-17, pp. 270–278. IEEE Computer Society, Los Alamitos (2001)

[HJMR07] Harsha, P., Jain, R., McAllester, D., Radhakrishnan, J.: The communication complexity of correlation. In: Twenty-Second Annual IEEE Conference on Computational Complexity, CCC 2007, pp. 10–23. IEEE (2007)

[Jai10] Jain, R.: A strong direct product theorem for two-way public coin communication complexity. Arxiv preprint arXiv:1010.0846 (2010)

[JSR08] Jain, R., Sen, P., Radhakrishnan, J.: Optimal direct sum and privacy trade-off results for quantum and classical communication complexity. Arxiv preprint arXiv:0807.1267 (2008)

[Kla04] Klauck, H.: Quantum and approximate privacy. Theory Comput. Syst. 37(1), 221–246 (2004)

[Kla10] Klauck, H.: A strong direct product theorem for disjointness. In: Proceedings of the 42nd ACM Symposium on Theory of Computing, pp. 77–86. ACM (2010)

[KLL+12] Kerenidis, I., Laplante, S., Lerays, V., Roland, J., Xiao, D.: Lower bounds on information complexity via zero-communication protocols and applications. Arxiv preprint arXiv:1204.1505 (2012)

[KN97] Kushilevitz, E., Nisan, N.: Communication complexity. Cambridge University Press, New York (1997)

[KSDW04] Klauck, H., Spalek, R., De Wolf, R.: Quantum and classical strong direct product theorems and optimal time-space tradeoffs. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 12–21. IEEE (2004)

[LSS08] Lee, T., Shraibman, A., Spalek, R.: A direct product theorem for discrepancy. In: 23rd Annual IEEE Conference on Computational Complexity, CCC 2008, pp. 71–80. IEEE (2008)

[MNSW95] Miltersen, P.B., Nisan, N., Safra, S., Wigderson, A.: On data structures and asymmetric communication complexity. In: Proceedings of the Twenty-Seventh Annual ACM Symposium on Theory of Computing, pp. 103–111. ACM (1995)

[Raz92] Razborov, A.A.: On the distributional complexity of disjointness. Theor. Comput. Sci. 106(2), 385–390 (1992)

[Sha03] Shaltiel, R.: Towards proving strong direct product theorems. Computational Complexity 12(1), 1–22 (2003)

[Smi88] Smirnov, D.V.: Shannon's information methods for lower bounds for probabilistic communication complexity. Master's thesis, Moscow State University (1988)

[Vio11] Viola, E.: The communication complexity of addition. Electronic Colloquium on Computational Complexity (ECCC) 18, 152 (2011)

[Yao79] Yao, A.C.-C.: Some complexity questions related to distributive computing (preliminary report). In: STOC, pp. 209–213 (1979)

On the Coin Weighing Problem with the Presence of Noise

Nader H. Bshouty

Technion, Israel
bshouty@cs.technion.ac.il

Abstract. In this paper we consider the following coin weighing problem: Given n coins for which some of them are counterfeit with the same weight. The problem is: given the weights of the counterfeit coin and the authentic coin, detect the counterfeit coins a with minimal number of weighings. This problem has many applications in computational learning theory, compressed sensing and multiple access adder channels.

An old optimal non-adaptive polynomial time algorithm of Lindstrom can detect the counterfeit coins with $O(n/\log n)$ weighings. An information theoretic proof shows that Lindstrom's algorithm is optimal. In this paper we study non-adaptive algorithms for this problem when some of the answers of the weighings received are incorrect or unknown.

We show that no coin weighing algorithm exists that can detect the counterfeit coins when the number of incorrect weighings is more than $1/4$ fraction of the number of weighings. We also give the tight bound $\Theta(n/\log n)$ for the number of weighings when the number of incorrect answers is less than $1/4$ fraction of the number of weighings.

We then give a non-adaptive polynomial time algorithm that detects the counterfeit coins with $k = O(n/\log\log n)$ weighings even if some constant fraction of the answers of the weighings received are incorrect. This improves Bshouty and Mazzawi's algorithm [7] that uses $O(n)$ weighings. This is the first sublinear algorithm for this problem.

1 Introduction

The coin weighing problem with the presence of noise can be reduced to the following *reconstructing hidden* $(0,1)$*-vectors* problem. Let v be a size n hidden $(0,1)$-vector. Suppose that we are allowed to ask queries of the form

$$Q(x) := x^T v,$$

where $x \in \{0,1\}^n$. Suppose that some of the answers received are incorrect (also called *errors*) and some of them are unknown, labeled with "?" (also called *erasures*). Our goal is to exactly reconstruct the hidden vector v with a minimal number of queries.

We distinguish between two type of algorithms for solving our problems. Non-adaptive algorithms are algorithms that ask all queries in advance, before receiving any answer. On the other hand, adaptive algorithms are algorithms that

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 471–482, 2012.
© Springer-Verlag Berlin Heidelberg 2012

take into account the outcome of previous queries. In this paper we study non-adaptive algorithms for the coin weighing problem.

The coin weighing problems were heavily studied in the noiseless case, that is, the case where all the answers are available and correct. The information theoretic lower bound for the query complexity (number of weighings) is

$$\Omega\left(\frac{n}{\log n}\right).$$

This problem was studied by Cantor in [8], Soderberg and Shapiro in [35] Erdös and Rényi in [18], Lindström in [26–29] and Cantor and Mills in [10]. Lindström [26] and independently Cantor and Mills [10] gave a non-adaptive polynomial time algorithm for the problem with query complexity that matches the lower bound. Simplifications appear in [29, 1, 5].

The coin weighing problem is a combinatorial search problem that is motivated by real-world problems, such as, problems in computational learning theory, compressed sensing and multiple access adder channels. As such, it is important to study it in the presence of noise that causes some incorrect and unknown answers.

In this paper we study the coin weighing problem (reconstructing hidden $(0,1)$-vector problem) with noise. We say that a coin weighing algorithm (reconstructing algorithm) *tolerates* m incorrect or/and unknown weighings (incorrect answers) if the algorithm can detect the counterfeit coins (reconstruct the hidden vector) when the number of incorrect or/and unknown weighings (answers of the queries) is bounded above by m. Obviously, an algorithm that can tolerate m incorrect weighings can also tolerate m unknown weighings.

We first study non-constructive non-adaptive algorithms for the coin weighing problem. We prove

Theorem 1. *There is no coin weighing algorithm that can detect the counterfeit coins when the number of incorrect (respectively, unknown) weighings is more than* $1/4$ *(respectively, more than* $1/2$*) fraction of the number of weighings.*

We then prove the following Theorem that shows that the ratio $1/4$ is tight and also gives an upper bound on the number of weighings.

Theorem 2. *Let* $0 \le \epsilon < 1/4$ *(respectively,* $0 \le \epsilon < 1/2$*) be a constant. Let*

$$k = \frac{(2\log 3)n}{(1-2\epsilon)\log n} + O\left(\frac{n\log\log n}{\log^2 n}\right).$$

(respectively,

$$k = \frac{(2\log 3)n}{(1-\epsilon)\log n} + O\left(\frac{n\log\log n}{\log^2 n}\right).)$$

There exists a non-adaptive coin weighing algorithm that detects the counterfeit coins with k *weighings and tolerates* $m = \epsilon k$ *incorrect (respectively, unknown) weighings.*

We also prove the following lower bound.

Theorem 3. *Let $0 \leq \epsilon < 1/4$ (respectively, $0 \leq \epsilon < 1/2$) be a constant. Every non-adaptive coin weighing algorithm with k weighings that tolerates $m = \epsilon k$ incorrect (respectively, unknown) weighings must satisfies*

$$k \geq \frac{2n}{(1-\epsilon)\log n} - O\left(\frac{n \log\log n}{\log^2 n}\right).$$

(respectively,

$$k \geq \frac{4n}{(2-\epsilon)\log n} - O\left(\frac{n \log\log n}{\log^2 n}\right).)$$

For polynomial time non-adaptive algorithms for the coin weighing problem, in [7], Bshouty and Mazzawi gave a technique that show that for every $1 \leq s \leq n$ there is a polynomial time algorithm that detects the counterfeit coins with $O(n/\log s)$ weighings when $O(n/s)$ of the answers are incorrect. When a constant fraction of the weighings are incorrect Bshouty and Mazzawi's algorithm requires $O(n)$ weighings. In this paper we prove

Theorem 4. *Let $\epsilon > 0$ be any small constant. There exists a non-adaptive polynomial time coin weighing algorithm with*

$$k = O\left(\frac{n}{\log\log n}\right)$$

weighings that tolerates $(1/16 - \epsilon)k$ incorrect weighings.

1.1 Applications

In this subsection we introduce one application for the coin weighing problem with noise. We introduce the signature coding problem for the multiple access adder channels [26, 10, 11, 23, 17, 15, 25, 3].

Consider n stations or users that transmit information using a common channel. Each user i has a component code $C_i = \{0^k, x_i\} \subset \{0,1\}^k$. The codes C_i for $i \in [n]$ are of the same length k. Each user i wants to send one of the words $y_i \in C_i$. Assume that codewords sent through the channel by the users are bit and block synchronized. The codeword go through an adder that sends $Y = \sum_{i=1}^{n} y_i$ through a noisy channel. We denote by $\tilde{Y}$ the output of the channel. The goal is to recover all the messages of the users using $\tilde{Y}$.

There are two models for this problem, the *permanently active* model where all stations are active all the time and the *partially active* model where at most m stations are active in each transmission (the inactive stations turn off their transmitter, an action that is equivalent to sending the zero codeword 0^k).

A non-adaptive algorithm for the weighing problems with the presence of noise yields signature codes for noisy channels in the permanently active model. The existence of such codes was studied in the literature [11, 17, 36, 12, 15]. Before

we present the results, we note that no technique was known from the literature that can handle erasures (incorrect output).

In [11], Chang and Weldon gave the first technique to reconstruct signature codes for noisy channels. Using this technique, one can build a codes of length $O(n/(\epsilon \log n))$ for n users, where a messages can be correctly recovered if the error vector, $\tilde{Y} - Y$, has L_1 norm that is smaller that $O(n^{1-\epsilon})$. In other words, the users' messages can be recovered from $\tilde{Y}$ if

$$\sum_i |\tilde{Y}_i - Y_i| \leq O(n^{1-\epsilon}).$$

In [36], Wilson gave a similar technique. Using Wilson's technique, one can reconstruct codes of length $O(n/(\epsilon \log n))$ for n users, where a message can be correctly recovered if the error vector, $\tilde{Y} - Y$, has L_1 norm that is smaller than $O(n^{1-\epsilon})$. Moreover, the message can be recovered correctly even if we have $O(n^{\epsilon}/(\epsilon \log n))$ consecutive errors of magnitude $O(n^{1-\epsilon})$. That is, if the error vector $\tilde{Y} - Y$ has only $O(n^{\epsilon}/(\epsilon \log n))$ non-zero entries, where these non-zero entries are consecutive and each entry is bounded by $O(n^{1-\epsilon})$. Additional constructions also appear in [13, 12]. Finally, in [15], Cheng et. al. gave a code for n users of length n, where the messages can be recovered if the error vector has L_1 norm that is smaller than $\lfloor (n/2-1)/2 \rfloor$.

Bshouty and Mazzawi, [7], gave a signature codes for the permanently active model of length $O\left(n/(\epsilon \log n)\right)$, where the receiver is able to decode all the users' messages correctly with the presence of $O(n^{1-\epsilon})$ errors and erasures. That is, if the error vector has *Hamming weight* (rather than the L_1 norm) that is smaller than $O(n^{1-\epsilon})$. Moreover, if the errors are consecutive, then the message can be recovered even when we have $O(n/(\epsilon \log n))$ errors.

Note that the L_1 norm is always greater than or equal to the number of errors. Therefore, Bshouty and Mazzawi result in particular, implies all the results from the literature. On the other hand, since $\tilde{Y}, Y \in \{0, 1, \ldots, n\}^n$, one random error yield L_1 norm of average magnitude $O(n)$. Therefore, previous algorithms cannot handle one random error.

All the above results cannot handle constant rate error and erasures when code length is less than linear in the number of users. In this paper we give the first code of sublinear length for n users that can handle $O(n)$ errors and erasures.

2 Lower and Upper Bounds

In this section we give lower and upper bounds for the coin weighing problem with noise.

A non-adaptive algorithm for the coin weighing problem can be expressed as a $k \times n$ $(0,1)$-matrix M, called *search matrix*, where the ith row M_i in M is the ith query $Q(M_i) := M_i^T x$ in the algorithm. This matrix satisfies $Mx \neq My$ for every two distinct $x, y \in \{0,1\}^n$. That is, the answers to the queries $Q(M_i) = M_i^T x$, $i = 1, \ldots, k$, uniquely determine the hidden vector $x \in \{0,1\}^n$.

The algorithm *tolerates m incorrect answers* if and only if for every two vectors $u, v \in \mathbf{R}$ with $wt(u), wt(v) \leq m$, where $wt(u)$ is the number of nonzero entries in u, and every two distinct vectors $x, y \in \{0,1\}^n$ we have $Mx+u \neq My+v$. In other words, the answers Mx to the queries with any noise u that forces at most m answers to be incorrect, $Mx+u$, uniquely determines the hidden vector x. In that case we say that M *tolerates m incorrect answers*.

The algorithm *tolerates m unknown answers* if and only if for every $(k-m)\times n$ submatrix M' of M and every two distinct vectors $x, y \in \{0,1\}^n$ we have $M'x \neq M'y$. In other words, the answers Mx to the queries with any m missing answers uniquely determines the hidden vector x. In that case we say that M *tolerates m unknown answers*.

We now prove.

Lemma 1. *Let M be a $k \times n$ search matrix. Then*

1. *M tolerates m incorrect answers if and only if for every nonzero $z \in \{-1,0,1\}^n$ we have $wt(Mz) > 2m$.*
2. *M tolerates m unknown answers if and only if for every nonzero $z \in \{-1,0,1\}^n$ we have $wt(Mz) > m$.*
3. *M tolerates m incorrect answers if and only if M tolerates $2m$ unknown answers.*

Proof. (1, $\Rightarrow$) If M tolerates m incorrect answers then for every $u, v \in \mathbf{R}$ with $wt(u), wt(v) \leq m$ and every distinct $x, y \in \{0,1\}^n$ we have $Mx+u \neq My+v$. Suppose for the contrary there is $z \in \{-1,0,1\}^n$ such that $wt(Mz) \leq 2m$. Write Mz as a sum of two vectors u' and v', each of weight less or equal to m. Write $z = x'-y'$ where $x', y' \in \{0,1\}^n$. Then $u'+v' = Mz = M(x'-y')$ and therefore $Mx' + (-u') = My' + v'$. This is a contradiction.

(1, $\Leftarrow$) Now suppose for every $z \in \{-1,0,1\}^n$ we have $wt(Mz) > 2m$. Suppose for contrary there are two distinct vectors $x, y \in \{0,1\}^n$ and $u, v \in \mathbf{R}$ with $wt(u), wt(v) \leq m$ where $Mx+u = My+v$. Then $M(x-y) = (v-u)$, $z := x-y \in \{-1,0,1\}^n$ and $wt(Mz) = wt(v-u) \leq 2m$. This is a contradiction.

The proof of 2 is similar and 3 follows from 1 and 2.

We now give a lower bound for the number of incorrect and unknown answers that any M can tolerate.

Theorem 1. *There is no coin weighing algorithm that can detect the counterfeit coins when the number of incorrect (respectively, unknown) weighings is more than $1/4$ (respectively, more than $1/2$) fraction of the number of weighings.*

Proof. Consider any $k \times n$ $(0,1)$-matrix (search matrix) M that tolerates αk incorrect answers. Let $M^{(i)}$ be the ith column of M and M_i the ith row of M. Let $\{e_i \mid i = 1, \ldots, n\}$ be the standard basis vectors of $\mathbf{R}^n$. Let N_w, $w = 0, 1, \ldots, n$ be the fraction of the number of rows in M of weight equal to w. That is

$$N_w = \frac{|\{j \mid wt(M_j) = w\}|}{k}.$$

Obviously, $N_0 = 0$ and

$$N_1 + N_2 + \cdots + N_n = 1. \tag{1}$$

Since M tolerates αk incorrect answers, for every $1 \le i_1 < i_2 \le n$, by Lemma 1, we have $wt(M(e_{i_1} - e_{i_2})) = wt(M^{(i_1)} - M^{(i_2)}) > 2(\alpha k)$. Therefore

$$\begin{aligned}\binom{n}{2}(2\alpha k) &< \sum_{1\le i_1<i_2\le n} wt(M^{(i_1)} - M^{(i_2)}) \\ &= ((n-1)1N_1 + (n-2)2N_2 + \cdots \\ &\quad +2(n-2)N_{n-2} + 1(n-1)N_{n-1})k.\end{aligned}$$

The latter equality follows from the fact that each row in M of weight w contributes $w(n-w)$ to the sum. Therefore, by (1),

$$\begin{aligned}\alpha n(n-1) &< (n-1)1N_1 + (n-2)2N_2 + \cdots + 1(n-1)N_{n-1} \\ &\le \frac{n}{2}\frac{n}{2}(N_1 + \cdots + N_{n-1} + N_n) = \frac{n^2}{4}.\end{aligned}$$

and

$$\alpha < \frac{1}{4}\frac{n}{n-1} = \frac{1}{4} + o(1).$$

We now give a lower bound for the maximal possible incorrect weighings and unknown weighings for which a coin weighing algorithm exists and an upper bound for the total number of weighings.

Theorem 2. *Let $0 \le \epsilon < 1/4$ (respectively, $0 \le \epsilon < 1/2$) be a constant. Let*

$$k = \frac{(2\log 3)n}{(1-2\epsilon)\log n} + O\left(\frac{n\log\log n}{\log^2 n}\right).$$

(respectively,

$$k = \frac{(2\log 3)n}{(1-\epsilon)\log n} + O\left(\frac{n\log\log n}{\log^2 n}\right).)$$

There exists a non-adaptive coin weighing algorithm that detects the counterfeit coins with k weighings and tolerates $m = \epsilon k$ incorrect (respectively, unknown) weighings.

Proof. Consider a random uniform $k \times n$ search matrix M. We will show that $\mathbf{Pr}[(\exists z \in \{-1,0,1\}^n)\ wt(Mz) < 2m] < 1$ which by Lemma 1 implies the result.

Let $M_i \in \{0,1\}^n$ be the ith row of M. Consider a vector $z \in \{-1,0,1\}^n$ with $w = w^+ + w^-$ nonzero entries, w^+ entries that are equal to one and w^- entries that are equal to -1 (and $n-w$ entries that are zero). It is easy to see that

$$q_{w^+,w^-} := \mathbf{Pr}[M_i z = 0] = \frac{\binom{w}{w^-}}{2^w} \le \sqrt{\frac{2}{3w}}$$

and $q_{w^+,w^-} \le 1/2$ for every $w \ge 1$. Therefore for $m = \epsilon k$,

$$\mathbf{Pr}[wt(Mz) < 2m] = \sum_{j=1}^{2m} (1 - q_{w^+,w^-})^{2m-j} q_{w^+,w^-}^{k-2m+j} \binom{k}{2m-j}.$$

Since $m < k/4$ and $q_{w^+,w^-} \le 1/2$, we have

$$\begin{aligned}(1 - q_{w^+,w^-})^{2m-j} q_{w^+,w^-}^{k-2m+j} &\le (1 - q_{w^+,w^-})^{2m} q_{w^+,w^-}^{k-2m} \\ &\le (1 - q_{w^+,w^-})^{k/2} q_{w^+,w^-}^{k/2} \le 1/2^k\end{aligned}$$

and therefore

$$\mathbf{Pr}[wt(Mz) < 2m] \le \sum_{j=1}^{2m} 2^{-k} \binom{k}{2m-j} \le 2^{H(2\epsilon)k-k} = 2^{-ck} \tag{2}$$

for some constant $c > 0$ where $H(x) = -x \cdot \log x - (1-x) \cdot \log(1-x)$ is the binary entropy function. Since $(1 - q_{w^+,w^-})^{2m-j} q_{w^+,w^-}^{k-2m+j} \le q_{w^+,w^-}^{k-2m}$ we also have

$$\begin{aligned}\mathbf{Pr}[wt(Mz) < 2m] &\le q_{w^+,w^-}^{k-2m} \sum_{j=1}^{2m} \binom{k}{2m-j} \le q_{w^+,w^-}^{(1-2\epsilon)k} 2^{H(2\epsilon)k} \\ &\le \left(\frac{2}{3w}\right)^{\frac{1-2\epsilon}{2}k} 2^{H(2\epsilon)k}. \end{aligned} \tag{3}$$

Now by (2) and (3) we have

$$\begin{aligned}\mathbf{Pr}[(\exists z)\ wt(Mz) < 2m] &\le \sum_{w=1}^{n} \binom{n}{w} 2^w \min\left(\left(\frac{2}{3w}\right)^{\frac{1-2\epsilon}{2}k} 2^{H(2\epsilon)k}, 2^{-ck}\right) \\ &\le \sum_{w=1}^{n/\log^3 n} \binom{n}{w} 2^w 2^{-ck} \\ &\quad + \sum_{w=n/\log^3 n}^{n} \binom{n}{w} 2^w \left(\frac{2}{3w}\right)^{\frac{1-2\epsilon}{2}k} 2^{H(2\epsilon)k}. \end{aligned} \tag{4}$$

For $w \le n/\log^3 n$ we have

$$\binom{n}{w} 2^w 2^{-ck} \le n^{2w} 2^{-ck} \le 2^{\frac{2n}{\log^2 n} - c\frac{n}{\log n}} < \frac{1}{n}. \tag{5}$$

For $w \ge w_0 := n/\log^3 n$ we have

$$\begin{aligned}\binom{n}{w} 2^w \left(\frac{2}{3w}\right)^{\frac{1-2\epsilon}{2}k} 2^{H(2\epsilon)k} &\le 3^n \left(\frac{2}{3w_0}\right)^{\frac{1-2\epsilon}{2}k} 2^{H(2\epsilon)k} \\ &\le 2^{(\log 3)n - \frac{1-2\epsilon}{2}k \log n} \cdot 2^{H(2\epsilon)k + \frac{1-2\epsilon}{2}k \log((2/3)\log^3 n)} \\ &= 2^{-O\left(\frac{n \log\log n}{\log n}\right)} 2^{H(2\epsilon)k + \frac{1-2\epsilon}{2}k \log((2/3)\log^3 n)} \\ &< \frac{1}{n}. \end{aligned} \tag{6}$$

Therefore, by (4), (5) and (6) we have $\mathbf{Pr}[(\exists z)\ wt(Mz) < 2m] < 1$ and the result follows.

We now give a lower bound for the number of weighings. We prove

Theorem 3. *Let $0 \le \epsilon < 1/4$ (respectively, $0 \le \epsilon < 1/2$) be a constant. Every non-adaptive coin weighing algorithm with k weighings that tolerates $m = \epsilon k$ incorrect (respectively, unknown) weighings must satisfy*

$$k \ge \frac{2n}{(1-\epsilon)\log n} - O\left(\frac{n \log\log n}{\log^2 n}\right).$$

(respectively,

$$k \ge \frac{4n}{(2-\epsilon)\log n} - O\left(\frac{n \log\log n}{\log^2 n}\right).)$$

Proof. Let M be any $k \times n$ search matrix that tolerates ϵk incorrect answers. By Lemma 1, for every $x, y \in \{0,1\}^n$ where $x \ne y$ we have $dist(Mx, My) := wt(Mx - My) = wt(M(x-y)) > 2\epsilon k$. Let M_i be the ith row of M and $wt(M_i) = w_i$. Let $w = (w_1/2, \ldots, w_k/2)$. Then $dist(Mx - w, My - w) = dist(Mx, My) > 2\epsilon k$ for every $x, y \in \{0,1\}^n$ where $x \ne y$.

Let x be a uniform random $(0,1)$-vector in $\{0,1\}^n$. By Moivre-Laplace theorem (that follows from Stirling's approximation) the probability that $|M_i x - w_i/2| \ge 2\sqrt{w_i \log w_i}$ is less than $O(1/w_i^8)$. If $w_i > \sqrt{n}$ then the probability that $|M_i x - w_i/2| \ge 2\sqrt{n \log n}$ is less than $O(1/n^4)$ and if $w_i \le \sqrt{n}$ then $|M_i x - w_i/2| \le 2\sqrt{n} \le 2\sqrt{n \log n}$ with probability 1. Therefore the probability that $Mx - w \in [-q, q]^k$ where $q = 2\sqrt{n \log n}$ is greater than $1 - O(1/n^3)$. This shows that for at least $2^n(1 - O(1/n^3)) \ge 2^{n-1}$ of the vectors $x \in \{0,1\}^n$ we have $Mx - w \in [-q, q]^k$. Let $S \subseteq \{0,1\}^n$ be the set of such vectors and $W = \{Mx - w \mid x \in S\} \subseteq [-q, q]^k$. Then $|W| = |S| \ge 2^{n-1}$.

For $u \in W$ let $B(u) = \{y \in [-q, q]^k \mid dist(u, y) \le \epsilon k\}$. Since the vectors in W are at distance more than $2\epsilon k$ from each other, for every two distinct vectors $w_1, w_2 \in W$, we have $B(w_1) \cap B(w_2) = \emptyset$. We also have, for each $v \in W$

$$|B(v)| = \sum_{i=0}^{\epsilon k} (2q)^i \binom{k}{i}.$$

Therefore

$$\begin{aligned}(2q+1)^k = |[-q,q]^k| &\ge \left|\bigcup_{v \in W} B(v)\right| = \sum_{v \in W} |B(v)| = |W| \sum_{i=1}^{\epsilon k} (2q)^i \binom{k}{i} \\ &\ge |W|(2q)^{\epsilon k}\binom{k}{\epsilon k} \ge |W|(2q)^{\epsilon k} 2^{\epsilon k} \ge 2^{n-1}(4q)^{\epsilon k} \ge 2^n (2q+1)^{\epsilon k}\end{aligned}$$

This implies the result.

3 Polynomial Time Algorithms

In this section, we present a polynomial time algorithm for the coin weighing problem with sublinear number of weighings that can tolerate incorrect weighings that are constant fraction of the number of weighings.

We build a search matrix which is a Kronecker product of two matrices. The first matrix is a generating matrix of a $[N, K, D]$ code where $N = n/\sqrt{\log n}$ is the length of the code, $D = (1/2 - o(1))N$ is the minimum distance and $K = O(N)$ is the dimension. Concatenation code is an example of such code that has polynomial time decoding algorithm. The second matrix is a $O(\sqrt{\log n}/\log\log n) \times \sqrt{\log n}$ dimensional matrix that tolerate $1/4 - o(1)$ fraction of incorrect answers. This matrix can be built by exhaustive search and has polynomial time (in n) algorithm. We show that the combined matrix tolerates $(1/16 - o(1))t$ incorrect answers where $t = O(n/\log\log n)$ is the number of weighings. We also give a polynomial time algorithm for detecting the counterfeit coins.

We show the following,

Theorem 4. *Let $\epsilon > 0$ be any small constant. There exists a non-adaptive polynomial time coin weighing algorithm with*

$$k = O\left(\frac{n}{\log\log n}\right)$$

weighings that tolerates $(1/16 - \epsilon)k$ incorrect weighings (respectively, $(1/4 - \epsilon)k$ unknown weighings).

Proof. We give the proof of the case of incorrect weighings. For unknown weighings the proof is similar.

To prove the result, we give a search matrix for the problem. That is, we build a $k \times n$ $(0,1)$-matrix B such that given $Bv + e$, where $v \in \{0,1\}^n$ and e is any n-vector with Hamming weight smaller than $(1/16 - \epsilon)n$, one can reconstruct v in polynomial time.

The following lemma is proved in the full version of the paper. A weaker version of this lemma was proved in [7].

Lemma 2. *Let $\mathcal{C}$ be a linear code $[p, k, d]$ over $\mathbf{Z}_2$ with the generating $k \times p$ matrix G. Let $\mathcal{D}$ be a polynomial time decoding algorithm for $\mathcal{C}$ that decodes in the presence of $d' \leq \frac{d-1}{2}$ errors. Let $\bar{G} \in \mathbf{Z}^{k \times p}$ be equal to G.*

There is an algorithm that runs in polynomial time $poly(p, \log n)$ and satisfies the following: Given $b = \bar{G}^T w + e$, where $w \in \{0, 1, \ldots, n\}^k$ and $e, b \in \mathbf{Z}^p$ are any p-vectors. If $wt(e) \leq d'$, the algorithm returns w. If $wt(e) > d'$ the algorithm returns some vector $u \in \{0, 1, \ldots, 2n\}^k$.

Choose two small constants ϵ_1 and ϵ_2 such that $(1/4 - \epsilon_1)(1/4 - \epsilon_2/2) = (1/16 - \epsilon)$. Consider all the $(0,1)$-matrices in $\{0,1\}^{t \times s}$ where

$$t = \frac{(8\log 3)\sqrt{\log n}}{\log\log n} \text{ and } s = \sqrt{\log n}.$$

There are at most $2^{ts} \le n$ such matrices. By Theorem 2, there is $M \in \{0,1\}^{t\times s}$ that tolerates $(1/4-\epsilon_1)t$ incorrect answers for any small constant ϵ_1. By Lemma 1 and the choice of t and s, such matrix can be found in polynomial time and given $Mx+e$ where $x \in \{0,1\}^s$ and $wt(e) \le (1/4-\epsilon_1)t$ one can find x and e by exhaustive search in polynomial time. We denote this algorithm by $\mathcal{ES}$.

Let $r = n/s$. Let $\mathcal{C}$ be a linear code $[N,K,D] := [c_1 r, r, (1/2-\epsilon_2)c_1 r]$ over $\mathbf{Z}_2$ with generating matrix $G \in \{0,1\}^{r\times(c_1 r)}$ and polynomial time decoding algorithm $\mathcal{D}$ where c_1 is a constant. Concatenated codes are an example to such code. See for example [34]. Now, we regard G as a $(0,1)$-matrix $\bar{G}$ over $\mathbf{Z}$. Let,

$$B = \bar{G}^T \otimes M = \begin{pmatrix} g_{1,1}M & g_{2,1}M & \dots & g_{r,1}M \\ g_{1,2}M & g_{2,2}M & \dots & g_{r,2}M \\ \vdots & \vdots & \ddots & \vdots \\ g_{1,c_1 r}M & g_{2,c_1 r}M & \dots & g_{r,c_1 r}M \end{pmatrix}.$$

Since $rs = n$, the matrix B is of size $k \times n$ where

$$k = \frac{(8c_1 \log 3)n}{\log\log n}.$$

We now argue that B tolerates $k' = (1/4-\epsilon_1)(1/4-\epsilon_2/2)k = (1/16-\epsilon)k$ incorrect answers and given $b = Bv + e$, where $v \in \{0,1\}^n$, $e \in \mathbf{Z}^k$ and $wt(e) \le k'$, one can reconstruct v in polynomial time.

Let M_i be the ith row of M. Divide v into size s-vectors

$$v = \left(\begin{array}{c} v^{(1)} \\ \hline v^{(2)} \\ \hline \vdots \\ \hline v^{(r)} \end{array}\right) \text{ and let } w^{(i)} = \left(\begin{array}{c} M_i v^{(1)} \\ \hline M_i v^{(2)} \\ \hline \vdots \\ \hline M_i v^{(r)} \end{array}\right)$$

for $i = 1, 2, \dots, t$. Then for $b^{(i)} = (b_i, b_{t+i}, b_{2t+i}, \dots, b_{(c_1 r-1)t+i})^T$ and $e^{(i)} = (e_i, e_{t+i}, e_{2t+i}, \dots, e_{(c_1 r-1)t+i})^T$ we have $b^{(i)} = \bar{G}^T w^{(i)} + e^{(i)}$. Since $w^{(i)} \in \{0, 1, \dots, s\}^r$, by Lemma 2, there is a polynomial time algorithm $\mathcal{A}$ such that if $wt(e^{(i)}) < (1/4-\epsilon_2/2)c_1 r$ then the algorithm returns $w^{(i)}$. Otherwise, the algorithm returns some $u^{(i)} \in \{0,1,\dots,2s\}^r$. Let $J := \{j_1, j_2, \dots, j_m\} \subseteq \{1,2,\dots,t\}$ such that $wt(e^{(i)}) > (1/4-\epsilon_2/2)c_1 r$ for $i \in J$ and $wt(e^{(i)}) \le (1/4-\epsilon_2/2)c_1 r$ for $i \notin J$. Since

$$\sum_{i=1}^{t} wt(e^{(i)}) = wt(e) \le k',$$

we have

$$m \le \frac{k'}{(1/4-\epsilon_2/2)c_1 r} = \left(\frac{1}{4} - \epsilon_1\right) t.$$

We run the algorithm $\mathcal{A}$ for each $b^{(i)}$ and obtain some vector $z^{(i)}$. By Lemma 2, $z^{(i)}$ is some vector $u^{(i)} \in \{0,1,\dots,2s\}^r$ if $i \in J$ and $z^{(i)} = w^{(i)}$ if $i \notin J$. Let

$a^{(i)} = (z_i^{(1)}, \ldots, z_i^{(t)})^T$ and $g^{(i)} = (w_i^{(1)} - z_i^{(1)}, \ldots, w_i^{(t)} - z_i^{(t)})^T$ for $i = 1, 2, \ldots, r$. Since $a^{(i)} + g^{(i)} = Mv^{(i)}$ we have $a^{(i)} = Mv^{(i)} - g^{(i)}$. Since for every i we have $wt(g^{(i)}) \leq m \leq (1/4 - \epsilon_1)t$ the vectors $v^{(i)}$, $i = 1, 2, \ldots, r$ can be found in polynomial time by the algorithm $\mathcal{ES}$.

4 Conclusion and Open Problems

In this paper we studied the coin weighing problem when some fraction of the weighings are incorrect or unknown. We give the tight bound 1/4 for the fraction for the case of incorrect answers and 1/2 for the case of unknown answers. We then give an upper and lower bounds for the number of weighings. There is a constant gap between the upper bound in Theorem 2 and the lower bound in Theorem 3. It is interesting to close this gap.

We then give a polynomial time algorithm with $O(n/\log\log n)$ weighings that detects the counterfeit coins when 1/16 fraction (respectively 1/4 fraction) of the weighing are incorrect (respectively, unknown). Two open problems arise. The first is to find a polynomial time algorithm with a better number of weighings. The second is to find a polynomial time algorithm with sublinear number of weighings that can detect better fraction of incorrect and unknown weighings.

References

1. Aigner, M.: Combinatorial Search. John Wiley and Sons (1988)
2. Alon, N., Asodi, V.: Learning a Hidden Subgraph. SIAM J. Discrete Math. 18(4), 697–712 (2005)
3. Biglieri, E., Györfi, L.: Multiple Access Channels Theory and Practice Volume 10 NATO Security through Science Series - D: Information and Communication Security (April 2007)
4. Bruneau, L., Germinet, F.: On the singularity of random matrices with independent entries. Proc. Amer. Math. Soc. 137, 787–792 (2009)
5. Bshouty, N.H.: Optimal Algorithms for the Coin Weighing Problem with a Spring Scale. In: Conference on Learning Theory (2009)
6. Bshouty, N.H., Mazzawi, H.: Toward a Deterministic Polynomial Time Algorithm with Optimal Additive Query Complexity. In: Hliněný, P., Kučera, A. (eds.) MFCS 2010. LNCS, vol. 6281, pp. 221–232. Springer, Heidelberg (2010)
7. Bshouty, N.H., Mazzawi, H.: Algorithms for the Coin Weighing Problems with the Presence of Noise. ECCC, TR11-124
8. Cantor, D.: Determining a set from the cardinalities of its intersections with other sets. Canadian Journal of Athematics 16, 94–97 (1962)
9. Cheng, J., Kamoi, K., Watanabe, Y.: User Identification by Signature Code for Noisy Multiple-Access Adder Channel. In: ISIT (2006)
10. Cantor, D., Mills, W.: Determining a Subset from Certain Combinatorial Properties. Canad. J. Math. 18, 42–48 (1966)
11. Chang, S.C., Weldon, E.J.: Coding for T-user multiple access channels. IEEE Transactions on Information Theory 25(6), 684–691 (1979)
12. Cheng, J., Watanabe, Y.: A Multiuser k-Ary Code for the Noisy Multiple-Access Adder Channel. IEEE Transactions on Information Theory 47, 6 (2001)

13. Cheng, J., Watanabe, Y.: Affine Code for T-User Noisy Multiple Access Adder Channel. IEICE Trans. Fundamentals E83-A(3) (2000)
14. Choi, S., Han Kim, J.: Optimal Query Complexity Bounds for Finding Graphs. In: STOC, pp. 749–758 (2008)
15. Cheng, J., Kamoi, K., Watanabe, Y.: User Identification by Signature Code for Noisy Multiple-Access Adder Channel. In: IEEE International Symposium on Information Theory, pp. 1974–1977 (2006)
16. Du, D., Hwang, F.K.: Combinatorial group testing and its application. Series on applied mathematics, vol. 3. World Science (1993)
17. Danev, D., Laczay, B., Ruszinkó, M.: Multiple Access Adder Channel. Multiple Access Channels - Theory and Practice, pp. 26–53. IOS Press (2007)
18. Erdös, Rényi, A.: On two problems of information theory. Publ. Math. Inst. Hung. Acad. Sci. 8, 241–254 (1963)
19. Grebinski, V., Kucherov, G.: Optimal Reconstruction of Graphs Under the Additive Model. Algorithmica 28(1), 104–124 (2000)
20. Grebiniski, V., Kucherov, G.: Reconstructing a hamiltonian cycle by querying the graph: Application to DNA physical mapping. Discrete Applied Mathematics 88, 147–165 (1998)
21. Grebinski, V.: On the Power of Additive Combinatorial Search Model. In: Hsu, W.-L., Kao, M.-Y. (eds.) COCOON 1998. LNCS, vol. 1449, pp. 194–203. Springer, Heidelberg (1998)
22. Indyk, P., Ruzic, M.: Near-Optimal Sparse Recovery in the L1 Norm. In: FOCS 2008, pp. 199–207 (2008)
23. Khachatrian, G.K., Martirossian, S.S.: Codes for T-user Noiseless Adder Channel. Problems of Control and Information Theory 16, 187–192 (1987)
24. Komlós, J.: On the determinant of matrices. Studia. Sci. Math. Hungar. 2, 7–21 (1967)
25. Laczay, B.: Coding for the Multiple Access Adder Channel (2003)
26. Lindström, B.: On a combinatorial problem in number theory. Canad. Math. Bull. 8, 477–490 (1965)
27. Lindström, B.: On a combinatorial detection problem II. Studia Scientiarum Mathematicarum Hungarica 1, 353–361 (1966)
28. Lindström, B.: On Möbius functions and a problem in combinatorial number theory. Canad. Math. Bull. 14(4), 513–516 (1971)
29. Lindström, B.: Determining subsets by unramified experiments. In: Srivastava, J.N. (ed.) A Survey of Statistical Designs and Linear Models, pp. 407–418. North Holland, Amsterdam (1975)
30. Li, M., Vitányi, P.M.B.: Combinatorics and Kolmogorov Complexity. In: Structure in Complexity Theory Conference, pp. 154–163 (1991)
31. Moser, L.: The second moment method in combinatorial analysis. In: Combinatorial Structure and their Applications, pp. 283–384. Gordon and Breach (1970)
32. Pippenger, N.: An Informtation Theoretic Method in Combinatorial Theory. J. Comb. Theory, Ser. A 23(1), 99–104 (1977)
33. Pippenger, N.: Bounds on the performance of protocols for a multiple-access broadcast channel. IEEE Transactions on Information Theory 27(2), 145–151 (1981)
34. Roth, R.M.: Introduction to Coding Theory. Cambridge University Press, Cambridge (2006)
35. Soderberg, S., Shapiro, H.S.: A combinatory detection problem. American Mathematical Monthly 70, 1066–1070 (1963)
36. Wilson, J.H.: Error-Correcting Codes for a T-User Binary Adder Channel. IEEE Transactions of Information Theory 34(4) (1988)

Information Complexity versus Corruption and Applications to Orthogonality and Gap-Hamming*

Amit Chakrabarti, Ranganath Kondapally, and Zhenghui Wang

Department of Computer Science, Dartmouth College Hanover, NH 03755, USA
{ac,rangak,zhenghui}@cs.dartmouth.edu

Abstract. Three decades of research in communication complexity have led to the invention of a number of techniques to lower bound randomized communication complexity. The majority of these techniques involve properties of large submatrices (rectangles) of the truth-table matrix defining a communication problem. The only technique that does not quite fit is information complexity, which has been investigated over the last decade. Here, we connect information complexity to one of the most powerful "rectangular" techniques: the recently-introduced smooth corruption (or "smooth rectangle") bound. We show that the former subsumes the latter under *rectangular* input distributions.

As an application, we obtain an optimal $\Omega(n)$ lower bound on the information complexity—under the *uniform distribution*—of the so-called orthogonality problem (ORT), which is in turn closely related to the much-studied Gap-Hamming-Distance problem (GHD). The proof of this bound is along the lines of recent communication lower bounds for GHD, but we encounter a surprising amount of additional technical detail.

Keywords: Communication Complexity, Information Complexity, Corruption, Gap Hamming, Orthogonality.

1 Introduction

The basic, and most widely-studied, notion of communication complexity deals with problems in which two players—Alice and Bob—engage in a communication protocol designed to "solve a problem" whose input is split between them. The communication problem is modeled by a function $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$. As is often the case, we are most interested in lower bounds.

Lower Bound Techniques and the Odd Man Out. The preeminent textbook in the field remains that of Kushilevitz and Nisan [19], which covers the basics as well as several advanced topics and applications. Scanning that textbook, one finds a number of lower bounding techniques, i.e., techniques for proving lower bounds on $\mathrm{D}(f)$ and $\mathrm{R}(f)$, the deterministic and randomized (respectively) communication complexities of f. Some of the more important techniques are the fooling set technique, log rank, discrepancy and corruption.[1] Research postdating the publication of the book has produced

* Work supported in part by NSF Grant IIS-0916565.

[1] Though the corruption technique is discussed in Kushilevitz and Nisan, the term "corruption" is due to Beame et al. [4]. The technique has also been called "one-sided discrepancy" and "rectangle method" [18] by other authors.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 483–494, 2012.
© Springer-Verlag Berlin Heidelberg 2012

a number of other such techniques, including the factorization norms method [21], the pattern matrix method [24], the partition bound and the smooth corruption[2] bound [12]. Notably, all of these techniques ultimately boil down to a fundamental fact called the *rectangle property*. One way of stating it is that each *fiber* of a deterministic protocol, defined as a maximal set of inputs $(x,y) \in \mathscr{X} \times \mathscr{Y}$ that result in the same communication transcript, is a combinatorial rectangle in $\mathscr{X} \times \mathscr{Y}$. The aforementioned lower bound techniques ultimately invoke the rectangle property on a protocol that computes f; for randomized lower bounds, Yao's minimax lemma also comes into play.

One recent technique is an odd man out: namely, *information complexity*, which was formally introduced by Chakrabarti et al. [8], generalized in subsequent work [2,16,3], though its ideas appear in the earlier work of Ablayev [1] (see also Saks and Sun [22]). Here, one defines an *information cost* measure for a protocol that captures the "amount of information revealed" during its execution, and then considers the resulting complexity measure $\mathrm{IC}(f)$, for a function f. A precise definition of the cost measure admits a few variants, but all of them quite naturally lower bound the corresponding communication cost. The power of this technique comes from a natural direct sum property of information cost, which allows one to easily lower bound $\mathrm{IC}(f)$ for certain well-structured functions f. Specifically, when f is a "combination" of n copies of a simpler function g, one can often scale up a lower bound on $\mathrm{IC}(g)$ to obtain $\mathrm{IC}(f) \geq \Omega(n\,\mathrm{IC}(g))$. The burden then shifts to lower bounding $\mathrm{IC}(g)$, and at this stage the rectangle property is invoked, *but on protocols for g, not f*.

Lower bounding $\mathrm{R}(f)$ via a lower bound on $\mathrm{IC}(f)$ has the nice consequence that one obtains a *direct sum theorem* for free: that is, we obtain the bound $\mathrm{R}(f^n) \geq \Omega(n\,\mathrm{IC}(f))$ as an almost immediate corollary. We shall be more precise about this in Section 2.

Rectangular versus Informational Methods. It is natural to ask how, quantitatively, these numerous lower bounding techniques relate to one another. One expects the various "rectangular" techniques to relate to one another, and indeed several such results are known [18,21,12]. Our first theorem relates the "informational" technique to one of the most powerful rectangular techniques, with respect to randomized communication complexity.

Theorem 1. *Let ρ be a rectangular*[3] *input distribution for a communication problem $f : \mathscr{X} \times \mathscr{Y} \to \mathscr{Z}$. Then, with respect to ρ, for small enough errors ε, the information complexity bound $\mathrm{IC}^{\rho}_{\varepsilon}(f)$ is asymptotically as good as the smooth corruption bound $\mathrm{scb}^{\rho}_{400\varepsilon,\varepsilon}(f)$ with error parameter 400ε and perturbation parameter ε. That is, there exist absolute positive constants b, c such that $\mathrm{IC}^{\rho}_{\varepsilon}(f) \geq c\,\mathrm{scb}^{\rho}_{400\varepsilon,\varepsilon}(f) - b$.*

The above is only an informal statement. The terminology of the theorem is precisely defined in Section 2 and the theorem itself is fully formalized as Theorem 4.

Independent of our work, Kerenidis et al. [17] have recently proved a more general result: namely, they have shown a similar asymptotic relation between $\mathrm{IC}^{\rho}(f)$ and $\mathrm{scb}^{\rho}(f)$ for an *arbitrary* input distribution ρ. Despite this, we believe that our proof of

[2] Jain and Klauck [12] used the term "smooth rectangle bound", but we shall prefer the more descriptive term "corruption" to "rectangle" throughout this article.

[3] Some authors use the term "product distribution" for what we call rectangular distributions.

Theorem 1 remains interesting, since it uses only elementary combinatorial and information theoretic properties of protocol transcripts, and proceeds along intuitive lines. In contrast, the proof in [17] is more technical in two ways. First, it uses the sophisticated compression techniques applied to protocol trees as in Barak et al. [3] and Braverman and Weinstein [5]. Second, it relies on the specifics of the linear programming formulation of the smooth corruption bound, whereas we work solely with the intuitive variant.

Another result in the same spirit as ours is due to Braverman and Weinstein [5]: it lower bounds information complexity by discrepancy. This result is incomparable with ours, because on the one hand discrepancy is a weaker technique than corruption, but on the other hand there is no restriction on the input distribution.

Information Complexity of Orthogonality and Gap-Hamming. The APPROXIMATE-ORTHOGONALITY problem is a communication problem defined on inputs in $\{-1,1\}^n \times \{-1,1\}^n$ by the Boolean function

$$\mathrm{ORT}_{b,n}(x,y) = 1, \text{ if } |\langle x,y\rangle| \leq b\sqrt{n}, \text{ and } \mathrm{ORT}_{b,n}(x,y) = -1, \text{ otherwise.}$$

Here, b is to be thought of as a constant parameter. This problem arose naturally in Sherstov's work on the Gap-Hamming Distance (GHD) problem [25]. This latter problem is defined by the partial Boolean function

$$\mathrm{GHD}_n(x,y) = -1, \text{ if } \langle x,y\rangle \leq -\sqrt{n}, \text{ and } \mathrm{GHD}_n(x,y) = 1, \text{ if } \langle x,y\rangle \geq \sqrt{n}.$$

The Gap-Hamming problem has attracted plenty of attention over the last decade, starting from its formal introduction in Indyk and Woodruff [11] in the context of data stream lower bounds, leading up to a recent flurry of activity that has produced three different proofs [7, 27, 25] of an optimal lower bound $\mathrm{R}(\mathrm{GHD}_n) = \Omega(n)$. In some recent work, Woodruff and Zhang [28] identify a need for strong lower bounds on $\mathrm{IC}(\mathrm{GHD})$, to be used in direct sum results. We now attempt to address such a lower bound.

At first sight, these problems appear to be ideally suited for a lower bound via information complexity: they are quite naturally combinations of n independent communication problems, each of which gives Alice and Bob a single input bit each. One feels that the uniform input distribution ought to be hard for them for the intuitive reason that a successful protocol cannot afford to ignore $\omega(\sqrt{n})$ of the coordinates of x and y, and must therefore convey $\Omega(1)$ information per coordinate for at least $\Omega(n)$ coordinates. However, turning this intuition into a formal proof is anything but simple.

Here, we prove an optimal $\Omega(n)$ lower bound on $\mathrm{IC}^{\mu}(\mathrm{ORT}_{b,n})$ under μ, the uniform input distribution on $\{-1,1\}^n \times \{-1,1\}^n$. This is a consequence of Theorem 1 above, but there turns out to be a surprising amount of work in lower bounding $\mathrm{scb}^{\mu}(\mathrm{ORT})$. Our theorem involves the function $\overline{\Phi}$, the tail of the standard normal distribution:

$$\overline{\Phi}(x) := \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-x^2/2} dx.$$

Theorem 2. *For large enough constants b, the corruption bound $\mathrm{cb}_{\theta}^{1,\mu}(\mathrm{ORT}_{b,n}) = \Omega(n)$, where $\theta = \overline{\Phi}(2.01b)$. Hence, by Theorem 1, we have $\mathrm{IC}^{\mu}_{\theta/400}(\mathrm{ORT}_{b,n}) = \Omega(n)$.*

Again, the terminology is precisely defined in Section 2 and the theorem is fully formalized as Theorem 6. As it turns out, a slight strengthening of the parameter θ in

the above theorem would give us the result $\mathrm{IC}^{\mu}_{\theta'}(\mathrm{GHD}_n) = \Omega(n)$. This is because the following result—stated somewhat imprecisely for now—connects the two problems.

Theorem 3. *With* $\theta = \overline{\Phi}(1.99b)$*, we have* $\mathrm{scb}^{1,\mu}_{400\theta,\theta}(\mathrm{GHD}_n) = \Omega(\mathrm{cb}^{1,\mu}_{400\theta}(\mathrm{ORT}_{b,n})) - O(\sqrt{n})$ *for large constants b. By Theorem 1,* $\mathrm{IC}^{\mu}_{\theta}(\mathrm{GHD}_n) = \Omega(\mathrm{cb}^{1,\mu}_{400\theta}(\mathrm{ORT}_{b,n})) - O(\sqrt{n})$.

We note that Chakrabarti and Regev [7] state that their lower bound technique for $\mathrm{R}(\mathrm{GHD}_n)$ can be captured within the smooth rectangle bound framework. While this is true in spirit, there is a significant devil in the details (see Section 4): their technique does not yield a good lower bound on $\mathrm{scb}^{1,\mu}_{\varepsilon,\delta}(\mathrm{GHD}_n)$ for the *uniform* distribution μ.

These theorems suggest a natural follow-up conjecture: there exists a constant ε such that $\mathrm{IC}^{\mu}_{\varepsilon}(\mathrm{GHD}_n) = \Omega(n)$. This remains open despite the very recent work of Kerenidis et al. [17], which does not touch $\mathrm{IC}^{\mu}(\mathrm{GHD})$.

Direct Sum. A direct sum theorem states that solving m independent instances of a problem requires about m times the resources that solving a single instance does. It could apply to a number of models of computation, with "resources" interpreted appropriately. For our model of two-party communication, it works as follows. For a function $f : \mathcal{X} \times \mathcal{Y} \to \{-1,1\}$, let $f^m : \mathcal{X}^m \times \mathcal{Y}^m \to \{-1,1\}^m$ denote the function given by

$$f^m(x_1,\ldots,x_m,y_1,\ldots,y_m) \;=\; (f(x_1,y_1),\ldots,f(x_m,y_m)).$$

Notice that f^m is not a Boolean function. We will define $\mathrm{R}(f^m)$ to be the randomized communication complexity of the task of outputting a vector $(z_1,\ldots,z_m)$ such that for each $i \in [m]$, we have $f(x_i,y_i) = z_i$ with high probability. Then, a direct sum theorem for randomized communication complexity would say that $\mathrm{R}(f^m) = \Omega(m \cdot \mathrm{R}(f))$. Whether or not such a theorem holds for a general f is a major open question in the field.

Information complexity, by its very design, provides a natural approach towards proving a direct sum theorem. Indeed, this was the original motivation of Chakrabarti et al. [8] in introducing information complexity; they proved a direct sum theorem for randomized *simultaneous-message* and *one-way* complexity, for functions f satisfying a certain "robustness" condition. Still using information complexity, Jain et al. [14] proved a direct sum theorem for bounded-round randomized complexity, when f is hard under a product distribution. Recently, Barak et al. [3] used information complexity, together with a *protocol compression* approach, to mount the strongest attack yet on the direct sum question for $\mathrm{R}(f)$, for all f: they show that $\mathrm{R}(f^m) \approx \Omega(\sqrt{m}\,\mathrm{R}(f))$, where the "$\approx$" ignores logarithmic factors.

One consequence of our work here is a simple proof of a direct sum theorem for randomized communication complexity for functions whose hardness is captured by a smooth corruption bound (which in turn subsumes corruption, discrepancy and smooth discrepancy [12]) under a rectangular distribution. This includes the well-studied INNER-PRODUCT function, and thanks to our Theorem 5, it also includes ORT. Of course, using the very recent result of Kerenidis et al. [17], the rectangularity constraint can be removed, which lets one capture additional important functions such as DISJOINTNESS.

We note that the protocol compression approach [3] gives a strong direct sum result for distributional complexity under rectangular distributions, but still not as strong as ours because their result contains a not-quite-benign polylogarithmic factor. We say more about this in Section 4.

Comparison with Direct Product. Other authors have considered a related, yet different, concept of direct *product* theorems. A strong direct product theorem (henceforth, SDPT) says that computing f^m with a correctness probability as small as $2^{-\Omega(m)}$—but more than the trivial guessing bound—requires $\Omega(m\,\mathrm{R}(f))$ communication, where "correctness" means getting *all* m coordinates of the output right. It is known that SDPTs do not hold in all situations [23], but do hold for (generalized) discrepancy [20,26], an especially important technique in lower bounding quantum communication. A recent manuscript offers an SDPT for bounded-round randomized communication [13].

SDPTs may look stronger than direct sum theorems,[4] but are in fact incomparable. A protocol could conceivably achieve low error on each coordinate of $f^m(x_1,\ldots,x_m, y_1,\ldots,y_m)$ while also having zero probability of getting the entire m-tuple right.

2 Preliminaries

Consider a (partial) function $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}_*$, where $\mathcal{X},\mathcal{Y},\mathcal{Z}$ are nonempty finite sets and $\mathcal{Z}_* := \mathcal{Z} \cup \{*\}$. We consider f to be undefined on an input $(x,y) \in \mathcal{X} \times \mathcal{Y}$ if $f(x,y) = *$. An important special case is $\mathcal{X} = \mathcal{Y} = \{-1,1\}^n$ and $\mathcal{Z} = \{-1,1\}$, where n is a large integer. We can interpret such a function f as a *communication problem* wherein Alice receives an input $x \in \mathcal{X}$, Bob receives an input $y \in \mathcal{Y}$, and the players must communicate according to a *protocol* P to come up with a value $z \in \mathcal{Z}$ that is hopefully equal to $f(x,y)$. When $f(x,y) = *$, P is deemed correct on (x,y) regardless of what P outputs. The sequence of messages exchanged by the players when executing P on input (x,y) is called the *transcript* of P on that input, and denoted $P(x,y)$. We require that the transcript be a sequence of bits, and end with (a binary encoding of) the agreed-upon output. We denote the output corresponding to a transcript t by $\mathrm{out}(t)$: thus, the output of P on input (x,y) is $\mathrm{out}(P(x,y))$.

Our protocols will, in general, be randomized protocols with a public coin as well as a private coin for each player. When we disallow the public coin, we will explicitly state that the protocol is private-coin. Notice that $P(x,y)$ is a random string, even for a fixed input (x,y). For a real quantity $\varepsilon \geq 0$, we say that P computes f with ε error if $\forall x,y : \Pr[f(x,y) \neq * \ \wedge\ \mathrm{out}(P(x,y)) \neq f(x,y)] \leq \varepsilon$, the probability being with respect to the randomness used by P and the input distribution. We define the cost of P to be the worst case length of its transcript, $\max|P(x,y)|$, where we maximize over all inputs (x,y) and over all possible outcomes of the coin tosses in P. Finally, the ε-error randomized communication complexity of f is defined by $\mathrm{R}_\varepsilon(f) = \min\{\mathrm{cost}(P) : P \text{ computes } f \text{ with error } \varepsilon\}$. In case $\mathcal{Z} = \{-1,1\}$, we also put $\mathrm{R}(f) = \mathrm{R}_{1/3}(f)$.

For random variables A,B,C, we use notations of the form $\mathrm{H}(A)$, $\mathrm{H}(A \mid C)$, $\mathrm{H}(AB)$, $\mathrm{I}(A : B)$, and $\mathrm{I}(A : B \mid C)$ to denote entropy, conditional entropy, joint entropy, mutual information, and conditional mutual information respectively. For discrete probability

[4] Some authors interpret "direct sum" as requiring correctness of the entire m-tuple output with high probability. Under this interpretation, direct product theorems indeed subsume direct sum theorems. Our definition of direct sum is arguably more natural, because under our definition, we at least have $\mathrm{R}(f^m) = O(m\,\mathrm{R}(f))$ always.

distributions λ, μ, we use $\mathrm{D}_{\mathrm{KL}}(\lambda \parallel \mu)$ to denote the relative entropy (a.k.a., informational divergence or Kullback-Leibler divergence) from λ to μ using logarithms to the base 2. These standard information theoretic concepts are well described in a number of textbooks, e.g., Cover and Thomas [9].

Let λ be an input distribution for f, i.e., a probability distribution on $\mathscr{X} \times \mathscr{Y}$. We say that λ is a *rectangular distribution* if we can write it as a tensor product $\lambda = \lambda_1 \otimes \lambda_2$, where λ_1, λ_2 are distributions on $\mathscr{X}, \mathscr{Y}$ respectively. Now consider a general λ and let $(X,Y) \sim \lambda$ be a random input for f drawn from this joint distribution. We define the λ-information-cost of the protocol P to be $\mathrm{icost}^{\lambda}(P) = \mathrm{I}(XY : P(X,Y) \mid R)$, where R denotes the public randomness used by P. This cost measure gives us a different complexity measure called the ε-error *information complexity* of f, under λ: $\mathrm{IC}^{\lambda}_{\varepsilon}(f) = \inf\{\mathrm{icost}^{\lambda}(P) : P \text{ computes } f \text{ with error } \varepsilon\}$. We note that in the terminology of Barak et al. [3], the above quantity would be called the *external* information complexity, as opposed to the *internal* one, which is based on the cost function $\mathrm{I}(X : P(X,Y), R \mid Y) + \mathrm{I}(Y : P(X,Y), R \mid X)$. As they show, the two cost measures coincide under a rectangular input distribution. Since our work only concerns rectangular distributions, this internal/external distinction is not important to us.

It is easy to see (and by now well-known) that information complexity under *any* input distribution lower bounds randomized communication complexity.

Fact 1. *For every input distribution λ and error ε, we have $\mathrm{R}_{\varepsilon}(f) \geq \mathrm{IC}^{\lambda}_{\varepsilon}(f)$.* □

Corruption and Smooth Corruption. Let $f : \mathscr{X} \times \mathscr{Y} \to \mathscr{Z}_*$ define a communication problem. Pick a particular $z \in \mathscr{Z}$. Call a set $S \subseteq \mathscr{X} \times \mathscr{Y}$ *rectangular* if $S = S_1 \times S_2$, where $S_1 \subseteq \mathscr{X}$, $S_2 \subseteq \mathscr{Y}$. Following Beame et al. [4], we say that S is ε-error z-monochromatic for f under λ if $\lambda(S \setminus (f^{-1}(z) \cup f^{-1}(*))) \leq \varepsilon\, \lambda(S)$. We then define

$$\varepsilon\text{-mono}^{z,\lambda}(f) = \max\{\lambda(S) : S \text{ is rectangular and } \varepsilon\text{-error } z\text{-monochromatic}\}, \tag{1}$$

$$\mathrm{cb}^{z,\lambda}_{\varepsilon}(f) = -\log(\varepsilon\text{-mono}^{z,\lambda}(f)), \tag{2}$$

$$\mathrm{scb}^{z,\lambda}_{\varepsilon,\delta}(f) = \max\{\mathrm{cb}^{z,\lambda}_{\varepsilon}(g) : g \in \mathscr{Z}_*^{\mathscr{X}\times\mathscr{Y}}, \Pr_{(X,Y)\sim\lambda}[f(X,Y) \neq g(X,Y)] \leq \delta\}. \tag{3}$$

The quantities $\mathrm{cb}^{z,\lambda}_{\varepsilon}(f)$ and $\mathrm{scb}^{z,\lambda}_{\varepsilon,\delta}(f)$ are called the corruption bound and the smooth corruption bound respectively, under the indicated choice of parameters: we call ε the *error parameter* and δ the *perturbation parameter*. One can go on to define bounds independent of z and λ by appropriately maximizing over these two parameters. We note that Jain and Klauck [12] use somewhat different notation: what we have called scb above is the logarithm of (a slight variant of) the quantity that they call the "natural definition of the smooth rectangle bound" and denote $\widetilde{srec}$.

What justifies calling these quantities "bounds" is that they can be shown to lower bound $\mathrm{R}_{\varepsilon'}(f)$ for sufficiently small $\delta, \varepsilon, \varepsilon'$, under a mild condition on λ. It is clear that $\mathrm{scb}^{z,\lambda}_{\varepsilon,\delta}(f) \geq \mathrm{cb}^{z,\lambda}_{\varepsilon}(f)$, so we mention only the stronger result, that involves the smooth corruption bound.

Fact 2 (Jain and Klauck [12]). *Let $f : \mathscr{X} \times \mathscr{Y} \to \mathscr{Z}_*$, $z \in \mathscr{Z}$ and distribution λ on $\mathscr{X} \times \mathscr{Y}$ be such that $\lambda(f^{-1}(z)) \geq 1/3$. Then there is an absolute constant $c > 0$ such that, for a sufficiently small constant ε, we have $\mathrm{R}_{\varepsilon}(f) \geq c \cdot \mathrm{scb}^{z,\lambda}_{5\varepsilon,\varepsilon/2}(f)$.* □

The constant $1/3$ above is arbitrary and can be parametrized, but we avoid doing this to keep things simple. The proof of the above fact is along the expected lines: an application of (the easy direction of) Yao's minimax lemma, followed by a straightforward estimation argument applied to the rectangles of the resulting deterministic protocol. Note that we never have to involve the linear-programming-based smooth rectangle bound as defined by Jain and Klauck.

3 Information Complexity versus Corruption

We sketch the proof of our first theorem.

Theorem 4 (Precise restatement of Theorem 1). *Suppose we have a function $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}_*$, a rectangular distribution ρ on $\mathcal{X} \times \mathcal{Y}$, and a value $z \in \mathcal{Z}$ satisfying $\rho(f^{-1}(z)) \geq 3/20$. Let $\varepsilon, \varepsilon'$ be reals with $0 \leq 384\varepsilon \leq \varepsilon' < 1/4$. Then*

$$\mathrm{IC}^{\rho}_{\varepsilon}(f) \geq \tfrac{1}{400}\,\mathrm{scb}^{z,\rho}_{\varepsilon',\varepsilon}(f) - \tfrac{1}{50} = \Omega\big(\mathrm{scb}^{z,\rho}_{\varepsilon',\varepsilon}(f)\big) - O(1)\,.$$

Let ρ be an input distribution for a communication problem, let P be a protocol for the problem, and let t be a transcript of P. We define $\sigma_t = \sigma_t(\rho)$ to be the distribution $(\rho \mid P(X,Y) = t)$. We think of the relative entropy $\mathrm{D}_{\mathrm{KL}}(\sigma_t \parallel \rho)$ as a *distortion* measure for t: intuitively, if t conveys little information about the inputs, then this distortion should be low. The following lemma, based on Markov inequalities, makes this intuition precise. Notice that it does not assume that ρ is rectangular.

While handling partial functions, we write "$g(x,y) \neq z$" to actually denote the event $g(x,y) \neq z \wedge g(x,y) \neq *$ for $z \in \mathcal{Z}$, unless specified otherwise.

Lemma 1. *Let P be a private-coin protocol that computes $g : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}_*$ with error $\varepsilon < 1/500$. Let $z \in \mathcal{Z}$ and let ρ be an arbitrary distribution on $\mathcal{X} \times \mathcal{Y}$ with $\rho(g^{-1}(z)) \geq 3/20 - 1/500$. Then, there exists a ("low-distortion") transcript t of P such that $\mathrm{out}(t) = z$, $\mathrm{D}_{\mathrm{KL}}(\sigma_t \parallel \rho) \leq 50\,\mathrm{icost}^{\rho}(P)$, and $\Pr[g(X,Y) \neq z \mid T = t] \leq 8\varepsilon$, where $(X,Y) \sim \rho$ and $T = P(X,Y)$.* □

The third property in the above lemma is a low-error guarantee for the transcript t. We can show that the existence of such a transcript implies the existence of a "large" low-corruption rectangle, provided the input distribution ρ is rectangular: this is the only point in the proof that uses rectangularity. One has to be careful with the interpretation of "large" here: it means large under σ_t, and not ρ. However, later on we will add in the low-distortion guarantee of Lemma 1 to conclude largeness under ρ as well.

Lemma 2. *Let t be a transcript of a private-coin protocol P for $g : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}_*$. Let ρ be a rectangular distribution on $\mathcal{X} \times \mathcal{Y}$, $z \in \mathcal{Z}$, $(X,Y) \sim \rho$, $T = P(X,Y)$, and $\varepsilon \geq 0$. Suppose $\Pr[g(X,Y) \neq z \mid T = t] \leq \varepsilon$, then there exists a rectangle $L \subseteq \mathcal{X} \times \mathcal{Y}$ such that $\sigma_t(L) \geq 9/16$ and $\Pr[g(X,Y) \neq z \mid (X,Y) \in L] \leq 16\varepsilon$.* □

We need the (classical) Substate Theorem due to Jain, Radhakrishnan and Sen [15].

Fact 3 (Substate Theorem [15]). *For distributions λ_1, λ_2 on $\mathcal{X}$, with $\mathrm{D}_{\mathrm{KL}}(\lambda_1 \parallel \lambda_2) \leq d$, (where $d \geq 0$), for all $S \subseteq \mathcal{X}$, we have $\lambda_2(S) \geq \lambda_1(S)/2^{2+2/\lambda_1(S)+2d/\lambda_1(S)}$.* □

Proof of Theorem 4. Let P^* be a protocol for f achieving the ε-error information cost under ρ. By a standard averaging argument, we may fix the public randomness of P^* to obtain a private-coin protocol P that computes f with error 2ε, and has $\mathrm{icost}^\rho(P) \leq 2\,\mathrm{icost}^\rho(P^*)$. Let g be the function achieving the maximum in Eq. (3), the definition of the smooth corruption bound, with error parameter ε' and perturbation parameter ε. Then $\mathrm{scb}^{z,\rho}_{\varepsilon',\varepsilon}(f) = \mathrm{cb}^{z,\rho}_{\varepsilon'}(g)$ and P computes g with error $3\varepsilon \leq 1/500$. Furthermore,

$$\rho(g^{-1}(z)) \geq \rho(f^{-1}(z)) - \Pr_{(X,Y)\sim\rho}[f(X,Y) \neq g(X,Y)] \geq 3/20 - \varepsilon > 3/20 - 1/500\,.$$

By Lemma 1, there exists a transcript t of P with distortion at most $100\,\mathrm{icost}^\rho(P^*)$ and error at most $24\varepsilon \leq \varepsilon'/16$. Therefore, by Lemma 2, there exists a rectangle L such that $\sigma_t(L) \geq 9/16$ and $\Pr[g(X,Y) \neq z \mid (X,Y) \in L] \leq \varepsilon'$. The latter condition may be rewritten as $\rho(L \setminus (g^{-1}(z) \cup g^{-1}(*))) \leq \varepsilon'\rho(L)$, i.e., L is ε'-error z-monochromatic for g under ρ. Then, by the Substate Theorem, for every subset $S \subseteq \mathscr{X} \times \mathscr{Y}$, we have $\rho(S) \geq \sigma_t(S)/2^{2+2/\sigma_t(S)+2d/\sigma_t(S)}$, where $d := \mathrm{D}_{\mathrm{KL}}(\sigma_t \parallel \rho) < 100\,\mathrm{icost}^\rho(P^*)$ by the distortion bound. Taking S to be the above rectangle L, and noting that $\sigma_t(L) \geq 1/2$, we have $\rho(L) \geq 1/2^{4d+7}$. Since L is ε-error z-monochromatic, the definition of the corruption bound tells us that $\mathrm{cb}^{z,\rho}_{\varepsilon}(g) \leq -\log\rho(L) \leq 4d+7 < 400\,\mathrm{icost}^\rho(P^*)+7$, which completes the proof. □

4 Information Complexity of Orthogonality and Gap-Hamming

We now tackle Theorems 2 and 3. These results are closely connected with a few recent works, and are both conceptually and technically interesting in their own right. We refer the reader to the full version of this paper [6] for an important discussion on why their proofs take so much additional work.

For the remainder of this paper, μ_n will denote the uniform distribution on $\{-1,1\}^n \times \{-1,1\}^n$. We will almost always drop the subscript n and simply use μ.

The Orthogonality Problem. In order to lower bound $\mathrm{IC}(\mathrm{ORT}_{b,n})$, we now see that it suffices to lower bound $\mathrm{cb}^\lambda(\mathrm{ORT}_{b,n})$ for a *rectangular* λ. We make the most natural choice, picking $\lambda = \mu$, the uniform input distribution. Our proof is then heavily inspired by two recent proofs of an optimal $\Omega(n)$ lower bound on $\mathrm{R}(\mathrm{GHD}_n)$, namely those of Chakrabarti and Regev [7], and Sherstov [25]. At the heart of our proof is the following anti-concentration lemma, which says that when pairs (x,y) are randomly drawn from a large rectangle in $\{-1,1\}^n \times \{-1,1\}^n$, the inner product $\langle x,y\rangle$ cannot be too sharply concentrated around zero.

Lemma 3 (Anti-concentration). *Let n be sufficiently large, let $b \geq 66$ be a constant, and let $\varepsilon = \overline{\Phi}(2.01b)$. Then there exists $\delta > 0$ such that for all $A,B \subseteq \{-1,1\}^n$ with $\min\{|A|,|B|\} \geq 2^{n-\delta n}$, we have $\Pr_{(X,Y)\in_R A\times B}\left[\langle X,Y\rangle \notin [-b\sqrt{n}, b\sqrt{n}]\right] \geq \varepsilon$, where "$\in_R$" denotes "is chosen uniformly at random from".*

The proof of this anti-concentration lemma has several technical steps, and we outline this proof in Section 5. We can prove following theorem using this lemma.

Theorem 5 (Precise restatement of Theorem 2). *Let $b \geq 1/5$ be a constant. Then* $\mathrm{cb}_{\theta}^{1,\mu}(\mathrm{ORT}_{b,n}) = \Omega(n)$, *for* $\theta = \overline{\Phi}(2.01\max\{66,b\})$. *Hence,* $\mathrm{IC}^{\mu}_{\theta/400}(\mathrm{ORT}_{b,n}) = \Omega(n)$.

The Gap-Hamming Problem. We now address the issue of proving a strong lower bound on $\mathrm{IC}^{\mu}(\mathrm{GHD})$. Our idea is that, for large b, the function GHD_n is at least as "hard" as a function that is "close" to $\mathrm{ORT}_{b,n}$, under a uniform input distribution. To be precise, we have the following connection between GHD and ORT. Recall that μ_n is the uniform distribution on $\{-1,1\}^n \times \{-1,1\}^n$.

Theorem 6 (Precise restatement of Theorem 3). *Let n be sufficiently large, let $b \geq 100$ be a constant, and let $\overline{\Phi}(1.99b) \leq \theta \leq 1/1600$. Let $n' = n + \frac{1}{2}(1.99b-1)\sqrt{n}$. Then,*

$$\mathrm{scb}^{1,\mu_n}_{400\theta,\theta}(\mathrm{GHD}_n) = \Omega(\mathrm{cb}^{1,\mu_{n'}}_{400\theta}(\mathrm{ORT}_{b,n'})) - O(\sqrt{n}).$$

Invoking Theorem 4, we then have $\mathrm{IC}^{\mu_n}_{\theta}(\mathrm{GHD}_n) = \Omega(\mathrm{cb}^{1,\mu_{n'}}_{400\theta}(\mathrm{ORT}_{b,n'})) - O(\sqrt{n})$.

Remark. If we could strengthen Theorem 5 by showing that $\mathrm{cb}^{1,\mu}_{\varepsilon}(\mathrm{ORT}_{b,n}) = \Omega(n)$ with $\varepsilon = \overline{\Phi}(1.98b)$, for large b. Then the present theorem would give us $\mathrm{IC}^{\mu}_{\varepsilon/400}(\mathrm{GHD}_n) = \Omega(n)$, since $\varepsilon/400 > \overline{\Phi}(1.99b)$. This would resolve our conjecture about $\mathrm{IC}^{\mu}(\mathrm{GHD}_n)$.

Proof. Put $t = n' - n = \frac{1}{2}(1.99b-1)\sqrt{n}$. Consider the padding $(x,y) \in \{-1,1\}^n \longmapsto (x',y') \in \{-1,1\}^{n'}$ defined by $x' = (1,1,\ldots,1,x)$ and $y' = (-1,-1,\ldots,-1,y)$. Then we have $\langle x',y'\rangle = \langle x,y\rangle - t$. For $b' := 1.99b$, we can verify that $\langle x,y\rangle \in \left[-\sqrt{n}, b'\sqrt{n}\right] \implies \langle x',y'\rangle \in \left[-b\sqrt{n'}, b\sqrt{n'}\right]$.

Define $h(x,y) := \mathrm{GHD}_n(x,y)$, if $\langle x,y\rangle \leq b'\sqrt{n}$ and $h(x,y) := -\mathrm{GHD}_n(x,y)$, if $\langle x,y\rangle > b'\sqrt{n}$. We can verify that if R is ε-error 1-monochromatic for the partial function h under μ_n, then R' is ε-error 1-monochromatic for $\mathrm{ORT}_{b,n'}$ under $\mu_{n'}$, where $R' \subseteq \{-1,1\}^{n'} \times \{-1,1\}^{n'}$ is the rectangle obtained by padding each $(x,y) \in R$ as above. Hence, we have $\varepsilon\text{-mono}^{1,\mu_n}(h) \leq 2^{2t}\varepsilon\text{-mono}^{1,\mu_{n'}}(\mathrm{ORT}_{b,n'})$ and thus, $\mathrm{cb}^{1,\mu_n}_{\varepsilon}(h) \geq \mathrm{cb}^{1,\mu_{n'}}_{\varepsilon}\left(\mathrm{ORT}_{b,n'}\right) - 2t$.

By standard estimates of the tail of a binomial distribution [10], we have

$$\Pr_{(X,Y)\sim\mu_n}[h(X,Y) \neq \mathrm{GHD}_n(X,Y)] = \Pr_{(X,Y)\sim\mu_n}[\langle X,Y\rangle > b'\sqrt{n}] \leq \overline{\Phi}(b') = \overline{\Phi}(1.99b). \quad (4)$$

Therefore, $\mathrm{scb}^{1,\mu_n}_{\varepsilon,\theta}(\mathrm{GHD}_n) \geq \mathrm{cb}^{1,\mu_n}_{\varepsilon}(h) \geq \mathrm{cb}^{1,\mu_{n'}}_{\varepsilon}(\mathrm{ORT}_{b,n'}) - 2t$ with $\theta \geq \overline{\Phi}(1.99b)$. The proof is now completed by applying Theorem 4. □

5 Proof of the Anti-concentration Lemma

Finally, we turn to the most technical part of this work: a proof of our new anti-concentration lemma, stated as Lemma 3 earlier. We only outline the broad steps; details can be found in the full version of the paper [6].

Let us begin with some convenient notation. We denote the (density function of the) standard normal distribution on the real line $\mathbb{R}$ by γ. We also denote the standard n-dimensional Gaussian distribution by γ^n. For a set $A \subseteq \mathbb{R}^n$, we denote by $\gamma^n|_A$ the distribution γ^n conditioned on belonging to A.

The Setup. For a contradiction, we begin by assuming the negation of Lemma 3. That is, we assume that $\exists$ constant $b \geq 66$ such that $\forall\, \delta > 0$, $\exists A, B \subseteq \{-1,1\}^n$ such that

$$\min\{|A|,|B|\} \geq 2^{n-\delta n}, \text{ and} \tag{5}$$

$$\Pr_{(X,Y)\in_R A\times B}\left[\langle X,Y\rangle \notin [-b\sqrt{n}, b\sqrt{n}]\right] < \varepsilon := \overline{\Phi}(2.01b)\,. \tag{6}$$

We treat the sets A and B asymmetrically in the proof. Using the largeness of A, and appealing to a concentration inequality of Talagrand, we identify a subset $V \subseteq A$ consisting of $\lceil\sqrt{\delta}n\rceil$ vectors such that

(P1) the vectors in V are, in some sense, near-orthogonal; and
(P2) the quantity $\langle x,Y\rangle$, where $Y \in_R B$, is concentrated around zero for *each* $x \in V$, in the sense of (6).

This step is a simple generalization of the first part of Sherstov's argument in his proof that $\mathrm{R}(\mathrm{GHD}_n) = \Omega(n)$.

As for the set B, we consider its *Gaussian analogue* $\widetilde{B} := \{\tilde{y} \in \mathbb{R}^n : \mathrm{sign}(\tilde{y}) \in B\}$. Specifically, we focus on random variable $Q_x = \langle x,\widetilde{Y}\rangle/\sqrt{n}$, for an arbitrary $x \in V$ and $\widetilde{Y} \sim \gamma^n|_{\widetilde{B}}$. Even more specifically, we focus on the *escape probability* $p_x := \Pr[|Q_x| > (c+\alpha)b]$, where $c := \sqrt{2/\pi}$ and $\alpha > 0$ is a constant we shall fix later. To obtain a contradiction, we shall analyze p_x—for a suitable x—in two ways.

Upper Bounding the Escape Probability. On the one hand, property (P2) is a *concentration* statement for the random variable $\langle x,Y\rangle$. By the connection between Y and $\tilde{Y}$, we can show that this implies a certain concentration for Q_x for all $x \in V$. Specifically, we can upper bound p_x for an arbitrary $x \in V$.

For simplicity, we assume, w.l.o.g., that $x = (1,1,\ldots,1)$ so that $\langle x,y\rangle = \sum_{i=1}^n y_i$. This is legitimate because, if $x_i = -1$, we can flip x_i to 1 and y_i to $-y_i$ without changing $\langle x,y\rangle$. Recall that each coordinate $\widetilde{Y}_i$ of $\widetilde{Y}$ has the same distribution as $Y_i|W_i|$, where the variables $\{W_i\}$ are independent and each $W_i \sim \gamma$. Define $T := \langle x,Y\rangle/\sqrt{n} = \sum_{i=1}^n Y_i/\sqrt{n}$; so T is a *discrete* random variable. After some reordering of coordinates, we can rewrite

$$\sqrt{n}\,Q_x = \langle x,\widetilde{Y}\rangle = \left(|W_1| + |W_2| + \cdots + \left|W_{\frac{n}{2}+\frac{T\sqrt{n}}{2}}\right|\right) - \left(\left|W_{\frac{n}{2}+\frac{T\sqrt{n}}{2}+1}\right| + \cdots + |W_n|\right).$$

Each $|W_i|$ has a so-called *half normal* distribution. This is a well-studied distribution: in particular, for each i, we know that $\mathbb{E}\left[|W_i|\right] = \sqrt{2/\pi}$, $\mathrm{Var}\left[|W_i|\right] = 1 - 2/\pi$. The variables $\{W_i\}$ are independent and behave well enough for us to apply Lindeberg's version of the central limit theorem [10]: doing so tells us that as n grows, the distribution of Q_x converges to $\mathcal{N}(cT,\sigma^2)$, where $c = \sqrt{2/\pi}$ and $\sigma = \sqrt{1-2/\pi}$. Using the convergence and the property (P2) of T, we can easily prove the following lemma.

Lemma 4. *Recall that $c = \sqrt{2/\pi}$. Let $\sigma = \sqrt{1-2/\pi}$. For all $x \in V$, $\alpha > 0$, and sufficiently large n, we have $p_x \leq 4\overline{\Phi}(\alpha b/\sigma) + 4\overline{\Phi}(2.01b)$.* □

Lower Bounding the Escape Probability. On the other hand, arguing along the lines of Chakrabarti-Regev [7], we cannot have too much concentration along so many near-orthogonal directions (as Property (P2) suggests), because $\widetilde{B}$ is a "large" subset of $\mathbb{R}^n$. Specifically, the largeness of B implies that the relative entropy $\mathrm{D}_{\mathrm{KL}}(\gamma^n|_B \,\|\, \gamma^n)$ is small. This in turn implies that there exists a direction $x^* \in V$ for which the projection Q_{x^*} behaves quite similarly to a mixture of shifted standard normal variables (i.e., variances close to 1, but arbitrary means). We highlight that these variances are larger than σ, and this is what allows us to lower bound the escape probability.

This line of argument yields following lemma, proved in the full paper [6].

Lemma 5. *There exists some $x^* \in V$, such that for all $c, \alpha > 0$ and sufficiently large n, we have $p_{x^*} \geq \frac{1}{2}(1-\delta^{1/4})\left(\overline{\Phi}\left((c+\alpha)b/(1-4\sqrt{\delta})\right) - 2\delta^{1/8}\right)$.* □

To complete the proof of the anti-concentration lemma, we combine the lower bound with the upper bound for p_{x^*} to obtain

$$\frac{1-\delta^{1/4}}{2}\left(\overline{\Phi}\left(\frac{(c+\alpha)b}{1-4\sqrt{\delta}}\right) - 2\delta^{1/8}\right) \leq 4\overline{\Phi}\left(\frac{\alpha b}{\sigma}\right) + 4\overline{\Phi}(2.01b)\,.$$

The above inequality is supposed to hold for some constant $b \geq 66$, $c = \sqrt{2/\pi}$, $\sigma = \sqrt{1-2/\pi}$ and all constants $\alpha, \delta > 0$. However, if we set $\alpha = 2.01\sigma$, we can get a contradiction: as $\delta \to 0$, the left-hand side approaches $\frac{1}{2}\overline{\Phi}((c+2.01\sigma)b)$, whereas the right-hand side is $8\overline{\Phi}(2.01b)$. Plugging in the values of c and σ, we note that $c + 2.01\sigma < 2.01$. Therefore, if we choose δ small enough, we have a contradiction.

Acknowledgment. We are grateful to Ryan O'Donnell for an important technical discussion.

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., Kumar, R., Sivakumar, D.: An information statistics approach to data stream and communication complexity. J. Comput. Syst. Sci. 68(4), 702–732 (2004)
3. Barak, B., Braverman, M., Chen, X., Rao, A.: How to compress interactive communication. In: Proc. 41st Annual ACM Symposium on the Theory of Computing, pp. 67–76 (2010)
4. Beame, P., Pitassi, T., Segerlind, N., Wigderson, A.: A strong direct product theorem for corruption and the multiparty communication complexity of disjointness. Comput. Complexity 15(4), 391–432 (2006)
5. Braverman, M., Weinstein, O.: A Discrepancy Lower Bound for Information Complexity. In: Gupta, A., et al. (eds.) A. Gupta et al (Eds.): APPROX/RANDOM 2012. LNCS, vol. 7408, pp. 459–470. Springer, Heidelberg (2012)
6. Chakrabarti, A., Kondapally, R., Wang, Z.: Information complexity versus corruption and applications to orthogonality and gap-hamming. CoRR abs/1205.0968 (2012)
7. Chakrabarti, A., Regev, O.: An optimal lower bound on the communication complexity of GAP-HAMMING-DISTANCE. In: Proc. 43rd Annual ACM Symposium on the Theory of Computing, pp. 51–60 (2011)

8. 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)
9. Cover, T.M., Thomas, J.A.: Elements of Information Theory, 2nd edn. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ (2006)
10. Feller, W.: An Introduction to Probability Theory and its Applications. John Wiley, New York (1968)
11. Indyk, P., Woodruff, D.P.: Tight lower bounds for the distinct elements problem. In: Proc. 45th Annual IEEE Symposium on Foundations of Computer Science, pp. 283–289 (2003)
12. Jain, R., Klauck, H.: The partition bound for classical communication complexity and query complexity. In: Proc. 25th Annual IEEE Conference on Computational Complexity, pp. 247–258 (2010)
13. Jain, R., Pereszlényi, A., Yao, P.: A direct product theorem for bounded-round public-coin randomized communication complexity. CoRR abs/1201.1666 (2012)
14. Jain, R., Radhakrishnan, J., Sen, P.: A Direct Sum Theorem in Communication Complexity via Message Compression. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 300–315. Springer, Heidelberg (2003)
15. 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)
16. Jayram, T.S., Kumar, R., Sivakumar, D.: Two applications of information complexity. In: Proc. 35th Annual ACM Symposium on the Theory of Computing, pp. 673–682 (2003)
17. Kerenidis, I., Laplante, S., Lerays, V., Roland, J., Xiao, D.: Lower bounds on information complexity via zero-communication protocols and applications. Technical Report TR12-038, ECCC (2012)
18. Klauck, H.: Rectangle size bounds and threshold covers in communication complexity. In: Proc. 18th Annual IEEE Conference on Computational Complexity, pp. 118–134 (2003)
19. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)
20. Lee, T., Shraibman, A., Špalek, R.: A direct product theorem for discrepancy. In: Proc. 23rd Annual IEEE Conference on Computational Complexity, pp. 71–80 (2008)
21. Linial, N., Shraibman, A.: Lower bounds in communication complexity based on factorization norms. Rand. Struct. Alg. 34(3), 368–394 (2009); Preliminary version in Proc. 39th Annual ACM Symposium on the Theory of Computing, pp. 699–708 (2007)
22. Saks, M., Sun, X.: Space lower bounds for distance approximation in the data stream model. In: Proc. 34th Annual ACM Symposium on the Theory of Computing, pp. 360–369 (2002)
23. Shaltiel, R.: Towards proving strong direct product theorems. Comput. Complexity 12(1-2), 1–22 (2003)
24. Sherstov, A.A.: The pattern matrix method for lower bounds on quantum communication. In: Proc. 40th Annual ACM Symposium on the Theory of Computing, pp. 85–94 (2008)
25. Sherstov, A.A.: The communication complexity of gap hamming distance. Technical Report TR11-063, ECCC (2011)
26. Sherstov, A.A.: Strong direct product theorems for quantum communication and query complexity. In: Proc. 43rd Annual ACM Symposium on the Theory of Computing, pp. 41–50 (2011)
27. Vidick, T.: A concentration inequality for the overlap of a vector on a large set, with application to the communication complexity of the gap-hamming-distance problem. Technical Report TR11-051, ECCC (2011)
28. Woodruff, D.P., Zhang, Q.: Tight bounds for distributed functional monitoring. Technical Report (2011), http://arxiv.org/abs/1112.5153

An Explicit VC-Theorem for Low-Degree Polynomials

Eshan Chattopadhyay, Adam Klivans, and Pravesh Kothari

University of Texas at Austin

Abstract. Let $X \subseteq R^n$ and let $\mathcal{C}$ be a class of functions mapping $\mathbb{R}^n \to \{-1, 1\}$. The famous VC-Theorem states that a random subset S of X of size $O(\frac{d}{\epsilon^2} \log \frac{d}{\epsilon})$, where d is the VC-Dimension of $\mathcal{C}$, is (with constant probability) an ϵ-approximation for $\mathcal{C}$ with respect to the uniform distribution on X. In this work, we revisit the problem of constructing S explicitly. We show that for any $X \subseteq \mathbb{R}^n$ and any Boolean function class $\mathcal{C}$ that is uniformly approximated by degree k polynomials, an ϵ-approximation S can be be constructed deterministically in time $\mathsf{poly}(n^k, 1/\epsilon, |X|)$ provided that $\epsilon = \Omega\left(W \cdot \sqrt{\frac{\log |X|}{|X|}}\right)$ where W is the weight of the approximating polynomial. Previous work due to Chazelle and Matousek suffers an $d^{O(d)}$ factor in the running time and results in superpolynomial-time algorithms, even in the case where $k = O(1)$.

We also give the first hardness result for this problem and show that the existence of a $\mathsf{poly}(n^k, |X|, 1/\epsilon)$-time algorithm for deterministically constructing ϵ-approximations for circuits of size n^k for every k would imply that $\mathsf{P} = \mathsf{BPP}$. This indicates that in order to construct explicit ϵ-approximations for a function class $\mathcal{C}$, we should not focus solely on $\mathcal{C}$'s VC-dimension.

Our techniques use deterministic algorithms for discrepancy minimization to construct hard functions for Boolean function classes over *arbitrary* domains (in contrast to the usual results in pseudorandomness where the target distribution is uniform over the hypercube).

1 Introduction

The VC-Theorem is one of the most important results in Statistics and Machine Learning and gives a quantitative bound on the rate of convergence of an empirical estimate of the bias of every function in a Boolean concept class to its true bias. The rate depends on the well-known *VC-dimension* of a function class (we refer the reader to Vapnik [14] for background material), and for the purposes of this paper we state the theorem as follows:

Theorem 1 (VC Theorem [13], See also [14]). *Let $\mathcal{C}$ be a Boolean function class mapping $\mathbb{R}^n \to \{-1, 1\}$ and let d denote its VC-dimension. Let $X \subseteq \mathbb{R}^n$ be finite and let $\mathcal{U}_X$ be the uniform distribution on X. Let S be a set of points obtained by taking $O(\frac{d}{\epsilon^2} \log \frac{d}{\epsilon})$ random draws from $\mathcal{U}_X$ and let $\mathcal{U}_S$ be the uniform distribution over points in S. Then with probability at least $1/2$ over the choice of S, for every $c \in \mathcal{C}$*

$$|\Pr_{\mathcal{U}_S}[c(x) = 1] - \Pr_{\mathcal{U}_X}[c(x) = 1]| \leq \epsilon.$$

This bound on the sample size was improved to $O(\frac{d}{\epsilon^2})$ in [6].

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 495–504, 2012.
© Springer-Verlag Berlin Heidelberg 2012

In this paper we are concerned with the following question: given as input the set X and a bound on the VC-dimension d, can we construct S in deterministic polynomial time in $n, 1/\epsilon, |X|$, and d? We refer to the set S as an ϵ-approximation.

From the perspective of computational complexity (and in particular pseudorandomness), it is natural to try to understand the complexity of explicitly constructing ϵ-approximations. In computational geometry, this problem has been studied before, most notably in work due to Chazelle and Matousek [3]. The parameters d and n, however, were considered constants, and the main goal was to obtain algorithms with run-time linear in $|X|$. Their work has had applications for finding deterministic algorithms for solving low-dimensional linear programs.

In an impressive new paper by Feldman and Langberg [4], the authors prove that ϵ-approximations for the function class of *halfspaces* in n dimensions give new algorithms for constructing core-sets and subsequently yield new applications for a host of clustering problems. Their paper inspires the following challenging open problem:

Open Problem 1. *Given a finite set of points X in $\mathbb{R}^n$, construct an ϵ-approximation for the class of halfspaces in deterministic time* $\mathsf{poly}(n, |X|, 1/\epsilon)$.

Previous work in computational geometry on explicit constructions of ϵ-approximations requires an enumeration of all possible labelings induced by $\mathcal{C}$ on the set X and suffers a $d^{O(d)}$ factor in the running time where d is the VC-dimension of the function class. Since halfspaces in n dimensions have VC-dimension $n+1$, these results run in time $n^{\Omega(n)}$.

1.1 Our Contributions

Unfortunately, we were unable to make progress on the above open problem. It turns out, however, that it is challenging to explicitly construct ϵ-approximations even for very simple classes of Boolean functions such as conjunctions and constant-depth decision trees. Here we give the first explicit constructions of ϵ-approximations for these classes that run in time subexponential in the VC-dimension. For the case of constant-depth decision trees, we give a polynomial-time deterministic construction. Our most general positive result is the following:

Theorem 2 (ϵ-Approximation For Functions Approximated by Low Degree Polynomials). *Let $\mathcal{C}$ be any class of boolean functions on n inputs on any finite set $X \subseteq [-1,1]^n$ such that $\mathcal{C}$ is δ-uniformly approximated on X by polynomials of degree at most d and weight at most W. Then, there is an algorithm that constructs an $(\epsilon + \delta \log |X|)$-approximation for $\mathcal{C}$ of size* $\mathsf{poly}(W, k, \log n, \frac{1}{\epsilon})$ *and runs in time* $\mathsf{poly}(W, n^k, \frac{1}{\epsilon}) \cdot |X|$ *whenever $\epsilon = \Omega\left(W\sqrt{\frac{k \log n}{|X|}}\right)$. (The exact dependence on the parameters can be found in Corollary 6.)*

Remark 1. As in Chazelle and Matousek [3], we require a lower bound on ϵ in order to achieve non-trivial bounds on the size of the ϵ-approximation. Roughly speaking, if ϵ is chosen to be too small, then we allow ourselves to output the entire set X as the approximation.

Combining Theorem 2 with known results on uniformly approximation Boolean functions by polynomials, we obtain the following corollary for Boolean conjunctions:

Corollary 1 (Informal Statement; see Section 4 for precise bounds). *Let $X \subseteq \{-1,1\}^n$. Then there is a deterministic algorithm that constructs an ϵ-approximation for the class of Boolean conjunctions on n variables of size $n^{\tilde{O}(\sqrt{n})}/\epsilon^2$ with running time $n^{O(\sqrt{n})}/\epsilon^2 \cdot |X|$.*

Recall that the class of conjunctions on n literals has VC-dimension n, so previous work due to Chazelle and Matousek [3] would require time $\Omega(2^n)$ to produce such an approximation. We point out here, however, that the *size* of the ϵ-approximations output by Chazelle and Matousek will be much better than ours (in fact, they achieve the optimal $O(\frac{d}{\epsilon^2}\log\frac{d}{\epsilon})$ bound on the size of S).

For the class of constant-depth decision trees, we can obtain a polynomial-time, deterministic algorithm:

Corollary 2 (Informal Statement; see Section 4 for precise bounds). *Let $X \subseteq \{-1,1\}^n$ and let $\mathcal{C}$ be the class of depth k decision trees. Then there is a deterministic algorithm that constructs an ϵ-approximation for $\mathcal{C}$ of size $n^{O(k)}/\epsilon^2$ with running time $\frac{n^{O(k)}}{\epsilon^2} \cdot |X|$.*

Constant-depth decision trees have VC-dimension $\Omega(\log n)$, so previous work will require time $\Omega(n^{\log\log n})$ to produce ϵ-approximations.

1.2 Our Contributions: Hardness Result

Since we are allowed to run in time polynomial in $|X|$, it may seem that the problem of construction ϵ-approximations is easier or at least incomparable to the problem of building pseudorandom generators for particular target distributions (such as $\{-1,1\}^n$). We prove, however, that giving explicit constructions of ϵ-approximations in time polynomial in n and the VC-dimension d is at least as hard as proving $\mathsf{P} = \mathsf{BPP}$ (recall that polynomial-size circuits have polynomial VC-dimension):

Theorem 3. *Let $\mathcal{C}_k^m$ be the class of all boolean circuits of size at most m^k on m inputs (each such circuit computes a function from $\{-1,1\}^m \to \{-1,1\}$). Suppose, there exists an algorithm $\mathcal{B}^k$ for some $k > 1$, which takes as input $X \subseteq \{-1,1\}^m$ and computes an $\frac{1}{3}$-approximation for $\mathcal{C}_m^k$ on X of size at most $\frac{|X|}{2}$ in time $\mathsf{poly}(|X|, m^k)$. Then $\mathsf{P} = \mathsf{BPP}$.*

In fact, we can show that constructing ϵ-approximations even for *linear*-size circuits deterministically in polynomial-time would imply $\mathsf{P} = \mathsf{BPP}$. As far as we are aware, this is the first hardness result for the general problem of deterministically constructing ϵ-approximations and explains why Chazelle and Matousek's work suffers an $d^{O(d)}$ factor in the running time. We conclude that we must consider additional properties of a concept class for which we wish to build ϵ-approximations other than its VC-dimension.

Our hardness result does *not*, however, rule out polynomial-time, deterministic constructions of ϵ-approximations for restricted function classes such as halfspaces (in fact, the relationship between Open Problem 1 and the problem of constructing pseudorandom generators for halfspaces with respect to $\{-1,1\}^n$ remains unclear).

1.3 Our Approach

Previous approaches for constructing ϵ-approximations over general domains $X \subseteq \mathbb{R}^n$ due to Chazelle and Matousek [3] involve an enumeration of all possible labelings induced on X with respect to a concept class $\mathcal{C}$. Naively this results in an algorithm with time complexity $|X|^{O(d)}$ where d is the VC-dimension of $\mathcal{C}$. Chazelle and Matousek instead only enumerate labelings on small subsets of X and use an elaborate method of partitioning and merging these subsets to reduce the time complexity to $|X| \cdot d^{O(d)}$ while maintaining an optimal size ϵ-approximation.[1] We wish to avoid this sort of enumeration altogether and run in time subexponential or even polynomial in the VC-dimension (although one drawback of our results is that we do not achieve optimal size ϵ-approximations).

Inspired by results on pseudorandom generators, our approach generates *average-case hard functions* with respect to arbitrary domains for a fixed concept class $\mathcal{C}$. In many cases in the pseudorandomness literature, an average-case hard function with respect to the uniform distribution on $\{-1,1\}^n$ can be used to build a pseudorandom generator. Here we show how to directly translate average-case hard functions with respect $\mathcal{U}_X$ for an arbitrary X in order to generate an ϵ-approximation for X.

One advantage of this approach is that once we have generated an average-case hard function for a concept class $\mathcal{C}$, we also obtain hard functions for any class $\mathcal{C}'$ approximated in ℓ_1 by $\mathcal{C}$. As such, we focus on generating hard functions for low-degree polynomials, as they can approximate several interesting Boolean function classes.

The question remains– how do we deterministically obtain average-case hard functions with respect to arbitrary domains? Although this problem is well-understood in the complexity literature for the case of the uniform distribution over $\{-1,1\}^n$, very little is known for other domains (e.g., arbitrary subsets of the hypercube). In retrospect, it is not difficult to see that a suitably constructive proof of an upper-bound on the VC dimension of a concept class $\mathcal{C}$ will yield a worst-case hard function with respect to any domain.

Still, constructive proofs of upper-bounds on the VC-dimension of interesting function classes are hard to come by, and we require average-case, not worst-case hardness. To this end, we turn to well-studied tools for generating *low-discrepancy* colorings. We use a derandomized version of a low-discrepancy coloring for the class of monomials, and this coloring will correspond to an average-case hard function for polynomials.

Finally, we can combine our techniques with the work of Chazelle and Matousek to obtain deterministic algorithms with a run-time that is linear in $|X|$.

We do not include any proofs here. For the proofs and the details of the algorithm, the reader is referred to the full version [1].

2 Preliminaries

We will deal with uniform distributions on arbitrary finite sets X and bounded functions from X to $[-1,1]$. This normalization of the range is without loss of generality and just

[1] To reduce the time complexity to $d^{O(d)}$ Chazelle and Matousek require a *subsystem oracle* for the class $\mathcal{C}$. The existence of such an oracle is dependent on the concept class.

fixes the scale for our parameters. We will refer to such functions as just bounded functions and the corresponding classes as bounded function classes. The uniform distribution on X, denoted by $\mathcal{U}_X$ fixes our notions of correlations (inner products) and norms. We will also encounter classes of boolean valued functions (boolean functions from now) defined on *arbitrary* finite domains X that range over $\{-1,1\}$. We do not include the standard definitions of inner products, norms, uniform and ℓ_1-approximators and weight of the polynomial due to lack of space. The reader is referred to the full version [1] for these.

2.1 Discrepancy

We now provide the basic definitions from discrepancy theory. We will only concern ourselves with *combinatorial discrepancy* (discrepancy from now) here. For further details, the reader may consult [2, 7, 11].

Definition 1 (Discrepancy of a Set System). *Let $(X,\mathbb{S})$ be a set system with $\mathbb{S}=\{S_1,S_2,...,S_m\}$ and $|X|=n$. Let $\chi : X \to \{-1,1\}$ be a coloring of X. The discrepancy of χ with respect to any set S is defined as $\chi(S)=\sum_{x\in S}\chi(x)$. The discrepancy of the coloring χ with respect to the set system $(X,\mathbb{S})$ is defined as $disc[X,\mathbb{S}](\chi)=max_{S\in\mathbb{S}}|\chi(S)|$. The discrepancy of the set system $(X,\mathbb{S})$ is defined as $disc(X,\mathbb{S})=\min_{\chi:X\to\{-1,1\}} disc[X,\mathbb{S}](\chi)$.*

In our setting, we will deal with discrepancies of arbitrary bounded functions. This definition is just a simple generalization of the preceding definition of discrepancy of a set system. Computing a coloring required in this generalization is sometimes referred to as the *lattice approximation problem* (see for example [12]).

Definition 2 (Discrepancy of a Function Class). *Let $\mathcal{C}$ be a set of functions $c : X \to [-1,1]$ and let $\chi : X \to \{-1,1\}$ be a coloring of X. The discrepancy of χ with respect to the function c is defined as $\chi(c)=\sum_{x:c(x)\geq 0}\chi(x)\cdot c(x)$. The discrepancy of χ with respect to the class $\mathcal{C}$ on X is defined as $disc[X,\mathcal{C}](\chi)=\max_{c\in\mathcal{C}}|\chi(c)|$.*

Note that when the function class is the set of indicator functions of a set system $(X,\mathbb{S})$ we recover the definition of the discrepancy of a set system as in Definition 1.

A uniformly random coloring turns out to be a low discrepancy coloring.

Lemma 1 (Discrepancy of Random Coloring). *Let $\mathcal{C}$ be a bounded function class of m functions on domain X, $|X|=n$. Let $\chi : X \to \{-1,1\}$ be chosen uniformly and independent at random for every $x \in X$, i.e. $\Pr[\chi(x)=1]=\frac{1}{2}$ for every $x \in X$. Then with probability at least $\frac{1}{2}$, $disc[X,\mathcal{C}](\chi) \leq O(\sqrt{n\log m})$.*

For the case of set systems a simple derandomization by conditional expectations of the random coloring method described above yields a deterministic construction of a low discrepancy coloring. For bounded function classes, we can use Nisan's deterministic simulation [8, 9] to compute such a coloring deterministically (see [12]) in $\mathsf{poly}(m,n)$ time.

Lemma 2 (Deterministic Construction of Low Discrepancy Coloring [2, 12]). *Let $\mathcal{C}$ be a bounded function class of m functions on X. There exists a deterministic algorithm running in time $\mathsf{poly}(m,|X|)$ that produces a coloring with discrepancy at most $O(\sqrt{|X|\log m})$.*

2.2 Definition of ϵ-Approximations for Boolean Function Classes

We now define the idea of an ϵ-approximation discussed in the introduction. Note that this definition is only for the case of *boolean* function classes.

Definition 3 (ϵ-approximation). *Let $\mathcal{C}$ be a* boolean *function class on the domain X. An ϵ-approximation for $\mathcal{C}$ on X is a set $Y \subseteq X$ such that for every $c \in \mathcal{C}$,*

$$|\Pr_{x \sim \mathcal{U}_Y}[c(x) = 1] - \Pr_{x \sim \mathcal{U}_X}[c(x) = 1]| \leq \epsilon.$$

As we noted in Theorem 1, the famous VC Theorem shows that, for a class $\mathcal{C}$ on a finite domain X, a random subset of X of size $O(\frac{d}{\epsilon^2} \log \frac{d}{\epsilon})$ is an ϵ-approximation with constant probability.

2.3 Hard Functions and Discrepancy

Here we describe a simple connection between the idea of low discrepancy colorings and hard functions. This connection is almost an equivalence when the underlying class consists of boolean functions and is slightly more involved for the class of bounded functions. As we shall see, this translation both simplifies the proofs and facilitates our polynomial-approximation based approach.

We first show how an algorithm for constructing a low discrepancy coloring for a boolean function class yields a hard function for the class. This translation is immediate and just results in loss of a constant factor in the parameters.

Proposition 1 (Low Discrepancy $\Rightarrow$ Hard Function). *Let $\mathcal{C}$ be a class of bounded functions from X to $[-1, 1]$. Let $-C = \{-c : c \in C\}$ denote the class of all negated functions from $\mathcal{C}$. If $\chi : X \to \{-1, 1\}$ is a coloring of X with discrepancy at most $\epsilon|X|$ with respect to $\mathcal{C} \cup -\mathcal{C}$ then χ is 2ϵ-hard for $\mathcal{C}$ on X.*

It is now easy to see that for a class of m boolean functions on X, the algorithm for Lemma 2 yields a $2\sqrt{\frac{\log m}{|X|}}$-hard function for the class on X and runs in time $\mathsf{poly}(m, |X|)$.

Definition 4 (Absolute Value Class). *For any function $c : X \to [-1, 1]$, we define $|c| : X \to [0, 1]$ as the absolute value of c on X. That is, $|c|(x) = |c(x)|$ for every $x \in X$. For a class $\mathcal{C}$ of bounded functions on a finite set X, we denote by $\mathcal{C}^{abs}$ the class of functions defined as $\mathcal{C}^{abs} = \{|c| \mid c \in \mathcal{C}\}$.*

Proposition 2 (Hard Function $\Rightarrow$ Low discrepancy). *Let $\mathcal{C}$ be a class of bounded functions on X. Suppose $\chi : X \to \{-1, 1\}$ is ϵ-hard for the class $\mathcal{C} \cup \mathcal{C}^{abs}$ on X. Then $\chi : X \to \{-1, 1\}$ is a coloring of X for $\mathcal{C}$ with discrepancy at most $\epsilon|X|$.*

3 Constructing Hard Functions on Arbitrary Domains

In this section, we describe our constructions of hard functions for boolean function classes on arbitrary domains that are approximated by low weight linear combinations

of functions from another small class. We first show that if a class $\mathcal{C}$ has good approximations as low weight linear combinations of functions from a class $\mathbb{F}$, then one can construct a hard function for $\mathcal{C}$ on X by constructing a hard function for the class $\mathbb{F}$ on X. (Using Proposition 2 from Section 2.3 will give us a low discrepancy coloring from these hard functions). We start by introducing the notion of a (W, δ)-approximating class.

Definition 5 ((W, δ)-approximating class). *Let $\mathcal{C}$ be a class of bounded functions on X. A class of bounded functions $\mathbb{F}$ on X is a (W, δ)-approximating class for $\mathcal{C}$, if $\forall\, c \in \mathcal{C}$ $\exists$ reals α_i for $1 \leq i \leq r$ satisfying $\sum_{i=1}^{r} |\alpha_i| \leq W$ and r functions, $f_1, f_2, .., f_r \in \mathbb{F}$ such that $\mathbb{E}_{x \sim \mathcal{U}_X}[|c(x) - \sum_{i=1}^{r} \alpha_i f_i(x)|] \leq \delta$.*

We now show that if a class $\mathcal{C}$ has a (W, δ) approximating class $\mathbb{F}$ on X, then a hard function for $\mathbb{F}$ is a hard function for $\mathcal{C}$ on X.

Theorem 4 (Hard Functions Through Low Weight Approximators). *Let $\mathcal{C}$ be a class of boolean functions on an arbitrary finite set X. Suppose a class of bounded functions $\mathbb{F}$ is a (W, δ)-approximating class for $\mathcal{C}$ on X. If $\chi : X \to \{-1, 1\}$ is $\frac{\epsilon}{W}$-hard for $\mathbb{F} \cup \mathbb{1}$ on X then χ is $(\epsilon + \delta)$-hard for $\mathcal{C} \cup \mathbb{1}$ on X.*

Remark 2. The above theorem gives us both a hard function and a low discrepancy coloring for the class $\mathcal{C}$ (see Proposition 2).

Fix any finite set $X \subseteq [-1, 1]^n$. Consider the class of all monomials of degree at most k in n variables as functions on X (denoted by $\mathcal{M}_k$). These monomials form a class of size $O(n^k)$ of bounded functions. We can construct a hard function for the class of all monomials of degree at most k using the algorithm from Lemma 2. Thus we have the following result.

Lemma 3 (Hard Functions for Monomials). *Let $\mathcal{M}_k$ be the class of all monomials of degree at most k in n variables on $X \subseteq [-1, 1]^n$. Then the algorithm from Lemma 2 runs in time $O(n^k|X|)$ and produces a function $\chi : X \to \{-1, 1\}$ such that for every monomial $m \in \mathcal{M}_k$, $|\langle m, \chi \rangle| = O\left(\sqrt{\frac{k \log n}{|X|}}\right)$.*

Now, using Theorem 4 and Lemma 3 we can translate the hard function for monomials to a hard function for functions approximated by polynomials on arbitrary finite domains.

Corollary 3 (Hard Functions for Functions Approximated by Low Degree Polynomials). *Let $\mathcal{C}$ be a class of boolean functions on $[-1, 1]^n$ and $X \subseteq [-1, 1]^n$ a finite set such that for each $c \in \mathcal{C}$ there is a polynomial $p : \mathbb{R}^n \to \mathbb{R}$, of degree at most k and weight at most W which δ-approximates c in ℓ_1-norm on X. Then, there exists an algorithm that runs in time $\mathsf{poly}(n^k) \cdot |X|$ that constructs a function $\chi : X \to \{-1, 1\}$ that is $(\epsilon + \delta)$-hard for $\mathcal{C}$ on X where $\epsilon = O\left(W\sqrt{\frac{k \log n}{|X|}}\right)$.*

Hard Functions for Conjunctions. Our result for the class of all conjunctions (and disjunctions), is obtained from the existence of uniformly approximating polynomials of low degree and weight. Although [10] only talks about the degree of their approximating polynomial, a simple inspection of their proof (which uses Chebyshev polynomials of the first kind) shows the weight bound noted below.

Theorem 5 (Uniform Approximation on $\{0,1\}^n$ [10]). *Let $f : \{-1,1\}^n \to \{-1,1\}$ be any boolean conjunction (or disjunction). Then, for every $\delta > 0$, there exists a polynomial $p : \mathbb{R}^n \to \mathbb{R}$ of degree $O(\sqrt{n}\cdot(\log n + \log \frac{1}{\delta}))$ such that for all $x \in \{-1,1\}^n$, we have $|f(x) - p(x)| \leq \delta$. Further the weight of this polynomial is bounded by $n^{O(\sqrt{n}\cdot(\log n+\log \frac{1}{\delta}))}$.*

As a result of Theorem 5 and Corollary 3 we obtain the following construction of hard functions for the class of conjunctions which generalizes to all linear sized formulas of constant depth.

Corollary 4. *Let $\mathcal{C}$ denote the class of boolean conjunctions. There exists an algorithm that runs in time $n^{O(\sqrt{n}\cdot(\log n+\log \frac{1}{\epsilon}))}|X|$ and computes an ϵ-hard function for the class $\mathcal{C}$ for any $\epsilon = \frac{1}{\sqrt{|X|}} \cdot n^{\Omega(\sqrt{n}\log n)}$.*

Remark 3. The uniform approximation theorem in [10] shows that for the more general class of formulas F of bounded depth. Our result for constructing hard functions directly translates to this more general class. Details are deferred to the full version.

Hard Functions for Decision Trees

Constant depth decision trees are computed exactly by low degree polynomials.

Theorem 6. *Let $f : \{-1,1\}^n \to \{-1,1\}$ be a boolean function computed by a decision tree of depth k. Then f is* exactly *computed by a real valued polynomial of degree k and weight at most 2^k.*

This result is well known and for completeness a proof appears in the full version. As a corollary of Theorem 6 and Theorem 3 we obtain a hard function for the class of all bounded depth decision trees.

Corollary 5. *Let DT^k denote the class of all boolean functions on n inputs computed by depth k decision trees. Then for any $X \subseteq \{-1,1\}^n$, there exists a deterministic algorithm which runs in time $\mathsf{poly}(n^k) \cdot |X|$ and outputs an ϵ-hard function for the class DT^k where $\epsilon = O(2^k\sqrt{\frac{k\log n}{|X|}})$.*

4 ϵ-Approximation from Hard Functions

In this section we show how to construct ϵ-approximations from hard functions combining our techniques with those of Chazelle-Matousek's [3]. We defer a detailed discussion to the full version [1] where we describe the algorithm. Due to a lack of space, we only state our results here.

The following is our general result for constructing ϵ-approximations from hard functions based on [3].

Theorem 7. *Let $\mathcal{C}$ be a class of boolean functions. Suppose $\mathcal{A}$ is a deterministic subroutine that takes input any finite set X and runs in time $T(\mathcal{C}, |X|)$ to give a $g(n,\mathcal{C})\sqrt{\frac{\log|X|}{|X|}}$-hard function for $\mathcal{C}$ on X. Then, for every $\epsilon = \Omega\left(g(n,\mathcal{C})\sqrt{\frac{\log|X|}{|X|}}\right)$, there exists an algorithm that constructs an ϵ-approximation for $\mathcal{C}$ on X of size $O(\frac{g(n,\mathcal{C})^2}{\epsilon^2}\log\frac{g(n,\mathcal{C})}{\epsilon})$ and runs in time $O(T(\mathcal{C},K)\cdot|X|)$ for $K = O(\frac{g(n,\mathcal{C})^6}{\epsilon^2}\log\frac{g(n,\mathcal{C})}{\epsilon})$.*

Using this result with Theorem 3 we obtain the following construction of ϵ-approximation for function classes approximated by low degree polynomials .

Corollary 6 (ϵ-Approximation For Functions Approximated by Low Degree Polynomials). *Let $\mathcal{C}$ be any class of boolean functions on n inputs and $X \subseteq [-1,1]^n$ such that $\mathcal{C}$ is δ-uniformly approximated by the class real valued polynomials of degree at most k and weight at most W on X. Then, there exists an algorithm which combined with the hard function construction from Theorem 3, constructs an $(\epsilon + \delta \log |X|)$-approximation for $\mathcal{C}$ of size $O(\frac{W^2 k}{\epsilon^2} \log \frac{Wk}{\epsilon}) \cdot \mathsf{poly}(n^k)$ and runs in time $\mathsf{poly}(n^k) \cdot |X| \cdot \frac{W^6 k^3}{\epsilon^2} \log \frac{Wk}{\epsilon}$ for any $\epsilon = \Omega\left(W\sqrt{\frac{k \log n}{|X|}}\right)$.*

As decision trees are exactly computed by polynomials of degree k and weight 2^k (Theorem 6), using the above result we have:

Corollary 7 (ϵ-Approximation For Decision Trees). *Let DT^k be the class of all boolean functions on n inputs computed by decision trees of depth at most k and $X \subseteq \{-1,1\}^n$. Then, there exists an algorithm which combined with the hard function construction from Corollary 5, constructs an ϵ-approximation for DT^k of size $O(\frac{2^{2k}k^2}{\epsilon^2} \log \frac{k}{\epsilon})\mathsf{poly}(n^k)$ and runs in time $O(\frac{2^{6k}k^4}{\epsilon^2} \log \frac{k}{\epsilon}) \cdot \mathsf{poly}(n^k) \cdot |X|$ for any $\epsilon = \Omega\left(2^k\sqrt{\frac{k \log n}{|X|}}\right)$. Note that the size of the ϵ-approximation produced is $n^{O(k)}$ and time complexity is $n^{O(k)}|X|$ for this range of ϵ.*

Similarly, combining the construction in Corollary 6 with Corollary 4 we obtain the following construction ϵ-approximation for the class of boolean conjunctions.

Corollary 8 (ϵ-Approximations for Conjunctions). *Let $\mathcal{C}$ be the class of all boolean conjunctions on n inputs. Then, there exists an algorithm which combined with the hard function construction from Corollary 4, constructs an ϵ-approximation for $\mathcal{C}$ of size $\frac{n^{O(\sqrt{n}\cdot \log n)}}{\epsilon^2}$ and runs in time $n^{O(\sqrt{n}\cdot \log n)} \cdot |X|$ whenever $\epsilon = n^{\Omega(\sqrt{n}\cdot \log n)} \cdot \sqrt{\frac{\log n}{|X|}}$.*

5 Hardness of Computing ϵ-Approximations

In this section we show that an efficient deterministic algorithm to compute an ϵ-approximation (or a hard function) for the class of polynomial size circuits on arbitrary inputs $X \subseteq \{-1,1\}^n$ implies $\mathsf{P} = \mathsf{BPP}$.

Theorem 8 (Impagliazzo-Wigderson [5]). *Suppose there is a function $f : \{-1,1\}^n \to \{-1,1\}$ computable in time $2^{O(n)}$ but cannot be computed by any circuit of size at most $2^{\delta n}$ for some fixed $\delta > 0$. Then $\mathsf{P} = \mathsf{BPP}$.*

We first show the hardness of constructing a hard function.

Theorem 9 (Hardness of Constructing a Hard Function). *For any $k \geq 1$, let $\mathcal{C}_k^m$ be the class of all boolean circuits of size at most m^k on m inputs (each such circuit computes a function from $\{-1,1\}^m \to \{-1,1\}$). Suppose $\mathcal{A}^k$ is an algorithm that takes as input any $X \subseteq \{-1,1\}^m$ and returns a function $f : X \to \{-1,1\}$ such that for*

every $c \in \mathcal{C}_k^m$, $|\langle f, c\rangle| < 1$ *and runs in time* $\mathsf{poly}(|X|, m^k)$. *Then there exists a function* $L : \{-1,1\}^n \to \{-1,1\}$ *computable in time* $2^{O(n)}$ *such that for any* $0 < \delta < \frac{1}{k}$, *no circuit on* n *inputs of size at most* $2^{k\delta n}$ $(= 2^{(1-\beta)n}$ *for some* $\beta > 0)$ *can compute* L.

We now show that an algorithm for computing an ϵ-approximation of any non-trivial size for polynomial sized circuits over arbitrary domain X implies the existence of the algorithm $\mathcal{A}^k$ as in the statement of Theorem 9. Combined with Theorem 9 and 8 we have the required result.

Theorem 10. *Let* $\mathcal{C}_k^m$ *be the class of all boolean circuits of size at most* m^k *on* m *inputs (each such circuit computes a function from* $\{-1,1\}^m \to \{-1,1\}$*). Suppose, there exists an algorithm* $\mathcal{B}^k$ *for some* $k > 0$, *which takes as input* $X \subseteq \{-1,1\}^m$ *and computes an* $\frac{1}{3}$*-approximation for* $\mathcal{C}_m^k$ *on* X *of size at most* $\frac{|X|}{2}$ *in time* $\mathsf{poly}(|X|, m^k)$. *Then* $\mathsf{P} = \mathsf{BPP}$.

Acknowledgments. We thank the anonymous reviewers for their detailed comments to improve the presentation of the paper.

References

[1] Chattopadhyay, E., Klivans, A., Kothari, P.: An explicit vc-theorem for low degree polynomials. Full Version (2012)
[2] Chazelle, B.: The discrepancy method: randomness and complexity. Cambridge University Press, New York (2000)
[3] Chazelle, B., Matousek, J.: On linear-time deterministic algorithms for optimization problems in fixed dimension. J. Algorithms 21(3), 579–597 (1996)
[4] Feldman, D., Langberg, M.: A unified framework for approximating and clustering data. In: STOC, pp. 569–578 (2011)
[5] Impagliazzo, R., Wigderson, A.: P = BPP unless E has sub-exponential circuits: Derandomizing the XOR lemma (preliminary version). In: Proceedings of the 29th STOC, pp. 220–229. ACM Press (1996)
[6] Li, Y., Long, P.M., Srinivasan, A.: Improved bounds on the sample complexity of learning. Journal of Computer and System Sciences 62, 2001 (2000)
[7] Matousek, J.: Geometric Discrepancy: An Illustrated Guide (Algorithms and Combinatorics), 1st edn. Springer (1999)
[8] Nisan, N.: Pseudorandom generators for space-bounded computation. Combinatorica 12(4), 449–461 (1992)
[9] Nisan, N.: RL $\subseteq$ SC. In: STOC, pp. 619–623 (1992)
[10] O'Donnell, R., Servedio, R.A.: New degree bounds for polynomial threshold functions. In: STOC, pp. 325–334 (2003)
[11] Pach, J., Agrawal, P.: Combinatorial Geometry. Wiley-Interscience (October 1995)
[12] Sivakumar, D.: Algorithmic derandomization via complexity theory. In: IEEE Conference on Computational Complexity, p. 10 (2002)
[13] Vapnik, V.N., Chervonenkis, A.Y.: On the uniform convergence of relative frequencies of events to their probabilities. Theory of Probability and its Applications 16(2), 264–280 (1971)
[14] Vapnik, V.N.: Statistical learning theory. Wiley (1998)

Tight Bounds on the Threshold for Permuted k-Colorability

Varsha Dani[1], Cristopher Moore[1,2], and Anna Olson[3]

[1] Computer Science Department, University of New Mexico
[2] Santa Fe Institute
[3] Computer Science Department, University of Chicago

Abstract. If each edge (u, v) of a graph $G = (V, E)$ is decorated with a permutation $\pi_{u,v}$ of k objects, we say that it has a *permuted k-coloring* if there is a coloring $\sigma : V \to \{1, \ldots, k\}$ such that $\sigma(v) \neq \pi_{u,v}(\sigma(u))$ for all $(u, v) \in E$. Based on arguments from statistical physics, we conjecture that the threshold d_k for permuted k-colorability in random graphs $G(n, m = dn/2)$, where the permutations on the edges are uniformly random, is equal to the threshold for standard graph k-colorability. The additional symmetry provided by random permutations makes it easier to prove bounds on d_k. By applying the second moment method with these additional symmetries, and applying the first moment method to a random variable that depends on the number of available colors at each vertex, we bound the threshold within an additive constant. Specifically, we show that for any constant $\varepsilon > 0$, for sufficiently large k we have

$$2k \ln k - \ln k - 2 - \varepsilon \leq d_k \leq 2k \ln k - \ln k - 1 + \varepsilon \,.$$

In contrast, the best known bounds on d_k for standard k-colorability leave an additive gap of about $\ln k$ between the upper and lower bounds.

1 Introduction

We consider random graphs $G(n, m)$ with n vertices and m edges chosen uniformly without replacement. We give each edge (u, v) an arbitrary orientation, and then associate it with a uniformly random permutation $\pi_{u,v} \in S_k$, where S_k denotes the group of permutations of k objects. A *permuted k-coloring* of this decorated graph is a function $\sigma : V \to \{1, \ldots, k\}$ such that $\sigma(v) \neq \pi_{u,v}(\sigma(u))$ for all edges (u, v). For convenience we will sometimes reverse the orientation of an edge, and write $\pi_{v,u} = \pi_{u,v}^{-1}$ when u and v are distinct.

We conjecture that there is a sharp threshold for the existence of such a coloring in terms of the average degree $d = 2m/n$:

Conjecture 1. *For each $k \geq 3$ there is a constant d_k such that*

$$\lim_{n\to\infty} \Pr\left[G(n, m = dn/2) \text{ has a permuted } k\text{-coloring}\right] = \begin{cases} 1 & d < d_k \\ 0 & d > d_k \,, \end{cases}$$

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 505–516, 2012.
© Springer-Verlag Berlin Heidelberg 2012

where the probability space includes both the graph $G(n,m)$ and the set of permutations $\{\pi_{u,v}\}$. Moreover, we conjecture that d_k is also the threshold for standard graph k-colorability, which is the special case where $\pi_{u,v}$ is the identity permutation for all u, v:

Conjecture 2. *For the same d_k as in Conjecture 1,*

$$\lim_{n\to\infty} \Pr\left[G(n, m = dn/2) \text{ has a standard } k\text{-coloring}\right] = \begin{cases} 1 & d < d_k \\ 0 & d > d_k \end{cases},$$

Why might these two thresholds be the same? First note that if G is a tree, we can "unwind" the permutations on the edges, changing them all to the identity, by permuting the colors at each vertex. For any set of permutations $\{\pi_{u,v}\}$, this gives a one-to-one map from permuted colorings to standard colorings. Since sparse random graphs are locally treelike, for almost all vertices v we can do this transformation on a neighborhood of radius $\Theta(\log n)$ around v. The effect on v's neighborhood of the other vertices' colors is then the same as it would be in standard graph coloring, except that the colors on the boundary are randomly permuted.

In particular, suppose we choose a uniformly random coloring of a tree, erase the colors in its interior, and choose a new uniformly random coloring with the same boundary conditions. The *reconstruction threshold* is the degree d above which this new coloring retains a significant amount of information about the original coloring [8,21], and it is closely related to the clustering transition [19]. Since we can unwind the permutations on a tree, permuted k-colorability and standard k-colorability trivially have the same reconstruction threshold.

For another argument, consider the following alternate way to choose the permutations on the edges. First choose a uniformly random permutation π_u at each vertex u. Then, on each edge (u,v), let $\pi_{u,v} = \pi_u^{-1}\pi_v$. (Algebraically, $\pi_{u,v} : E \to S_k$ is the coboundary of $\pi_u : V \to S_k$.) Since $c(u) = \pi_{u,v}(c(v))$ if and only if $\pi_u(c(v)) = \pi_v(c(v))$, a local change of variables again gives a one-to-one map from permuted colorings to standard ones. Now note that choosing $\pi_{u,v}$ in this way yields a uniform joint distribution on any set of edges that does not include a cycle; for instance, on a graph of girth g the $\pi_{u,v}$ are $(g-1)$-wise independent and uniform. Since most cycles in a sparse random graph are long, we might hope that this distribution on the $\{\pi_{u,v}\}$ is the same, for all practical purposes, as the uniform distribution.

Finally, perhaps the most convincing argument for Conjecture 2 comes from statistical physics. Using cavity field equations to analyze the asymptotic behavior of message-passing algorithms such as belief propagation and survey propagation, we can derive thresholds for satisfiability or colorability [17,16,20], as well as other thresholds such as clustering, condensation, and freezing [23,13]. However, assuming that there is an equal density of vertices of each color, the cavity field equations for k-coloring in a random graph are identical to those for permuted k-coloring [14,22]: they simply express the fact that each edge (u,v) forbids u from taking a single color that depends on the color of v.

Thus if the physics picture is correct—and parts of it have been shown rigorously (e.g. [1,9,10])—then colorings and permuted colorings have the same "thermodynamics" on sparse random graphs, and hence the same thresholds for colorability, as well as for clustering, condensation, and freezing.

Conjecture 2 is attractive because it is easier, given current methods, to prove tight bounds on the threshold for permuted k-coloring than it is for standard k-coloring. First we recall a simple upper bound. Let X denote the number of permuted k-colorings. Since the permutations are chosen independently, the probability that any given coloring σ is proper is $(1-1/k)^m$. (Indeed, this is true of any multigraph with m edges.) Thus the expected number of colorings is

$$\mathbb{E}[X] = k^n(1-1/k)^m = \left[k(1-1/k)^{d/2}\right]^n . \tag{1}$$

This is exponentially small if $k(1-1/k)^{d/2} < 1$, in which case $X = 0$ with high probability by Markov's inequality. Thus

$$d_k \le \frac{2\ln k}{-\ln(1-1/k)} < 2k\ln k - \ln k\,. \tag{2}$$

Using the second moment method, we will prove a lower bound on d_k that is an additive constant below this upper bound. We also improve the upper bound, using a random variable that depends on the number of available colors at each vertex. Our results show that, for any constant $\varepsilon > 0$ and k sufficiently large,

$$2k\ln k - \ln k - 2 - \varepsilon \le d_k \le 2k\ln k - \ln k - 1 + \varepsilon\,.$$

In contrast, the best known lower bound on the threshold for standard k-colorability is roughly $\ln k$ below the first moment upper bound.

To simplify our arguments, we work in a modified random graph model $\tilde{G}(n,m)$ where the m edges are chosen uniformly with replacement, and the endpoints of each edge are chosen uniformly with replacement from the n vertices. As a consequence, both self-loops and multiple edges occur with nonzero probability. Note that, unlike standard k-colorability, a self-loop at a vertex v does not necessarily render the graph uncolorable: it simply means that $\sigma(v)$ cannot be a fixed point of the permutation on the loop, i.e., $\sigma(v) \neq \pi_{v,v}(\sigma(v))$.

However, if $\pi_{v,v}$ is the identity then coloring is impossible, and this occurs with probability $1/k!$. Similarly, if u and v have k edges between them, then with constant probability the permutations on these edges make a coloring impossible. As a consequence, the probability that $\tilde{G}(n,m)$ with random permutations has a permuted k-coloring is bounded below 1.

Our bounds proceed by showing that $\tilde{G}(n,m)$ is permuted-k-colorable with probability $\Omega(1)$ if d is sufficiently small, and is not permuted-k-colorable with high probability if d is sufficiently large. In the sparse case $m = O(n)$, $\tilde{G}(n,m)$ is simple with probability $\Omega(1)$, in which case it coincides with $G(n,m)$. Thus, assuming that Conjecture 1 is true, these values of d are bounds on the threshold d_k for $G(n,m)$.

The rest of the paper is organized as follows. In Section 2 we give our second moment lower bound. In Section 3 we give our upper bound, which uses a random variable that depends on the number of available colors, and which (roughly speaking) counts the number of clusters of colorings. In Section 4, we prove an isoperimetric inequality relevant to this random variable. The parts of our proofs that are "mere calculus" may be found in the full version.

2 The Second Moment Lower Bound

As in the simple first moment upper bound, let X denote the number of permuted k-colorings of a random multigraph $\tilde{G}(n,m)$ with uniformly random permutations on its edges. Applying the Cauchy-Schwarz inequality to the inner product $X \cdot \mathbf{1}_{X>0}$ gives

$$\Pr[X > 0] \geq \frac{\mathbb{E}[X]^2}{\mathbb{E}[X^2]} .$$

Our goal is to show that $\mathbb{E}[X^2]/\mathbb{E}[X]^2 = O(1)$, so that a permuted coloring exists with probability $\Omega(1)$, for a certain value of d. Assuming Conjecture 1, *i.e.,* that a threshold d_k exists, any such d is a lower bound on d_k.

Computing the second moment $\mathbb{E}[X^2]$ requires us to sum, over all pairs of colorings σ, τ, the probability $P(\sigma,\tau)$ that both σ and τ are proper. Since the edges of $\tilde{G}$ and their permutations are chosen independently, we have $P(\sigma,\tau) = p(\sigma,\tau)^m$, where $p(\sigma,\tau)$ is the probability that a random edge (u,v), with a random permutation π, is satisfied by both colorings. That is,

$$p(\sigma,\tau) = \Pr_{u,v,\pi}[\sigma(u) \neq \pi(\sigma(v)) \text{ and } \tau(u) \neq \pi(\tau(v))] .$$

For random constraint satisfaction problems where each variable takes one of two values, such as k-SAT or hypergraph 2-coloring [3,4,7], $p(\sigma,\tau)$ is a function $p(\zeta)$ just of the overlap between σ and τ, i.e., the fraction ζ of variables on which they agree. The second moment can then be bounded by maximizing a function of ζ, which is typically a simple calculus problem.

For pairs of k-colorings, however, $p(\sigma,\tau)$ depends on a k-by-k matrix of overlaps, where $\zeta_{i,j}$ is the fraction of vertices v such that $\sigma(v) = i$ and $\tau(v) = j$. Computing the second moment then requires us to bound a function of roughly k^2 variables, a difficult high-dimensional maximization problem. Achlioptas and Naor [6] used convexity arguments to bound this function on the Birkhoff polytope, showing that

$$d_k \geq 2(k-1)\ln(k-1) \approx 2k\ln k - 2\ln k .$$

This leaves an additive gap of about $\ln k$ between the upper and lower bounds. Note, however, that this bound is tight enough to determine, almost surely, the chromatic number $\chi(G)$ as a function of the average degree to one of two possible integers, namely k or $k+1$ where k is the smallest integer such that $2k\ln k > d$. Achlioptas and Moore extended these arguments to random regular graphs [5], determining $\chi(G)$ as a function of d to k, $k+1$, or $k+2$.

For permuted colorings, the second moment calculation is much easier. The random permutations create additional local symmetries, making $p(\sigma, \tau)$ a function only of the fraction ζ on which the two colorings agree. Thus we just have to maximize a function of a single variable. As a consequence, we can prove a lower bound on d_k that matches the upper bound (2) within an additive constant.

Theorem 3. *For any $\varepsilon > 0$, for sufficiently large k we have*

$$d_k > 2k \ln k - \ln k - 2 - \varepsilon \,. \tag{3}$$

Proof. To compute the second moment, we sum over all $k^n \binom{n}{k} (k-1)^{n-z}$ pairs of colorings that agree at z of the n vertices. We say that such a pair has overlap $\zeta = z/n$. Then

$$\mathbb{E}[X^2] = k^n \sum_{z=0}^{n} \binom{n}{z} (k-1)^{n-z} \, p(z/n)^m \,, \tag{4}$$

where $p(\zeta)$ is the probability that a random edge, with a random permutation, is satisfied by both colorings. Inclusion-exclusion gives

$$p(\zeta) = \zeta^2 \left(1 - \frac{1}{k}\right) + 2\zeta(1-\zeta) \left(1 - \frac{2}{k}\right) + (1-\zeta)^2 \left(1 - \frac{2}{k} + \frac{1}{k(k-1)}\right) .$$

Note that $p(1/k) = (1-1/k)^2$, corresponding to the fact that two independently random colorings typically have overlap $\zeta = 1/k + o(1)$.

We proceed as in [3]. Approximating (4) with an integral and using (1) gives

$$\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2} \sim \frac{1}{\sqrt{n}} \sum_{z=0}^{n} \mathrm{e}^{\phi(z/n)n} \sim \sqrt{n} \int_0^1 \mathrm{d}\zeta \, \mathrm{e}^{\phi(\zeta)n} \,, \tag{5}$$

where $\sim$ hides multiplicative constants, where

$$\phi(\zeta) = h(\zeta) + (1-\zeta)\ln(k-1) - \ln k + \frac{d}{2} \ln \frac{p(\zeta)}{(1-1/k)^2} \,,$$

and where $h(\zeta) = -\zeta \ln \zeta - (1-\zeta)\ln(1-\zeta)$ is the entropy function. Applying Laplace's method to the integral (5) then gives

$$\frac{\mathbb{E}[X^2]}{\mathbb{E}[X]^2} \sim \frac{\mathrm{e}^{\phi(\zeta_{\max})n}}{\sqrt{|\phi''(\zeta_{\max})|}} \,,$$

where $\zeta_{\max} = \operatorname{argmax}_{\zeta \in [0,1]} \phi(\zeta)$ is the global maximum of $\phi(\zeta)$, assuming that it is unique and that $\phi''(\zeta_{\max}) < 0$.

We have $\phi(1/k) = 0$, so if $\zeta_{\max} = 1/k$ and $\phi''(1/k) < 0$ then $\mathbb{E}[X^2]/\mathbb{E}[X]^2 = O(1)$ and $\Pr[X > 0] = \Omega(1)$. The proof is completed by the following lemma (see the full version for its proof):

Lemma 1. *For any constant $\varepsilon > 0$, if $d = 2k \ln k - \ln k - 2 - \varepsilon$ and k is sufficiently large, $\phi''(1/k) < 0$ and $\phi(\zeta) < 0$ for all $\zeta \neq 1/k$.* □

3 An Improved First Moment Upper Bound

In this section we apply the first moment method to a weighted random variable, and improve the upper bound (2) on d_k by a constant. Specifically, we will prove the following theorem:

Theorem 4. *For any $\varepsilon > 0$, for sufficiently large k we have*

$$d_k < 2k \ln k - \ln k - 1 + \varepsilon \,. \tag{6}$$

We define our random variable as follows. Every coloring (proper or not) of n vertices with k colors is an element of $[k]^n$ where $[k] = \{1, 2, \ldots, k\}$. Thus the set of colorings is an n-cube of side k, with a dimension for each vertex. The set of *proper* colorings is some subset of this cube, $S \subset [k]^n$. The classic first moment argument we reviewed above computes the expected number of proper (permuted) colorings, $X = |S|$. Here we define a new random variable, where each proper coloring is given a weight that depends on the "degree of freedom" at each vertex, i.e., the number of colors that vertex could take if the colors of all other vertices stayed fixed.

For any proper coloring σ, for each vertex v, let $c(\sigma, v)$ denote the number of colors available for v if its neighbors are colored according to σ. That is,

$$c(\sigma, v) = k - |\{\pi_{u,v}(\sigma(u)) \,|\, (u, v) \in E\}| \,.$$

Note that if there are multiple edges between u and v, they can each forbid v from taking a color. If v has a self-loop, we think of it as denying a color to itself, in each direction:

$$c(\sigma, v) = k - \left|\{\pi_{u,v}(\sigma(u)) \,|\, (u, v) \in E, u \neq v\} \cup \{\pi_{v,v}(\sigma(v)), \pi_{v,v}^{-1}(\sigma(v))\right| \,. \tag{7}$$

If σ is a proper coloring, we have $c(\sigma, v) \geq 1$ for all v, since v's current color $\sigma(v)$ is available. Let

$$w(\sigma) = \begin{cases} \prod_v \left(1/c(\sigma, v)\right) & \text{if } \sigma \text{ is proper} \\ 0 & \text{otherwise}\,, \end{cases}$$

and let

$$Z = \sum_{\sigma \in [k]^n} w(\sigma) \,.$$

Why this random variable? The expected number of colorings $\mathbb{E}[X]$ can be exponentially large, even above the threshold where $\Pr[X > 0]$ is exponentially small. But close to the threshold, solutions come in clusters, where some vertices are free to flip back and forth between several available colors. In a cartoon where each cluster is literally a subcube of $[k]^n$, a cluster containing a coloring σ contributes $\prod_v c(\sigma, v)$ to X, but only 1 to Z. Thus, roughly speaking, Z counts the number of clusters rather than the number of colorings. Since the sizes of the clusters vary, Z has smaller fluctuations than X does, and hence gives a tighter

upper bound on d_k. We note that "cluster counting" random variables of other sorts have been studied elsewhere, such as satisfying assignments with a typical fraction of free variables [9] and certain kinds of partial assignments [15].

In this same cartoon where clusters are subcubes, Z also counts the number of *locally maximal* colorings, i.e., those colorings where no vertex v can be flipped to a "higher" color $q > \sigma(v)$, since such colorings correspond to the highest corner of the cluster. Bounds on d_3 were derived by counting locally maximal 3-colorings in [2,12,11], culminating in $d_3 \leq 4.937$. The bounds we derive below are slightly weaker for $k = 3$, yielding $d_3 \leq 5.011$, since we treat the degrees of the vertices as independent rather than conditioning on the degree distribution. Nevertheless, our arguments are considerably simpler argument for general k.

Clearly $Z > 0$ if and only if $S \neq \varnothing$. However, in Section 4 we prove the following:

Lemma 2. *If $S \neq \varnothing$ then $Z \geq 1$.*

Applying Markov's inequality, we see that

$$\Pr[G \text{ has a permuted } k\text{-coloring}] = \Pr[Z \geq 1] \leq \mathbb{E}[Z],$$

and any d such that $\mathbb{E}[Z] < 1$ is an upper bound on the threshold d_k. Thus we will prove Theorem 4 by computing $\mathbb{E}[Z]$.

Given the symmetry provided by the random edge permutations, the expected weight $\mathbb{E}[w(\sigma)]$ of any given coloring is independent of σ. By linearity of expectation and the fact that any given σ is proper with probability $(1 - 1/k)^m$, we then have

$$\mathbb{E}[Z] = k^n \, \mathbb{E}[w(\sigma)] = k^n (1 - 1/k)^m \, \mathbb{E}[w(\sigma) \,|\, \sigma \text{ proper}].$$

Thus we are interested in the conditional expectation

$$\mathbb{E}[w(\sigma) \,|\, \sigma \text{ proper}] = \mathbb{E}\left[\prod_v \frac{1}{c(\sigma, v)} \,\middle|\, \sigma \text{ proper} \right].$$

Lemma 3. *Let σ be a permuted k-coloring. For any vertex v, the conditional distribution of the number of available colors $c(\sigma, v)$ is a function only of v's degree. In particular it does not depend on σ.*

Proof. For each neighbor u of v, there is a uniformly random permutation $\pi = \pi_{u,v}$ such that v is blocked from having the color $\pi(\sigma(u))$. Since σ is proper, we have $\pi(\sigma(u)) \neq \sigma(v)$. Subject to this condition, π is uniformly random among the permutations such that $\pi(\sigma(u)) \neq \sigma(v)$, and thus $\pi(\sigma(u))$ is uniformly random among the colors other than $\sigma(v)$. In particular, it does not depend on $\sigma(u)$.

We can think of the forbidden colors $\pi(\sigma(u))$ as balls, and the colors other than $\sigma(v)$ as bins. We toss $\deg v$ balls independently and uniformly into these $k - 1$ bins, one for each edge (u, v). Then $c(v)$ is the number of empty bins, plus one for $\sigma(v)$. □

Lemma 4. *Let σ be a permuted k-coloring, (u,v) an edge, and $\pi = \pi_{u,v}$ the associated random permutation. Then $\pi(\sigma(u))$ and $\pi^{-1}(\sigma(v))$ are independent, and are uniform over $[k] - \sigma(v)$ and $[k] - \sigma(u)$ respectively.*

Proof. Since σ is proper, the conditional distribution of π is uniform among all permutations such that $\pi(\sigma(u)) \neq \sigma(v)$ and $\pi^{-1}(\sigma(v)) \neq \sigma(u)$. For any pair of colors q, q' with $q \neq \sigma(v)$ and $q' \neq \sigma(u)$, there are exactly $(k-2)!$ permutations π such that $\pi(\sigma(u)) = q$ and $\pi^{-1}(\sigma(v)) = q'$. Thus all such pairs (q, q') are equally likely, and the pair $\big(\pi(\sigma(u)), \pi^{-1}(\sigma(v))\big)$ is uniform in $([k] - \sigma(v)) \times ([k] - \sigma(u))$. □

Note that Lemmas 3 and 4 apply even to self-loops. That is, if $\pi = \pi_{v,v}$ is uniformly random, then $\pi(\sigma(v))$ and $\pi^{-1}(\sigma(v))$ are independent and uniform in $[k] - \sigma(v)$. Thus a self-loop corresponds to two balls, each of which can forbid a color. Since a self-loop increases v's degree by 2, it has the same effect as two edges incident to v would have. Indeed, this is why we defined $c(\sigma, v)$ as in (7).

Now let $\{\deg v \mid v \in V\}$ denote the degree sequence of G. By Lemmas 3 and 4, the numbers of available colors $c(\sigma, v)$ at the vertices v are conditionally independent if their degrees are fixed. Thus

$$\begin{aligned}
\mathbb{E}[w(\sigma) \mid \sigma \text{ proper}, \{\deg v\}] &= \mathbb{E}\left[\prod_v \frac{1}{c(\sigma, v)} \,\middle|\, \sigma \text{ proper}, \{\deg v\} \right] \\
&= \prod_v \mathbb{E}\left[\frac{1}{c(\sigma, v)} \,\middle|\, \sigma \text{ proper}, \deg v \right] \\
&= \prod_v \sum_{c=1}^{k} \frac{Q(\deg v, k, c)}{c} \,,
\end{aligned}$$

where $Q(\deg v, k, c)$ denotes the probability that v has c available colors if it has $\deg v$; that is, the probability that if we toss $\deg v$ balls into $k-1$ bins, then $c-1$ bins will be empty. Thus

$$\mathbb{E}[w(\sigma) \mid \sigma \text{ proper}] = \mathop{\mathbb{E}}_{\{\deg v\}} \prod_v \sum_{c=1}^{k} \frac{Q(\deg v, k, c)}{c} \,,$$

where the expectation is taken over the distribution of degree sequences in $\tilde{G}(n, m)$.

The degree of any particular vertex in $\tilde{G}(n, m = dn/2)$ is asymptotically Poisson with mean d. The degrees of different vertices are almost independent, as the next lemma shows.

Lemma 5. *The joint probability distribution of the degree sequence of $\tilde{G}(n, m = dn/2)$ is the same as that of n independent Poisson random variables of mean d, conditioned on their sum being $2m$.*

Proof. We can generate $\tilde{G}(n, m = dn/2)$ as follows. There are n bins, one for each vertex. We throw $2m$ balls uniformly and independently into the bins, and

pair up consecutive balls to define the edges of the graph. The degree of each vertex is the number of balls in the corresponding bin. The joint distribution of these occupancies is the product of n independent Poisson distributions with mean d, conditioned on the total number of balls being $2m$ [18, Theorem 5.6]. □

The sum of n independent Poisson variables of mean d equals its mean $nd = 2m$ with probability $O(1/\sqrt{m}) = O(1/\sqrt{n})$. Conditioning on an event that holds with probability P increases the expectation by at most $1/P$, so

$$\begin{aligned}\mathbb{E}[w(\sigma) \,|\, \sigma \text{ proper}] &= \mathop{\mathbb{E}}_{\{\deg v\}} \prod_v \sum_{c=1}^{k} \frac{Q(\deg v, k, c)}{j} \\ &= O(\sqrt{n}) \left(\mathop{\mathbb{E}}_{\deg v} \sum_{c=1}^{k} \frac{Q(\deg v, k, c)}{c} \right)^n , \end{aligned} \tag{8}$$

where in the second line $\deg v$ is Poisson with mean d. Since our goal is to show that $\mathbb{E}[Z]$ is exponentially small, the $\sqrt{n}$ factor will be negligible.

Now consider a balls and bins process with $k-1$ bins. If the total number of balls is Poisson with mean d, then the number of balls in each bin is Poisson with mean $d/(k-1)$, and these are independent. The probability that any given bin is empty, i.e., that any given color $q \neq \sigma(v)$ is available, is

$$r = \mathrm{e}^{-d/(k-1)} .$$

The number of empty bins is binomially distributed as $\mathrm{Bin}(k-1, r)$, so

$$Q(\deg v, k, c) = \binom{k-1}{c-1} r^{c-1} (1-r)^{k-c} ,$$

and

$$\begin{aligned}\mathop{\mathbb{E}}_{\deg v} \sum_{c=1}^{k} \frac{Q(\deg v, k, c)}{c} &= \sum_{c=1}^{k} \frac{1}{c} \binom{k-1}{c-1} r^{c-1} (1-r)^{k-c} \\ &= \frac{1}{kr} \sum_{c=1}^{k} \binom{k}{c} r^c (1-r)^{k-c} \\ &= \frac{1}{kr} \left(1 - (1-r)^k\right) . \end{aligned}$$

Putting everything together, we have

$$\begin{aligned}\mathbb{E}[Z] &= k^n \left(1 - \frac{1}{k}\right)^m \mathbb{E}[w(\sigma) \,|\, \sigma \text{ proper}] \\ &= O(\sqrt{n})\, k^n \left(1 - \frac{1}{k}\right)^{dn/2} \left(\frac{1}{kr} \left(1 - (1-r)^k\right)\right)^n \\ &= O(\sqrt{n}) \left(1 - \frac{1}{k}\right)^{dn/2} \mathrm{e}^{dn/(k-1)} \left(1 - (1 - \mathrm{e}^{-d/(k-1)})^k\right)^n . \end{aligned}$$

Taking the logarithm and dividing by n, in which case we can ignore the polynomial term $\sqrt{n}$, yields the following function of d:

$$f(d) := \lim_{n\to\infty} \frac{\ln \mathbb{E}[Z]}{n} = \frac{d}{2} \ln\left(1 - \frac{1}{k}\right) + \frac{d}{k-1} + \ln\left(1 - \left(1 - \mathrm{e}^{-d/(k-1)}\right)^k\right) . \tag{9}$$

If $f(d) < 0$ then $\mathbb{E}[Z]$ is exponentially small, so any such d is an upper bound on d_k. The proof of Theorem 4 is now completed by the following Lemma, whose proof may be found in the full version.

Lemma 6. *For any constant $\varepsilon > 0$, if $d = 2k \ln k - \ln k - 1 + \varepsilon$ and k is sufficiently large, then $f(d) < 0$.*

4 An Isoperimetric Inequality

In this section we prove Lemma 2. It has nothing to do with colorings; it is simply a kind of isoperimetric inequality that applies to any subset of $[k]^n$. If $k = 2$, it is the classic isoperimetric inequality on the Boolean n-cube. That is, given $S \subseteq \{0,1\}^n$, for each $\sigma \in S$ let $\partial(\sigma)$ be the set of neighbors $\sigma' \in S$ that differ from σ on a single bit. Then $S \neq \varnothing$ if and only if $\sum_{\sigma\in S} 2^{-|\partial(\sigma)|} \geq 1$.

First, some notation. Let $V = \{1, \ldots, n\}$, and think of each element of $[k]^n$ as a function $\sigma : V \to [k]$. Let $S \subseteq [k]^n$. For each $\sigma \in S$ and $1 \leq v \leq n$, let $\partial_S(\sigma, v)$ denote the set of elements of S that are "neighbors of σ along the v axis," i.e., that agree with σ everywhere other than at v. That is,

$$\partial_S(\sigma, v) = \{\sigma' \in S \mid \forall u \neq v : \sigma'(u) = \sigma(u)\} .$$

Let $c_S(\sigma, v)$ denote the number of such neighbors,

$$c_S(\sigma, v) = |\partial_S(\sigma, v)| ,$$

and define the weight function w_S as follows:

$$w_S(\sigma) = \begin{cases} \prod_v (1/c_S(\sigma, v)) & \text{if } \sigma \in S \\ 0 & \text{if } \sigma \notin S . \end{cases}$$

Then define the weight of the entire set as

$$Z(S) = \sum_{\sigma\in[k]^n} w_S(\sigma).$$

Lemma 7. *If $S \neq \varnothing$ then $Z(S) \geq 1$.*

Proof. If $S = [k]^n$, then $w_S(\sigma) = 1/k^n$ and $Z(S) = 1$. Thus our goal will to enlarge S until $S = [k]^n$, showing that $Z(S)$ can only decrease at each step. For a given σ and v, let $\mathrm{Cyl}_v(\sigma)$ denote the set of $\tau \in [k]^n$ that we can obtain by letting $\sigma(v)$ vary arbitrarily:

$$\mathrm{Cyl}_v(\sigma) = \{\tau \in [k]^n \mid \tau(u) = \sigma(u) \text{ for all } u \neq v\} .$$

In particular, $\mathrm{Cyl}_v(\sigma) = \mathrm{Cyl}_v(\sigma')$ if and only if $\sigma' \in \partial_S(\sigma, v)$. Similarly, let $\mathrm{Cyl}_v(S)$ be the "thickening" of S along the v axis,

$$\mathrm{Cyl}_v(S) = \bigcup_{\sigma \in S} \mathrm{Cyl}_v(\sigma)\,.$$

We claim that this thickening can only decrease Z. That is, for any $S \neq \varnothing$ and any v,

$$Z(S) \geq Z(\mathrm{Cyl}_v\, S)\,.$$

To see this, let $T = \mathrm{Cyl}_v(S)$. Each $\sigma \in S$ contributes $c_v(\sigma, v)$ times to the union $\bigcup_{\sigma \in S} \mathrm{Cyl}_v(\sigma)$, so

$$Z(T) = \sum_{\sigma \in S} \frac{1}{c_S(\sigma, v)}\, w_T(\mathrm{Cyl}_v(\sigma))\,. \tag{10}$$

Since each $\sigma \in T$ has $c_T(\sigma, v) = k$, each $\tau \in \mathrm{Cyl}_v(\sigma)$ has $c_T(\tau, u) \geq c_S(\sigma, u)$ for all $u \neq v$, and $|\mathrm{Cyl}_v(\sigma)| = k$, we have

$$w_T(\mathrm{Cyl}_v(\sigma)) \leq k\, \frac{c_S(\sigma, v)}{k}\, w_S(\sigma) = c_S(\sigma, v)\, w_S(\sigma)\,.$$

Combining this with (10) gives

$$Z(T) = \sum_{\sigma \in S} \frac{1}{c_S(\sigma, v)}\, w_T(\mathrm{Cyl}_v(\sigma)) \leq \sum_{\sigma \in S} w_S(\sigma) = Z(S)\,.$$

Now let $T_0 = S$, and $T_v = \mathrm{Cyl}_v(T_{v-1})$ for $1 \leq v \leq n$. Then $T_n = [k]^n$, and

$$Z(S) = Z(T_0) \geq Z(T_1) \geq \cdots \geq Z(T_n) = 1$$

which completes the proof. □

Finally, we conclude this section with

Proof of Lemma 2. Let S be the set of permuted k-colorings. The number of available colors $c(\sigma, v)$ we defined in Section 3 is almost identical to $c_S(\sigma, v)$ as defined in Lemma 7. The only difference is that in Section 3, if v has a self-loop then it forbids two of its own colors, namely $\pi_{v,v}(\sigma(v))$ and $\pi_{v,v}^{-1}(v)$. Removing these self-loops can only increase $c(\sigma, v)$ and thus decrease Z, so if $S \neq \varnothing$ then $Z \geq 1$ by Lemma 7.

Acknowledgments. We benefited from conversations with Tom Hayes on the manuscript; with Lenka Zdeborová and Florent Krząkala on the Potts spin glass; and with Alex Russell and Dimitris Achlioptas on isoperimetric inequalities. This work was partly supported by the McDonnell Foundation and the National Science Foundation under grant CCF-1117426. Part of this work was done in 2008 while the third author was a student at Carnegie Mellon and a Research Experience for Undergraduates intern at the Santa Fe Institute.

References

1. Achlioptas, D., Coja-Oghlan, A., Ricci-Tersenghi, F.: On the solution-space geometry of random constraint satisfaction problems. Random Struct. Algorithms 38(3), 251–268 (2011)
2. Achlioptas, D., Molloy, M.: Almost All Graphs with $2.522n$ Edges are not 3-Colorable. Electronic Journal of Combinatorics 6 (1999)
3. Achlioptas, D., Moore, C.: Two moments suffice to cross a sharp threshold. SIAM Journal on Computing 36, 740–762 (2006)
4. Achlioptas, D., Moore, C.: On the 2-Colorability of Random Hypergraphs. In: Rolim, J.D.P., Vadhan, S.P. (eds.) RANDOM 2002. LNCS, vol. 2483, pp. 78–90. Springer, Heidelberg (2002)
5. Achlioptas, D., Moore, C.: The Chromatic Number of Random Regular Graphs. In: Jansen, K., Khanna, S., Rolim, J.D.P., Ron, D. (eds.) APPROX and RANDOM 2004. LNCS, vol. 3122, pp. 219–228. Springer, Heidelberg (2004)
6. Achlioptas, D., Naor, A.: The Two Possible Values of the Chromatic Number of a Random Graph. Ann. Math. 162(3), 1333–1349 (2005)
7. Achlioptas, D., Peres, Y.: The Threshold for Random k-SAT is $2k \log 2 - O(k)$. J. AMS 17, 947–973 (2004)
8. Bhatnagar, N., Vera, J.C., Vigoda, E., Weitz, D.: Reconstruction for Colorings on Trees. SIAM J. Discrete Math. 25(2), 809–826 (2011)
9. Coja-Oghlan, A., Panagiotou, K.: Catching the k-NAESAT threshold. In: Proc. STOC 2012, pp. 899–908 (2012)
10. Coja-Oghlan, A., Zdeborová, L.: The condensation transition in random hypergraph 2-coloring. In: Proc. SODA 2012, pp. 241–250 (2012)
11. Dubois, O., Mandler, J.: On the non-3-colorability of random graphs (preprint), arXiv:math/0209087v1
12. Kaporis, A.C., Kirousis, L.M., Stamatiou, Y.C.: A note on the non-colorability threshold of a random graph. Electronic Journal of Combinatorics 7(1) (2000)
13. Krząkala, F., Montanari, A., Ricci-Tersenghi, F., Semerjian, G., Zdeborová, L.: Gibbs states and the set of solutions of random constraint satisfaction problems. Proc. Natl. Acad. Sci. 104(25), 10318–10323 (2007)
14. Krząkala, F., Zdeborová, L.: Potts Glass on Random Graphs. Euro. Phys. Lett. 81, 57005 (2008)
15. Maneva, E.N., Sinclair, A.: On the satisfiability threshold and clustering of solutions of random 3-SAT formulas. Theor. Comp. Sci. 407(1-3), 359–369 (2008)
16. Mertens, S., Mézard, M., Zecchina, R.: Threshold values of Random k-SAT from the cavity method. Random Structures and Algorithms 28, 340–373 (2006)
17. Mézard, M., Parisi, G., Zecchina, R.: Analytic and Algorithmic Solution of Random Satisfiability Problems. Science 297 (2002)
18. Mitzenmacher, M., Upfal, E.: Probability and Computing: Randomized Algorithms and Probabilistic Analysis. Cambridge University Press (2005)
19. Montanari, A., Restrepo, R., Tetali, P.: Reconstruction and Clustering in Random Constraint Satisfaction Problems. SIAM J. Disc. Math. 25(2), 771–808 (2011)
20. Mulet, R., Pagnani, A., Weigt, M., Zecchina, R.: Coloring random graphs. Phys. Rev. Lett. 89 (2002)
21. Sly, A.: Reconstruction of Random Colourings. Communications in Mathematical Physics 288(3), 943–961 (2009)
22. Zdeborová, L., Boettcher, S.: Conjecture on the maximum cut and bisection width in random regular graphs. J. Stat. Mech. (2010)
23. Zdeborová, L., Krząkala, F.: Phase transitions in the coloring of random graphs. Phys. Rev. E 76, 031131 (2007)

Sparse and Lopsided Set Disjointness via Information Theory

Anirban Dasgupta, Ravi Kumar, and D. Sivakumar

Yahoo! Research, 701 First Ave, Sunnyvale, CA 94089
{anirban,ravikumar,dsiva}@yahoo-inc.com

Abstract. We study two natural variations of the set disjointness problem, arguably the most central problem in communication complexity.

For the k-sparse set disjointness problem, where the parties each hold a k-element subset of an n-element universe, we show a tight $\Theta(k \log k)$ bound on the randomized one-way communication complexity. In addition, we present a slightly simpler proof of an $O(k)$ upper bound on the general randomized communication complexity of this problem, due originally to Håstad and Wigderson.

For the lopsided set disjointness problem, we obtain a simpler proof of Pătraşcu's breakthrough result, based on the information cost method of Bar-Yossef et al. The information-theoretic proof is both significantly simpler and intuitive; this is the first time the direct sum methodology based on information cost has been successfully adapted to the asymmetric communication setting. Our result shows that when Alice has a elements and Bob has b elements ($a \ll b$) from an n-element universe, in any randomized protocol for disjointness, either Alice must communicate $\Omega(a)$ bits or Bob must communicate $\Omega(b)$ bits.

1 Introduction

In the *set disjointness* problem in communication complexity, Alice holds a subset X of an n-element universe U and Bob holds a subset Y of U, and their goal is to decide if $X \cap Y = \varnothing$. Set disjointness has played a fundamental role in the theory of communication complexity and data structure complexity, a role similar to that of the satisfiability problem in the complexity theory of polynomial time bounded computations.

It is fairly easy to show a tight $\Theta(n)$ bound on the deterministic communication complexity of set disjointness. It is also easy to establish a $\Theta(\log n)$ bound on the nondeterministic communication complexity[1] of set disjointness, and a $\Theta(n)$ bound on its co-nondeterministic communication complexity. For these results, please see [17]. An $\Omega(n)$ lower bound on the randomized communication complexity of set disjointness was first shown by Kalyanasundaram and Schnitger [15]. Later Razborov [20] presented a more accessible, if somewhat magical, proof of this result. Both proofs establish the lower bound by showing an $\Omega(n)$ lower bound on the *distributional complexity*[2] of the problem, and by appealing to Yao's minimax principle. From a technical

[1] Here we assume the standard convention that the YES instances of set disjointness to be the ones where $X \cap Y$ is non-empty and the NO instances to be the ones where $X \cap Y = \varnothing$.

[2] The δ-error distributional communication complexity of a problem f is defined as follows: the maximum, over all distributions μ on instances of f, of the minimum, over all deterministic protocols Π that correctly solve f on all but a δ fraction of the inputs when chosen according to μ, of the maximum communication incurred by Π.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 517–528, 2012.
© Springer-Verlag Berlin Heidelberg 2012

perspective, Razborov's proof is a celebrated result. To obtain an $\Omega(n)$ lower bound on the distributional complexity of set disjointness, one had to create a family of "hard instances" (X, Y) where X and Y could not be chosen independently[3]. Doing so, however, rendered the analysis considerably difficult. Razborov showed an elegant (and now standard) way to deal with this, by decomposing a non-product distribution as a convex combination of product distributions, and analyzing the resulting components. Raz [19] later used this idea in his famous "parallel repetition theorem." Bar-Yossef et al. [4] further illuminated this argument by casting it as a "direct sum."

The work of [4] was an interesting step in communication complexity for several reasons. First, it provided a powerful demonstration of the role of information-theoretic methods in communication complexity, an idea that was implicit in the works of [1, 21] and that was brought to light by the elegant direct sum theorem of Chakrabarti et al. [7] (whose ideas [4] built upon). Secondly, [4] highlighted the role of the Hellinger distance as a metric that offers an elegant way to obtain tight results in communication complexity — demonstrated by its use [4, 12] in obtaining tight lower bounds for data stream algorithms. From a technical viewpoint, the work of [4] "bypasses" the use of Yao's minimax principle and provides strong lower bounds directly on the randomized communication complexity of problems[4]. More broadly, there has been a great deal of progress during the past decade in understanding the interplay between communication complexity and information theory, and taking advantage of it to prove strong communication complexity lower bounds; the tour de force of Braverman and Rao [6] is the most recent development, but there has been plenty of others [14, 11, 9, 22, 13, 5].

Sparse and Lopsided Disjointness. The results of the present paper concern two natural variations of the set disjointness problem. In the first variant, which we dub the *sparse set disjointness* problem[5], Alice and Bob hold sets of size $k \leq n$, and the goal is to understand the communication complexity (specifically randomized communication complexity) of the resulting disjointness problem. In the second variant, which is known in the literature as the *lopsided set disjointness* problem, there is a significant asymmetry in the sizes of the sets that Alice and Bob hold; typically, Bob holds a (much) larger subset of the universe, and the goal is to obtain lower bounds as functions of the skew in their set sizes and how their set sizes relate to the size of the universe.

The sparse disjointness problem was first studied by Håstad and Wigderson [10], who showed an upper bound of $O(k)$ via a randomized protocol. We present a somewhat simpler alternate proof of this result. If one is only interested in establishing an upper bound that is independent of the universe size n (as [10] seemingly are) we show that an $O(k \log k)$ upper bound can be established easily using simple ideas commonly employed in probabilistic data structures. In fact, we show this by presenting a simultaneous protocol, in which Alice and Bob independently send a single message to a

[3] A result of Babai, Frankl, and Simon [2] shows that for any "product distribution" where X and Y are chosen independently, disjointness can be computed with negligible error using $O(\sqrt{n})$ communication.

[4] Subsequently, Jayram [private communication] has shown how to derive distributional lower bounds using the information theory methods employed in [4].

[5] We refer to this sometimes as the k-sparse set disjointness when we wish to make the parameter k explicit.

referee, who is then able to decide the given instance. We complement this result by establishing an $\Omega(k \log k)$ lower bound for the randomized one-way communication complexity of k-sparse disjointness. The proof of this uses an information-theoretic argument based on Fano's inequality, developed in [3], and elementary constructions of set systems with required properties. For the general case of r-round protocols, we conjecture a lower bound of $\Omega(k \log \frac{k}{2^r})$.

Turning to the lopsided disjointness problem, Pătraşcu's fundamental work [18] has shown that a variety of lower bound and trade-off results in data structure complexity can be obtained directly as consequences of tight lower bounds on lopsided set disjointness. Pătraşcu showed that there are instances of set disjointness where Alice has an N-element subset of an n-element universe and Bob has a subset with NB elements such that in any correct protocol for disjointness, for any $\tau > 0$, either Alice communicates $\Omega(\tau N \log B)$ bits or Bob communicates $NB^{1-O(\tau)}$ bits. We obtain a version of this theorem; when instantiated to parameters similar to those of [18], we obtain bounds that imply that either Alice communicates $\Omega(N)$ bits, or Bob communicates NB bits (corresponding to $\tau = 1/\log B$). Thus our result is slightly less general than that of Pătraşcu — we do not obtain the comparable tradeoff for larger values of τ; in particular, for constant τ, we lose the $\log B$ factor, but this factor is not important for the data structure results. The more significant feature is our proof: we show how to adapt the information-theoretic machinery of [4] seamlessly to the case of asymmetric communication complexity. The proof of [18] begins with intuition that can be expressed naturally in information-theoretic language, but in the execution, tedious technical details supersede the underlying clarity. We restore the simplicity and clarity of the information cost argument of [3] in the asymmetric setting; to our knowledge, this was not known.

2 Preliminaries

Let $f : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ be a Boolean function. We consider the two-party communication complexity model, where Alice holds $X \in \mathcal{X}$ and Bob holds $Y \in \mathcal{Y}$ and they wish to jointly compute $f(X, Y)$ by exchanging messages according to a (possibly randomized) protocol $\Pi(\cdot, \cdot)$. A protocol is said to be δ-*error* protocol for f if for all inputs, at the end of the protocol, Bob is able to correctly compute f with probability at least $1 - \delta$. The cost of a protocol is the maximum length of the transcript, over all inputs and over the random coins of Alice and Bob. The δ-*error randomized communication complexity* of f is the cost of the cheapest δ-error protocol for f. We assume that Alice and Bob have access to shared public randomness.

Let $\Pi = \Pi(X, Y)$ be the random variable denoting the transcript of the protocol for a fixed input pair (X, Y). We write Π_A to denote Alice's communication during the protocol, i.e., the sequence of messages written by Alice; similarly, we write Π_B to denote the sequence of Bob's messages. For convenience, we write $\Pi = \Pi_A \Pi_B$, even though the messages of Alice and Bob are interleaved and not concatenated.

In the *set disjointness problem* DISJ, Alice holds an n-element vector $X \in \{0, 1\}^n$ and Bob holds an n-element vector $Y \in \{0, 1\}^n$; by mild abuse of notation, we will think of X and Y as the characteristic vectors of subsets of $[n] = \{1, \ldots, n\}$. The YES instances of DISJ are pairs (X, Y) of inputs such that $X \cap Y \neq \varnothing$, and the NO instances

are pairs (X, Y) of inputs such that $X \cap Y = \varnothing$. For any set X and an element y, let the expression $X[y]$ denote the $0/1$ indicator variable "$y \in X$."

In this paper we focus on two variations of the set disjointness problem. In the *k-sparse set disjointness problem*, we have $|X| = |Y| = k \leq n$. In the *lopsided set disjointness problem*, we have $|Y| \gg |X|$. In this case, there are two parameters of interest: the size n of the universe and the relative sizes of X and Y.

Information Theory. We use the following elementary notions from information theory. Let $H(X)$ denote the entropy of a random variable X, let $H(X \mid Y)$ denote the conditional entropy of X given Y. The joint entropy $H(X, Y)$ is given by the chain rule $H(X, Y) = H(Y|X) + H(X)$. Let $I(X; Y) = H(X) - H(X \mid Y)$ denote the mutual information between X and Y. Conditioning on a random variable always reduces its entropy: $H(X \mid Y) \leq H(X)$ and conditional entropy satisfies sub-additivity: $H(X, Y \mid Z) \leq H(X \mid Z) + H(Y \mid Z)$, with equality holding if and only if X is independent of Y conditioned on Z. For more background, please see [8].

3 Sparse Disjointness

In this section we focus on the communication complexity of the k-sparse disjointness problem. Recall that in this case, both Alice and Bob have sets of size $k < n$ and they wish to determine if their sets intersect. We show a tight communication bound of $\Theta(k \log k)$ in the one-way and simultaneous models and $\Theta(k)$ in the general model. The interesting aspect of these bounds are that they are *independent* of the universe size n, a fact that was highlighted in a recent work of Håstad and Wigderson [10], who obtained an $O(k)$ multi-round protocol.

3.1 Upper Bounds

Our protocols use a simple data structure for encoding a set that allows membership testing with one-sided error; this data structure is based on the ideas underlying popular data structures such as Bloom filters. First we provide some background on the main tool that will be used in the protocols.

Let U and R be discrete sets. For a function $f : U \to R$ and a subset $X \subseteq U$, $f(X)$ will denote the set of values $\{f(x) \mid x \in X\}$. Call a function $f : U \to R$ a *random mapping* if for each $u \in U$, $f(u)$ is distributed uniformly at random in R, and the random variables $\{f(u) \mid u \in U\}$ are all independent of each other. We first collect some elementary facts about random mappings.

(1) Let $X \subseteq [n]$ be a set with k elements. If $g : [n] \to [2k]$ is a random mapping, for $y \notin X$ and $x \in X$, $\Pr[g(y) = g(x)] = 1/(2k)$, so by the union bound, $\Pr[g(y) \in g(X)] \leq 1/2$.

(2) Let $X, Y \subseteq [n]$ be two sets with k elements each such that $X \cap Y = \varnothing$. Let $\ell = 2 + \lceil \log k \rceil$. If $g_1, \ldots, g_\ell$ are independent random mappings from $[n]$ to $[2k]$, for $y \notin X$, Fact (1) implies that $\Pr[(\forall i \in [\ell])[g_i(y) \in g_i(X)]] \leq (1/2)^\ell = 1/(4k)$. Therefore, by the union bound, $\Pr[(\exists y \in Y)(\forall i \in [\ell])[g_i(y) \in g_i(X)]] \leq 1/4$.

(3) Let $X, Y \subseteq [n]$ be two sets with k elements each such that $X \cap Y = \varnothing$. Let $g : [n] \to [4k]$ be a random mapping, and let F denote the set of "false positives" $\{y \in$

$Y \mid g(y) \in g(X)\}$. Then $E[|F|] = E[\sum_{y \in Y} g(X)[g(y)]]$, where $g(X)[g(y)]$ denotes the 0–1 indicator random variable that is 1 iff $g(y) \in g(X)$. Since $E[g(X)[g(y)]] = \Pr[g(y) \in g(X)] \leq 1/4$, by the linearity of expectation, we have $E[|F|] \leq k/4$. Moreover, the variance of each $g(X)[y]$ is at most $3/16$, and since $|F|$ is the sum of independent random variables with this variance, we have $\mathrm{Var}[|F|] \leq 3k/16 \leq k/4$. By the Chebyshev inequality, this implies that $\Pr[|F| \geq k/2] \leq 4/k$.

(4) For any mapping $f : U \to R$, any subset $X \subseteq U$, and any $y \in U$, given access to the mapping f, the test $f(y) \in f(X)$ can be carried out given only the $|R|$ Boolean variables $\{(\exists x \in X)[f(x) = z] \mid z \in R\}$. When $|X| = \Theta(|R|)$, these variables can be compactly encoded in $|R|$ bits (as opposed to the $|X| \log |R|$ bits required to write down $f(x)$ for each $x \in X$).

Using the properties of the random mappings, we now obtain one-way and multi-round protocols for sparse disjointness. The main idea is for Alice to send Bob an encoding of her set using random mappings.

Theorem 1. *The randomized one-way communication complexity of the k-sparse disjointness problem is $O(k \log k)$.*

Proof. Alice will encode her set using random mappings (realized by using the shared public random bits) $h_1, \ldots, h_l$, where the value of l is from Fact (2), and $h_i : [n] \to [2k]$; for each $i \in [l]$, she communicates to Bob the $2k$-bit string that encodes the Boolean variables $\{(\exists x \in X) h_i(x) = j \mid j \in [2k]\}$. As noted in Fact (4), this allows Bob to check if $h_i(y) \in h_i(X)$ for each $i \in [l]$. Clearly, if there is some $y \in X \cap Y$, $h_i(y) \in h_i(X)$ is true for all $i \in [l]$. From Fact (2), if X and Y are disjoint, the probability that Bob accidentally concludes that $X \cap Y \neq \varnothing$ is bounded by $1/4$. □

It is easy to see that by using a single hash function into the range $[4k^2]$ and enumerating the hash values, this protocol can be realized in the simultaneous model as well. We now use the protocol of Theorem 1 to build an $O(k)$ multi-round protocol. The main idea is for Alice and Bob to exchange the encoding of their respective inputs. Alice then uses Bob's encoding of his input to eliminate a constant fraction of her own input elements from future consideration; Bob does likewise. The players then repeat. We can show the following (proof omitted).

Theorem 2 ([10]). *The randomized communication complexity of the k-sparse disjointness problem is $O(k)$.*

3.2 Lower Bounds

For arbitrary communication, a lower bound of $\Omega(k)$ follows from standard disjointness lower bounds by padding. We therefore focus on the one-way model. We prove an $\Omega(k \log k)$ lower bound on the one-way randomized communication complexity of the sparse set disjointness problem, thereby showing that Theorem 1 is tight. We do this by exhibiting a subset $\mathcal{X} \times \mathcal{Y}$ of Alice and Bob's inputs such that the one-way complexity of k-sparse set disjointness is $\Omega(k \log k)$ when Alice is given a uniformly chosen $X \in \mathcal{X}$ and Bob is given a uniformly chosen $Y \in \mathcal{Y}$. The proof is based on a technique introduced by Bar-Yossef et al. [3], using Fano's information theory inequality to derive

lower bounds on communication complexity. This method enables to formally prove an intuitive argument of the following flavor: suppose that for $X, X' \in \mathcal{X}$, the sequences $\{f(X,Y) \mid Y \in \mathcal{Y}\}$ and $\{f(X',Y) \mid Y \in \mathcal{Y}\}$ are different, then to ensure that the protocol will work correctly (whp.) for most $Y \in \mathcal{Y}$, Alice has no choice but to transmit X to Bob.

Definition 1 (Design). *A family $\mathcal{X}$ of subsets of $[k^2]$ is a* (k, α)-design *if $|X| = k$ for each $X \in \mathcal{X}$ and for $X \neq X'$, $|X \cap X'| \leq \alpha k$.*

The next two statements can be shown using the probabilistic method (proof omitted).

Lemma 1. *For sufficiently large integers k and any α, $0 \leq \alpha \leq 1$, there is a (k, α)-design of size $2^{\alpha k \log k}$.*

Lemma 2. *Let $\mathcal{X}$ denote a family of m subsets of $[k^2]$ that is a (k, α)-design, for $\alpha < 1/2$. There is a family $\mathcal{Y} \subseteq [k^2]$ such that:*

(1) $|Y| = k$ for every $Y \in \mathcal{Y}$;
(2) $|\mathcal{Y}| \leq ak \log k$, where a is an absolute constant (independent of k);
(3) for $X \neq X' \in \mathcal{X}$, there is at least one $Y \in \mathcal{Y}$ such that Y has non-empty intersection with precisely one of X and X'.

Theorem 3 (Fano's inequality). *Let F and P be two random variables that take values, respectively in S_F and S_P. Let $g : S_P \to S_F$ be a prediction function that, given an observed value of P guesses the value of F. Let δ be the prediction error: $\delta = \Pr_{F,P}[g(P) \neq F]$. Then $H(\delta) + \delta \log(|S_F| - 1) \geq H(F \mid P)$. In particular, if F is Boolean (i.e., $|S_F| = 2$), then $H(\delta) \geq H(F \mid P)$.*

Theorem 4. *For $k \leq \sqrt{n}$, the randomized one-way communication complexity of the k-sparse disjointness problem is $\Omega(k \log k)$.*

Proof. Given Lemma 2, it is possible to obtain a lower bound on the "VC dimension" of the function matrix of the k-sparse disjointness problem, and appeal to a Theorem of [16] to obtain the one-way randomized lower bound. Instead, we will present a direct proof, based on simple information theoretic ideas as in [3].

Let $\alpha = 1/4$. Let $\mathcal{X}$ denote a family of $2^{\alpha k \log k}$ subsets corresponding to a (k, α) design whose universe is $[k^2]$. Since $k^2 \leq n$, we will consider the sets in $\mathcal{X}$ to be subsets of $[n]$, and identify them with (Alice's) inputs in $\{0,1\}^n$. By Lemma 2, there is a set $\mathcal{Y}$ of $ak \log k$ subsets of $[k^2]$, hence of $[n]$, that correspond to (Bob's) inputs in $\{0,1\}^n$.

We will apply Yao's minimax principle (see [17]) and obtain a lower bound on the distributional one-way communication complexity of k-sparse disjointness. Specifically, we will show that for some $\delta > 0$ (independent of k and n), when the input to Alice is chosen uniformly from $\mathcal{X}$ and the input to Bob is chosen uniformly form $\mathcal{Y}$, no deterministic one-way communication protocol can achieve an error rate less than δ.

Let X denote a subset of $[n]$ chosen uniformly from $\mathcal{X}$. let Y denote a subset of $[n]$ chosen uniformly from $\mathcal{Y}$. Abusing notation, let $\textsc{Disj}(X, Y)$ denote the 0–1 indicator random variable that is 1 iff $X \cap Y \neq \varnothing$ when X is chosen uniformly at random from $\mathcal{X}$ and Y is chosen uniformly at random from $\mathcal{Y}$. By Lemma 1, $H(X)$, the entropy of the random variable X, equals $k \log k$. By Lemma 2, the sequence of values $\{\textsc{Disj}(X, y) \mid$

$y \in \mathcal{Y}\}$ uniquely identifies X, so $H(\{\textsc{Disj}(X,y) \mid y \in \mathcal{Y}\}) = H(X) = \alpha k \log k \geq 0.25k \log k$.

Let $\Pi = (A, B)$ denote a deterministic one-way communication protocol for the k-sparse set disjointness problem where Alice's function is denoted by A, and Bob's function is denoted by B; let δ denote the error of the protocol Π when the inputs X and Y are chosen as described above; we will assume that δ is sufficiently small so that $H(\delta) < 1/8a$, where a is the constant from the statement of Lemma 2. Also let $t = ak \log k$.

Now we have:

$$\begin{aligned}
H(\delta) &\geq H(\textsc{Disj}(X,Y) \mid A(X), Y) && \text{by Theorem 3,}\\
&= \frac{1}{t}\sum_{y\in\mathcal{Y}} H(\textsc{Disj}(X,y) \mid A(X), Y = y)\\
&= \frac{1}{t}\sum_{y\in\mathcal{Y}} H(\textsc{Disj}(X,y) \mid A(X)) && \text{since } X \perp Y,\\
&\geq \frac{1}{t} H(\{\textsc{Disj}(X,y)\}_{y\in\mathcal{Y}} \mid A(X)) && \text{by sub-additivity,}\\
&\geq \frac{1}{t}\left(H(\{\textsc{Disj}(X,y)\}_{y\in\mathcal{Y}}) - H(A(X))\right) && \text{by the chain rule,}\\
&\geq \frac{1}{t}\left(H(\{\textsc{Disj}(X,y)\}_{y\in\mathcal{Y}}) - |A(X)|\right).
\end{aligned}$$

Thus we have $|A(X)| \geq H(\{\textsc{Disj}(X,y)\}_{y\in\mathcal{Y}}) - tH(\delta)$. On the other hand, we have $H(\{\textsc{Disj}(X,y)\}_{y\in\mathcal{Y}}) \geq 0.25k \log k$ and $tH(\delta) \leq k \log k/8$, so it follows that $|A(X)| \geq k \log k/8$. □

It is easy to generalize Theorem 4 to the case when Alice has a set of size k and Bob has a set of size n/k to get a communication lower bound of $\Omega(k \log(n/k))$. In this case, if $k > \sqrt{n}$, then Bob's input can always be padded with dummy elements from the universe so that his input will be of the same size as Alice's input. These two together will extend Theorem 4 to work for all k.

4 Lopsided Disjointness

In this section we focus on the asymmetric communication complexity of the lopsided set disjointness problem. Recall that in the lopsided version of the problem, Bob holds a set that is much bigger than Alice's set and they wish to jointly determine if their sets intersect. The lopsided disjointness problem plays a central role in obtaining data structure lower bounds [18]. Our lower bound is based on an information-theoretic framework and we first provide some background material.

Let P and Q be two probability distributions over some universe Ω. We denote by $V(P,Q)$ the *total variation distance* between P and Q, defined by $V(P,Q) = (1/2)\sum_{\omega\in\Omega} |P(\omega) - Q(\omega)|$. We denote by $h(P,Q)$ the *Hellinger distance* between P and Q, defined by $h^2(P,Q) = \sum_{\omega\in\Omega}(\sqrt{P(\omega)} - \sqrt{Q(\omega)})^2$; it is known that $h(\cdot,\cdot)$

is a metric. We denote by $\mathrm{KL}(P\|Q)$ the *Kullback-Leibler divergence* between P and Q, defined by $\mathrm{KL}(P\|Q) = \sum_{\omega\in\Omega} P(\omega)\log_2 \frac{P(\omega)}{Q(\omega)}$.

We begin by listing some known facts about Hellinger distance that we use.

Fact 5. *$h^2(P,Q) \le V(P,Q) \le h^2(P,Q)(2-h^2(P,Q))^{1/2}$ and consequently, for any $0 \le \alpha \le 1$, $h^2(P, \alpha P + (1-\alpha)Q) \le (1-\alpha)/2$.*

Define the *Rényi divergence* $h_\alpha(P,Q)$ as $h_\alpha(P,Q) = 1 - \sum_{\omega\in\Omega} P(\omega)^\alpha Q(\omega)^{1-\alpha}$; note that the Hellinger distance $h(P,Q) = \sqrt{h_{1/2}(P,Q)}$. We have the following relationship between Rényi divergences.

Lemma 3 ([4]). *For $\alpha < \beta$, $\frac{\alpha}{\beta} h_\beta(P,Q) \le h_\alpha(P,Q) \le \frac{1-\alpha}{1-\beta} h_\beta(P,Q)$.*

Let $0 < \alpha < 1$ be a constant and denote $\mu = \alpha P + (1-\alpha)Q$. Define the generalized *Jensen–Shannon divergence* as follows: $\mathrm{JS}_\alpha(P,Q) = \alpha\mathrm{KL}(P\|\mu) + (1-\alpha)\mathrm{KL}(Q\|\mu)$. We now show a relationship between the generalized Jensen–Shannon divergence and the Hellinger distance (proof omitted).

Lemma 4. *For $\alpha < 1/2$, $\mathrm{JS}_\alpha(P,Q) \ge \frac{\alpha}{\ln 2} h^2(P,Q)$.*

Finally, the following statement, derived from [4], will be useful.

Fact 6. *Let ϕ denote a mapping from the set $\{z_1, z_2\}$ where $\phi(z_1)$ and $\phi(z_2)$ are two random variables, and let ϕ_z denote the probability distribution of $\phi(z)$. Let Z denote a random variable with uniform distribution in $\{z_1, z_2\}$. Suppose $\phi(z)$ is independent of Z for each $z \in \{z_1, z_2\}$. Then:*

(i) $I(Z : \phi(Z)) = \mathrm{JS}(\phi_{z_1}, \phi_{z_2})$.

(ii) If Y be another distribution taking values in $\{0,1\}$, and let $\psi(Z,Y)$ be a random variable that is independent of both Z and Y for each (z,y). Let $\psi_{z,y}$ denote the corresponding probability distributions. Let $\rho = \Pr[Y=1]$. Then $I(Z : \psi(Z,Y)) = \mathrm{JS}(\rho\psi_{01} + (1-\rho)\psi_{00}, \rho\psi_{11} + (1-\rho)\psi_{10})$.

4.1 Lower Bounds

To derive lower bounds for lopsided disjointness, we generalize the class of "hard instances" of disjointness introduced by Bar-Yossef et al. [4]. The instances defined by [4] have the property that each element of the universe belongs either to Alice (with probability $1/4$), or to Bob (with probability $1/4$), or to neither (with probability $1/2$), but never to both; this naturally necessitates a non-product distribution on the input instances. We generalize this in two ways — to allow for lopsidedness in the set sizes and to allow for product distributions.

Let α, β, and θ be three parameters in $[0,1]$. The parameters are allowed to be functions of n, the size of the universe. Since our proof will access the individual coordinates of the input, in this section, we will use $\mathbf{X}$ and $\mathbf{Y}$ to denote the inputs to Alice and Bob. We will define a vector $\mathbf{D} = \mathbf{D}_1 \ldots \mathbf{D}_n$ of random variables, and using these, the inputs $\mathbf{X} = \mathbf{X}_1, \ldots, \mathbf{X}_n$ and $\mathbf{Y} = \mathbf{Y}_1, \ldots, \mathbf{Y}_n$ for disjointness are defined as follows. The

random variables $\mathbf{D}_1, \ldots, \mathbf{D}_n$ will be n independent and identical random variables with the following distribution:

$$\mathbf{D}_i = \begin{cases} \text{A} & \text{w.p. } \theta, \\ \text{B} & \text{w.p. } \theta, \\ \text{AB} & \text{w.p. } 1 - 2\theta. \end{cases}$$

Let R_{A} and R_{B} denote independent random variables in the set $\{0, 1\}$, where $\Pr[R_{\text{A}} = 1] = \alpha$ and $\Pr[R_{\text{B}} = 1] = \beta$. Given $\mathbf{D}_i$, the pair $(\mathbf{X}_i, \mathbf{Y}_i)$ of input bits to Alice and Bob are generated (independently for each i) as follows:

$$(\mathbf{X}_i, \mathbf{Y}_i) = \begin{cases} (R_{\text{A}}, 0) & \text{if } \mathbf{D}_i = \text{A}, \\ (0, R_{\text{B}}) & \text{if } \mathbf{D}_i = \text{B}, \\ (R_{\text{A}}, R_{\text{B}}) & \text{if } \mathbf{D}_i = \text{AB}. \end{cases}$$

Let ζ denote the distribution of $((\mathbf{X}_i, \mathbf{Y}_i), \mathbf{D}_i)$ for $i \in [n]$. Then, $((\mathbf{X}, \mathbf{Y}), \mathbf{D})$ has distribution ζ^n. The proof of the following statement is omitted in this version.

Lemma 5. *$|\Pi_A| \geq I(\mathbf{X} : \Pi \mid \mathbf{D})$ and $|\Pi_B| \geq I(\mathbf{Y} : \Pi \mid \mathbf{D})$.*

Lemma 6 (Information cost decomposition). *$I(\mathbf{X} : \Pi \mid \mathbf{D}) \geq \sum_{i=1}^{n} I(\mathbf{X}_i : \Pi \mid \mathbf{D})$ and $I(\mathbf{Y} : \Pi \mid \mathbf{D}) \geq \sum_{i=1}^{n} I(\mathbf{Y}_i : \Pi \mid \mathbf{D})$.*

Let $\mathbf{D}_{-i}$ denote the $n-1$ random variables $\mathbf{D}_1, \ldots, \mathbf{D}_{i-1}, \mathbf{D}_{i+1}, \ldots, \mathbf{D}_n$. Let $\mathbf{d}$ denote a value of $\mathbf{D}$, and similarly let $\mathbf{d}_{-i}$ denote $\mathbf{d}_1, \ldots, \mathbf{d}_{i-1}, \mathbf{d}_{i+1}, \ldots, \mathbf{d}_n$. Let D, U, and V denote random variables such that $((U, V), D)$ is distributed according to ζ (the same distribution as $((\mathbf{X}_i, \mathbf{Y}_i), \mathbf{D}_i)$ for each i, $1 \leq i \leq n$).

Lemma 7 (Reduction from AND to DISJ). *If there is a δ-error protocol Π for the disjointness problem on n-element vectors, then for each j, $1 \leq j \leq n$, and $\mathbf{d}$, there is an ϵ-error protocol $P = P_{j,\mathbf{d}}$ for the problem of computing the* AND *of two bits, where $\epsilon \leq \delta + n(1-2\theta)\alpha\beta$. Moreover, when $((\mathbf{X}, \mathbf{Y}), \mathbf{D})$ has distribution ζ^n and $((U, V), D)$ has distribution ζ, the protocol P satisfies $I(\mathbf{X}_j : \Pi \mid \mathbf{D}_j, \mathbf{D}_{-j} = \mathbf{d}_{-j}) = I(U : P \mid D)$ and $I(\mathbf{Y}_j : \Pi \mid \mathbf{D}_j, \mathbf{D}_{-j} = \mathbf{d}_{-j}) = I(V : P \mid D)$.*

Proof. The proof follows the reduction presented in [4]. We begin with some notation. Given $\mathbf{x}' \in \{0,1\}^n$, $u \in \{0,1\}$, and $j \in \{1, \ldots, n\}$, define $\mathbf{X} = \mathbf{X}(u, j, \mathbf{x}')$ by $\mathbf{X}_i = \mathbf{x}'_i$ for $i \neq j$, and $\mathbf{X}_j = u$; similarly, for $\mathbf{y}'$, v, and j, define $\mathbf{Y} = \mathbf{Y}(v, j, \mathbf{y}')$ as a copy of y' with the jth coordinate replaced with v.

Given protocol Π for DISJ, we build the protocol P for AND as follows: Protocol P will have j and d hardwired into it. On the input (u, v), P first samples $\mathbf{x}'$ and $\mathbf{y}'$ according to the distribution ζ^n, conditioned on $\mathbf{D}_{-j} = \mathbf{d}$, and then simulates $\Pi(\mathbf{X}, \mathbf{Y})$, where $\mathbf{x} = \mathbf{X}(u, j, \mathbf{x}')$ and $\mathbf{y} = \mathbf{Y}(v, j, \mathbf{y}')$. Note that since $\mathbf{d}$ is hardwired into P, we can use the private coins of A and B to create the vectors $\mathbf{x}$ and $\mathbf{y}$ without any communication. Note that P is a possibly erroneous in computing the AND of u and v; this happens precisely when $\mathbf{x}_{-j}$ and $\mathbf{y}_{-j}$ are not disjoint, which happens with probability at most $n(1 - 2\theta)\alpha\beta$. Thus P is an ϵ-protocol for the AND function where $\epsilon \leq \delta + n(1 - 2\theta)\alpha\beta$.

By arguments identical to those of [4], it also follows that the joint distribution of $(U, V, D, P(U, V))$ is identical to the joint distribution of $(\mathbf{X}_j, \mathbf{Y}_j, \Pi(\mathbf{X}, \mathbf{Y}), \mathbf{D}_j)$, conditioned on the event $\mathbf{D}_{-j} = \mathbf{d}_{-j}$. Hence, $I(\mathbf{X}_j : \Pi \mid \mathbf{D}_j, \mathbf{D}_{-j} = \mathbf{d}_{-j}) = I(U : P \mid D)$ and $I(\mathbf{Y}_j : \Pi \mid \mathbf{D}_j, \mathbf{D}_{-j} = \mathbf{d}_{-j}) = I(V : P \mid D)$. □

Lemma 8 (Information cost LB for AND). *Let P be any ϵ-error protocol for the problem of computing the* AND *of two bits, and let $\Delta = (1 - 2\sqrt{\epsilon})/2$. Assume ϵ is small enough so that $\Delta \geq 1/4$. For each j, $1 \leq j \leq n$, either $I(U : P \mid D) \geq (\alpha/\ln 2)(\Delta/2)(\theta + (1-2\theta)(1-16\sqrt{\beta}))$ or $I(V : P \mid D) \geq (\beta/\ln 2)(\Delta/2)(\theta + (1-2\theta)(1-16\sqrt{\alpha}))$.*

Proof. Let P_{uv} denote the distribution of (the transcript of) $P(u, v)$.

$$\begin{aligned} I(U : P \mid D) &= \theta I(U : P \mid D = \text{A}) + \theta I(U : P \mid D = \text{B}) + (1-2\theta) I(U : P \mid D = \text{AB}) \\ &= \theta I(R_\text{A} : P(R_\text{A}, 0)) + (1-2\theta) I(R_\text{A} : P(R_\text{A}, R_\text{B})). \end{aligned}$$

For the first term, we can use Fact 6(i) and Lemma 4 to get

$$I(R_\text{A} : P(R_\text{A}, 0)) \geq \frac{\alpha}{\ln 2} h^2(P_{00}, P_{10}).$$

We now bound the second term as follows. We use Fact 6(ii) to bound

$$I(R_\text{A} : P(R_\text{A}, R_\text{B})) \geq \frac{\alpha}{\ln 2} h^2((1-\beta)P_{00} + \beta P_{01}, (1-\beta)P_{10} + \beta P_{11}).$$

Putting these together, we obtain
$I(U : P \mid D) \geq \frac{\alpha}{\ln 2}(\theta h^2(P_{00}, P_{10}) + (1-2\theta) h^2((1-\beta)P_{00} + \beta P_{01}, (1-\beta)P_{10} + \beta P_{11}))$. Similarly, $I(V : P \mid D) \geq \frac{\beta}{\ln 2}(\theta h^2(P_{00}, P_{01}) + (1-2\theta) h^2((1-\alpha)P_{00} + \alpha P_{10}, (1-\alpha)P_{10} + \alpha P_{11}))$.

Also we have:

$$\begin{aligned} 2(h^2(P_{00}, P_{01}) + h^2(P_{00}, P_{10})) &\geq h^2(P_{01}, P_{10}) \\ &\quad \text{via Cauchy–Schwarz and the triangle inequalities,} \\ &= h^2(P_{00}, P_{11}) \\ &\quad \text{by the Cut-and-paste Lemma [4, Lemma 6.3],} \\ &\geq 1 - 2\sqrt{\epsilon} \\ &\quad \text{since } P \text{ is an } \epsilon\text{-error protocol [4, Lemma 6.5].} \end{aligned}$$

By averaging, at least one of the following holds: $h^2(P_{00}, P_{10}) \geq (1 - 2\sqrt{\epsilon})/4$ or $h^2(P_{00}, P_{01}) \geq (1 - 2\sqrt{\epsilon})/4$. We work out the first case. Let $\Delta = (1 - 2\sqrt{\epsilon})/2$; wlog. we will assume that ϵ is small enough so that $\Delta \geq 1/4$. Denote $R = (1-\beta)P_{00} + \beta P_{01}$ and $S = (1-\beta)P_{10} + \beta P_{11}$. Now, using Fact 5 and the triangle inequality,

$$h(R, S) \;\geq\; h(P_{00}, P_{10}) - h(R, P_{00}) - h(S, P_{10}) \;\geq\; \sqrt{\Delta/2} - \sqrt{2\beta} - \sqrt{2\beta},$$

which yields

$h^2(R,S) \geq (\sqrt{\Delta/2} - \sqrt{8\beta})^2 \geq \Delta/2 - 4\sqrt{\Delta\beta} = \Delta/2(1 - 8\sqrt{\beta/\Delta}) \geq \Delta/2(1 - 16\sqrt{\beta})$.

Thus, we have $I(U : P \mid D) \geq (\alpha/\ln 2)(\Delta/2)(\theta + (1-2\theta)(1-16\sqrt{\beta}))$.

Similarly, if $h^2(P_{00}, P_{01}) \geq \Delta$, then we can show $I(V : P \mid D) \geq (\beta/\ln 2)(\Delta/2)(\theta + (1-2\theta)(1-16\sqrt{\alpha}))$. □

By combining Lemmata 5, 6, 7, and 8, we obtain a general lower bound.

Theorem 7. *For any $\delta > 0$, for sufficiently large n and all $\alpha, \beta \in [0,1]$, $\theta \in [0, 1/2]$ satisfying and $0 \leq (1-2\theta)\alpha\beta \leq 4\delta/n$, there are instances of the set disjointness problem where the expected size of Alice's set is $(1-\theta)\alpha n$, the expected size of Bob's set is $(1-\theta)\beta n$, and the expected size of the intersection is $(1-2\theta)\alpha\beta$, such that in any protocol that computes set disjointness with error at most δ, either Alice must communicate $\alpha C_0(\theta + (1-2\theta)(1-16\sqrt{\beta}))n$ bits or Bob must communicate $\beta C_0(\theta + (1-2\theta)(1-16\sqrt{\alpha}))n$ bits, where $C_0 = \frac{1}{4\ln 2}\left(1 - \sqrt{8\delta}\right)$. In addition, when $\theta = 1/2$, the sets are guaranteed to be disjoint.*

We now instantiate this lower bound to two special cases. By choosing $\beta = \Theta(1-\alpha)$ and $\theta = 1/2$, we obtain a near-tight lower bound for lopsided disjointness.

Corollary 1. *For any $\delta > 0$, for sufficiently large n and all α, $0 \leq \alpha \leq 1$, there are instances of the set disjointness problem where the expected size of Alice's set is $\alpha n/2$ and the expected size of Bob's set is $(1-\alpha)n/2$ such that in any protocol that computes set disjointness with error at most δ, either Alice must communicate $C_0\alpha n$ bits or Bob must communicate $C_0(1-\alpha)n$ bits, where $C_0 = \frac{1}{8\ln 2}\left(1 - \sqrt{8\delta}\right)$.*

Let us now compare Theorem 7 with the best known result concerning the communication complexity of lopsided disjointness: this breakthrough result is due to Pătraşcu [18, Theorem 1.4], who shows how to construct instances (X, Y) of set disjointness from a universe U such that $|X| = N$ and $|Y| = |U|/2 = NB/2$ for which in every randomized protocol for disjointness, for every value of a trade-off parameter $\tau > 0$, either Alice communicates $\tau N \log B$ bits or Bob communicates $NB^{1-O(\tau)}$ bits. By taking $\gamma = 0$, size of universe n, $\alpha = 2N/n$ and $\beta = B\alpha/2$, we obtain instances where Alice's set has expected size N, Bob's set has expected size $NB/2$, and Theorem 7 guarantees that in any randomized protocol for disjointness, either Alice communicates $\Omega(N)$ bits or Bob communicates $\Omega(NB)$ bits.

Next, we set $\theta = 0$ and $\alpha = \Theta(n^{-\gamma})$, $\beta = \Theta(\delta n^{-1+\gamma})$, where $0 < \gamma < 1$ is a constant. In other words, the input distribution is a product distribution. We obtain a lower bound for this case as well.

Corollary 2. *If $\theta = 0$ and $\alpha = n^{-\gamma}$, $\beta = 4\delta n^{-1+\gamma}$ where $0 < \gamma < 1$ is a constant, then the expected size of Alice's set is $\Theta(n^{1-\gamma})$ that of Bob is $\Theta(n^{\gamma})$, the expected size of the intersection is 4δ, and the probability of a non-zero intersection is $\Theta(1)$. Then, either Alice needs to send $C_0 n^{1-\gamma}$ bits or Bob needs to send $C_0 n^{\gamma}$ bits, where $C_0 = \frac{1}{4\ln 2}\left(1 - \sqrt{8\delta}\right)$.*

Note that if we further let $\gamma = 1/2$, then the lower bound matches the upper bound of $O(\sqrt{n})$ for set disjointness for product distributions [2].

Acknowledgments. We thank George Varghese for many discussions that led us to these problems. We are grateful to the reviewers of this paper for CCC 2012 and RANDOM 2012 for their thorough reviews and very helpful comments.

References

1. Ablayev, F.M.: Lower bounds for one-way probabilistic communication complexity and their application to space complexity. TCS 157(2), 139–159 (1996)
2. Babai, L., Frankl, P., Simon, J.: Complexity classes in communication complexity theory (preliminary version). In: FOCS, pp. 337–347 (1986)
3. Bar-Yossef, Z., Jayram, T.S., Kumar, R., Sivakumar, D.: Information theory methods in communication complexity. In: CCC, pp. 93–102 (2002)
4. Bar-Yossef, Z., Jayram, T.S., Kumar, R., Sivakumar, D.: An information statistics approach to data stream and communication complexity. JCSS 68(4), 702–732 (2004)
5. Barak, B., Braverman, M., Chen, X., Rao, A.: How to compress interactive communication. In: STOC, pp. 67–76 (2010)
6. Braverman, M., Rao, A.: Information equals amortized communication. In: FOCS, pp. 748–757 (2011)
7. Chakrabarti, A., Shi, Y., Wirth, A., Yao, A.C.-C.: Informational complexity and the direct sum problem for simultaneous message complexity. In: FOCS, pp. 270–278 (2001)
8. Cover, T.M., Thomas, J.A.: Elements of Information Theory. John Wiley & Sons, Inc. (1991)
9. Harsha, P., Jain, R., McAllester, D.A., Radhakrishnan, J.: The communication complexity of correlation. TOIT 56(1), 438–449 (2010)
10. Håstad, J., Wigderson, A.: The randomized communication complexity of set disjointness. ToC 3(1), 211–219 (2007)
11. Jain, R., Radhakrishnan, J., Sen, P.: A Direct Sum Theorem in Communication Complexity Via Message Compression. In: Baeten, J.C.M., Lenstra, J.K., Parrow, J., Woeginger, G.J. (eds.) ICALP 2003. LNCS, vol. 2719, pp. 300–315. Springer, Heidelberg (2003)
12. Jayram, T.S.: Hellinger Strikes Back: A Note on the Multi-party Information Complexity of AND. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 562–573. Springer, Heidelberg (2009)
13. Jayram, T.S., Kopparty, S., Raghavendra, P.: On the communication complexity of read-once AC^0 formulae. In: CCC, pp. 329–340 (2009)
14. Jayram, T.S., Kumar, R., Sivakumar, D.: Two applications of information complexity. In: STOC, pp. 673–682 (2003)
15. Kalyanasundaram, B., Schnitger, G.: The probabilistic communication complexity of set intersection. SIDMA 5(4), 545–557 (1992)
16. Kremer, I., Nisan, N., Ron, D.: On randomized one-round communication complexity. Computational Complexity 8(1), 21–49 (1999)
17. Kushilevitz, E., Nisan, N.: Communication complexity. Cambridge University Press (1997)
18. Pătraşcu, M.: Unifying the landscape of cell-probe lower bounds. SICOMP 40(3), 827–847 (2011)
19. Raz, R.: A parallel repetition theorem. SICOMP 27(3), 763–803 (1998)
20. Razborov, A.A.: On the distributional complexity of disjointness. TCS 106(2), 385–390 (1992)
21. Saks, M.E., Sun, X.: Space lower bounds for distance approximation in the data stream model. In: STOC, pp. 360–369 (2002)
22. Sen, P., Venkatesh, S.: Lower bounds for predecessor searching in the cell probe model. JCSS 74(3), 364–385 (2008)

Maximal Empty Boxes Amidst Random Points*

Adrian Dumitrescu[1,**] and Minghui Jiang[2]

[1] Department of Computer Science, University of Wisconsin–Milwaukee, USA
dumitres@uwm.edu
[2] Department of Computer Science, Utah State University, Logan, USA
mjiang@cc.usu.edu

Abstract. We show that the expected number of maximal empty axis-parallel boxes amidst n random points in the unit hypercube $[0,1]^d$ in $\mathbb{R}^d$ is $(1 \pm o(1)) \frac{(2d-2)!}{(d-1)!} n \ln^{d-1} n$, if d is fixed. This estimate is relevant for analyzing the performance of any exact algorithm for computing the largest empty axis-parallel box amidst n points in a given axis-parallel box R, that proceeds by examining all maximal empty boxes. While the $\Theta(n \log^{d-1} n)$ bound has been claimed for $d = 3$ for more than ten years by now, and has been recently used for all $d \geq 3$ in the analysis of algorithms for computing the largest empty box, it did not rely on a valid proof. Here we present the first valid proof for the $\Theta(n \log^{d-1} n)$ bound; only an $O(n \log^{d-1} n)$ bound was previously proved.

1 Introduction

Given an axis-parallel rectangle R in the plane containing n points, the problem of computing a maximum-area empty axis-parallel sub-rectangle contained in R is one of the oldest problems studied in computational geometry. For instance, this problem arises when a rectangular shaped facility is to be located within a similar region which has a number of forbidden areas, or in cutting out a rectangular piece from a large similarly shaped metal sheet with some defective spots to be avoided [20]. In higher dimensions, finding the largest empty axis-parallel box has applications in data mining, in finding large gaps in a multi-dimensional data set [12]. Throughout this paper we refer to this problem as the Maximum Empty Box problem.

Throughout this paper, a *box* is an *open* axis-parallel hyperrectangle contained in the unit hypercube $U_d = [0,1]^d$, $d \geq 2$. Given a set S of points in U_d, a box B is *empty* if it contains no points in S, i.e., $B \cap S = \emptyset$. Some planar examples are shown in Fig. 1.

Several algorithms have been proposed for the planar problem over time [1–3, 7, 9, 19–21]. The fastest one, due to Aggarwal and Suri [1], runs in $O(n \log^2 n)$ time and $O(n)$ space. A lower bound of $\Omega(n \log n)$ in the algebraic decision tree model for this problem has been shown by McKenna et al. [19].

Backer and Keil [4, 5] proved that Maximum Empty Box is NP-hard in high dimensions, i.e., when d is part of the input. Moreover, Giannopoulos et al. [13, Theorem 3] have recently shown that Maximum Empty Box is W[1]-hard with the dimension d as

* Due to space constraints, we omit the proofs of some lemmas in this extended abstract.
** Supported in part by NSF grant DMS-1001667.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 529–540, 2012.
© Springer-Verlag Berlin Heidelberg 2012

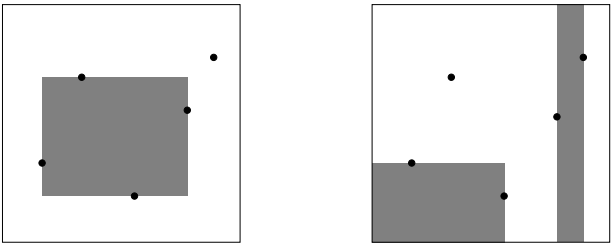

Fig. 1. A maximal empty rectangle supported by one point on each side (left), and two maximal empty rectangles supported by both points and sides of $[0,1]^2$ (right)

the parameter. This implies, by a standard technique in parameterized complexity theory [18, Section 6.3], that the existence of an exact algorithm running in $n^{o(d)}$ time is unlikely, i.e., unless the so-called Exponential Time Hypothesis (ETH) fails, i.e., unless 3-SAT can be solved in $2^{o(n)}$ time.

Backer and Keil [4, 5] also reported an exact algorithm running in $O(n^d \log^{d-2} n)$ time, for any $d \geq 3$. In particular, the running time of their algorithm for $d = 3$ is $O(n^3 \log n)$. Previously, Datta and Soundaralakshmi [10] had reported an $O(n^3)$ time exact algorithm for the $d = 3$ case, but their analysis for the running time seems incomplete. Specifically, the $O(n^3)$ running time depends on an $O(n^3)$ upper bound on the number of maximal empty boxes (see discussions in the next paragraph), but they only gave an $\Omega(n^3)$ lower bound.

An empty box of maximum volume must be maximal with respect to inclusion, thus the maximum-volume empty box in U_d is maximal. Naamad et al. [20] have shown that in the plane, the number of maximal empty rectangles is $O(n^2)$, and that this bound is tight. It was conjectured by Datta and Soundaralakshmi [10] that the maximum number of maximal empty boxes is $O(n^d)$ for each (fixed) d. The conjecture has been recently confirmed by Backer and Keil [4] for $d \geq 3$. For a fixed d, matching lower bounds has been obtained by Kaplan et al. [14], by Backer and Keil [4] and by Dumitrescu and Jiang [11], independently of each other. (Kaplan et al. [14] seemed to be unaware of the conjecture and the earlier papers [10, 20].)

Hence the maximum number of maximal empty boxes is $\Theta(n^d)$ for each fixed d. This means that any algorithm for computing a maximum-volume empty box based on enumerating maximal empty boxes is bound to be inefficient in the worst case. However, as it is the case with the algorithm of Backer and Keil [5] which was reported to run in $O(k \log^{d-2} n)$ time, the algorithm would be much faster in the case when there are only a few maximal empty boxes (here k denotes this number). On the other hand, a $(1-\varepsilon)$-approximation of the maximum volume empty box can be computed by the algorithm of Dumitrescu and Jiang [11] in $O\left(\left(\frac{8ed}{\varepsilon^2}\right)^d \cdot n \cdot \log^d n\right)$ time; their algorithm finds an empty box whose volume is at least $(1-\varepsilon)$ of the optimal.

Let us return to the exact algorithm performing in a typical case, for instance with points randomly and uniformly distributed in a box. Datta and Soundaralakshmi claimed in [10, Lemma 2] that in 3-space the expected number of maximal empty boxes is of the order $\Theta(n \log^2 n)$. Recently, Backer and Keil [5, p. 20] acknowledged this estimate

by citing it as a generalized bound of $\Theta(n \log^{d-1} n)$ for all $d \geq 3$. This bound, if true, would make their exact algorithm, which enumerates all maximal empty boxes in order to find a maximum-volume empty box, quite attractive in a typical case. Here we show that the proof given by Datta and Soundaralakshmi [10] in support of their claim does not stand. We then provide a correct proof of the generalized bound of $\Theta(n \log^{d-1} n)$ for all $d \geq 3$, and then obtain a sharper estimate.

Since the maximum-volume of an empty box is invariant under scaling, we can assume w.l.o.g. that the axis-parallel box is a hypercube. It suffices therefore to prove the following.

Theorem 1. *With n points independently and uniformly chosen in $[0,1]^d$, the expected number of maximal empty axis-aligned boxes $E(n,d)$ is $\Theta(n \log^{d-1} n)$, if d is fixed. Moreover, $E(n,d) = (1 \pm o(1))\frac{(2d-2)!}{(d-1)!}\, n \ln^{d-1} n$, if d is fixed.*

Remark. After our work has been completed, we have learned that Kaplan et al. [14] had obtained prior to us an upper bound of $O(n \log^{d-1} n)$ on $E(n,d)$; although many details in their proof are not spelled out and the arguments in [14] are quite sketchy, the proof seems to lead to an $O(n \log^{d-1} n)$ bound. The authors [14] were apparently unaware of previous work on this topic in the literature, such as [10, 20]. (The focus of their paper is range counting.) Obviously, if the results claimed by Datta and Soundaralakshmi [10] were correct, then the upper bound of $O(n \log^{d-1} n)$ obtained by Kaplan et al. [14] would *not* be a new result. Here we settle this discrepancy in the literature by acknowledging that the first correct upper bound $E(n,d) = O(n \log^{d-1} n)$ was obtained by Kaplan et al. [14]. It is also worth mentioning that the method used by Kaplan et al. [14] in deriving the upper bound is completely different than ours. We don't know if their method could be adapted to obtain the sharper estimate (with the dependence on d) that we obtain.

Our results. In Section 3 we derive an *exact* formula for this expectation (via (2) and (7)), and then obtain from it a tight asymptotic bound, $\Theta(n \log^{d-1} n)$, for a fixed d (Theorem 3). We further obtain a finer estimate (Theorem 4): for any fixed $d \geq 2$, $E(n,d) = (1 \pm o(1))\frac{(2d-2)!}{(d-1)!}\, n \ln^{d-1} n$.

Our estimates significantly sharpen previous estimates obtained by Kaplan et al. [14]: their upper bound $O(n \log^{d-1} n)$ had no matching lower bound; moreover our proof provides full details and our estimate is much more precise. In Section 2 we explore the connections between these results for maximal empty boxes and previous results [6, 15] on the expected number of maximal points and respectively, direct dominance pairs, in a set of n random points in $\mathbb{R}^d$, a problem left open by Kaplan et al. [14]. Specifically, we show that the expected number of direct dominance pairs with n random points, $\Theta(n \log^{d-1} n)$, yields the same lower bound for the expected number of maximal empty boxes amidst n random points.

Our estimates are relevant for analyzing the performance of any exact algorithm for computing the largest empty axis-parallel box amidst n points in a given axis-parallel box R, that proceeds by examining all maximal empty boxes. The current most efficient algorithm for this task, due to Backer and Keil [4], is thus expected to be much faster in instances where the points are close to being randomly distributed. Moreover, the only

approach currently known for computing the largest empty box amidst n points in a given box is by examining *all* candidates, i.e., maximal empty boxes.

2 Connections between Empty Boxes and Direct Dominance

For two points p and q in $\mathbb{R}^d$, we say that p *dominates* q if along each of the d axes the coordinate of p is larger than or equal to the coordinate of q. For a set S of points in $\mathbb{R}^d$, and a pair of points $p, q \in S$, we say that p *directly dominates* q if (i) p dominates q, and (ii) there is no other point r in S such that p dominates r and r dominates q; then (p, q) is called a *direct dominance pair*. Recall that a point (vector) is maximal if it is not dominated by any other point (vector) in the set. It is known [6] that the expected number of maximal points in a set S of n random points in $\mathbb{R}^d$ is $O(\log^{d-1} n)$ for any fixed $d \geq 2$. It is also known [15] that the expected number of direct dominance pairs in a set S of n random points in $\mathbb{R}^d$ is $\Theta(n \log^{d-1} n)$ for any fixed $d \geq 2$. In fact, one check that if the expected number of maximal points is $O(\log^{d-1} n)$, then the expected number of direct dominance pairs is $O(n \log^{d-1} n)$. This is because the points in S that are directly dominated by any point $p \in S$ are simply the maximal points among the subset $S_p \subseteq S$ of points that are dominated by p.

By symmetry, the concept of direct dominance can be generalized to include all 2^d different types, one for each combination of preferred directions along the d axes. For example, in $\mathbb{R}^2$, each point may directly dominate other points in each of its four quadrants. The expected number of such generalized direct dominance pairs is clearly still $\Theta(n \log^{d-1} n)$ for any fixed $d \geq 2$.

Datta and Soundaralakshmi [10, Lemma 2] observed that the expected number of maximal empty boxes amidst n random points in a hypercube in $\mathbb{R}^3$ is related to the number of direct dominance pairs determined by these points. Note that in $\mathbb{R}^3$, a maximal empty box may be supported by one point on each of its six faces. They argued that "once we fix the top support as a point p_i, the other five supports should be directly dominated by p_i in its four quadrants". Then, citing the previous known results [6, 15] on the expected number of direct dominance pairs, they jumped to the conclusion that the expected number of maximal empty boxes is of the same order, i.e., $\Theta(n \log^2 n)$ in $\mathbb{R}^3$. Recently, Backer and Keil [5, p. 20] acknowledged this estimate from [10] by citing it as a generalized bound of $\Theta(n \log^{d-1} n)$ in $\mathbb{R}^d$, $d \geq 3$, without a proof.

Here we show that the argument of Datta and Soundaralakshmi [10, Lemma 2] does not stand. In relating the expected number of maximal empty boxes to the expected number of direct dominance pairs, they correctly observed that each maximal empty box is associated with only a constant number (depending on d only) of direct dominance pairs, but they failed to provide any argument in the opposite direction to show that each direct dominance pair is also associated with a constant number of maximal empty boxes. For a bipartite graph with vertex partition $V = A \cup B$, the condition that every vertex in A has constant degree, without the symmetric condition that every vertex in B also has constant degree, is not sufficient to show that the number of vertices in A is of same order as the number of vertices in B.

We also note that the argument of Datta and Soundaralakshmi does not use any special property of the random distribution of the n points. Their observation that each

maximal empty box is associated with a constant number of direct dominance pairs continues to hold even for non-random point sets. As long as the points have distinct coordinates along each axis, each maximal empty box in $\mathbb{R}^3$ is supported by at most six points, one in each face. Note that the number of direct dominance pairs in any set of n points is at most the total number of pairs, i.e., $\binom{n}{2} = O(n^2)$. If their proof were sound, then following their argument, they could go further and claim that the number of maximal empty boxes amidst any n points in $\mathbb{R}^3$ is at most $O(n^2)$. But this claim is clearly false since for any fixed $d \geq 2$, there exist n-element point sets in $\mathbb{R}^d$ (or $[0,1]^d$) with at least $\Omega(n^d)$ maximal empty boxes amidst them [4, 11, 14].

It is not difficult, however, to obtain a lower bound on the expected number of maximal empty boxes. Consider the set of direct dominance pairs determined by n random points in $\mathbb{R}^d$ with distinct coordinates along each axis. Then each direct dominance pair (p, q) determines an empty box B with the two points p and q at the two opposite vertices of a main diagonal. This empty box B can be expanded, in both directions along each of the $d-1$ axes except the first axis, to a maximal empty box B' with the two points p and q supporting its two faces orthogonal to the first axis. Then each direct dominance pair is associated with a distinct maximal empty box. Thus the number of maximal empty boxes is at least the number of direct dominance pairs. Since the expected number of direct dominance pairs is $\Theta(n \log^{d-1} n)$, it follows that the expected number of maximal empty boxes is $\Omega(n \log^{d-1} n)$. Naamad et al. [20] obtained an $O(n \log n)$ upper bound for the planar case. So for $d = 2$, we already have a tight asymptotic bound $E(n, 2) = \Theta(n \log n)$. In the next section, we show that $E(n, d) = \Theta(n \log^{d-1} n)$ for any fixed $d \geq 2$.

3 Proof of Theorem 1

In this section we derive an exact formula for the expected number of maximal empty boxes amidst n random points in $[0,1]^d$; this yields a tight asymptotic bound $\Theta(n \log^{d-1} n)$, for a fixed d. In the end we obtain a finer estimate: for any fixed $d \geq 2$, $E(n, d) = (1 \pm o(1)) \frac{(2d-2)!}{(d-1)!} n \ln^{d-1} n$.

3.1 Setup

Let $d \geq 1$ and $n \geq 2d$. Let $X_i = (x_{i,1}, \ldots, x_{i,d})$, $1 \leq i \leq n$, be n random points in the unit hypercube $U_d = [0,1]^d$ in $\mathbb{R}^d$, with independent coordinates sampled uniformly from the interval $[0,1]$. Note that with probability 1, the n points have distinct coordinates in the open interval $(0,1)$ along each axis. Without loss of generality, we will assume this condition in our analysis.

For any pair of non-negative integers a and b such that $a + b \leq d$, let $A(a, b)$ be the event that

$$\begin{aligned} x_{2j-1,j} = \min\{x_{i,j} \mid 1 \leq i \leq 2a+b\} \text{ and } x_{2j,j} = \max\{x_{i,j} \mid 1 \leq i \leq 2a+b\}, \quad \text{for } 1 \leq j \leq a \\ x_{2a+j,a+j} = \max\{x_{i,a+j} \mid 1 \leq i \leq 2a+b\}, \quad \text{for } 1 \leq j \leq b \end{aligned} \tag{1}$$

and

$$\langle X_1, \ldots, X_{2a+b} \rangle \cap \{X_{2a+b+1}, \ldots, X_n\} = \emptyset,$$

where $\langle X_1, \ldots, X_{2a+b}\rangle$ denotes the box

$$(x_{1,1}, x_{2,1})\times\cdots\times(x_{2a-1,a}, x_{2a,a})\times(0, x_{2a+1,a+1})\times\cdots\times(0, x_{2a+b,a+b})\times(0,1)^{d-a-b}.$$

Then the expected number $E(n,d)$ of maximal empty boxes is

$$E(n,d) = \sum_{a=0}^{d}\sum_{b=0}^{d-a}\binom{d}{a}\binom{d-a}{b}2^b\, G(n,a,b), \tag{2}$$

where

$$G(n,a,b) = \binom{n}{2a+b}(2a+b)!\cdot \Pr(A(a,b)) \tag{3}$$

is the expected number of maximal empty boxes supported by one point in each of $2a+b$ faces: the two opposite faces orthogonal to each of the first a coordinate axes, and the upper face orthogonal to each of the next b coordinate axes. In particular, the expected number $F(n,d)$ of maximal empty boxes supported by one point in each of the $2d$ faces is

$$F(n,d) = G(n,d,0) = \binom{n}{2d}(2d)!\cdot \Pr(A(d,0)). \tag{4}$$

It remains to calculate $\Pr(A(a,b))$. Our uses of binomial expansion and conditional probability in the following are inspired by the techniques of Klein [15] for bounding the number of directed dominance pairs among n points uniformly and randomly selected in $[0,1]^d$.

Recall that the volume of U_d is exactly 1. Thus for any fixed $X_1, \ldots, X_{2a+b}$ satisfying condition (1),

$$\Pr(X_i \notin \langle X_1, \ldots, X_{2a+b}\rangle) = 1 - \mathrm{vol}\langle X_1, \ldots, X_{2a+b}\rangle$$

for each X_i with $2a+b+1 \le i \le n$. By the independence of the points,

$$\Pr(\langle X_1, \ldots, X_{2a+b}\rangle \cap \{X_{2a+b+1}, \ldots, X_n\} = \emptyset) = \big(1-\mathrm{vol}\langle X_1, \ldots, X_{2a+b}\rangle\big)^{n-2a-b}.$$

Therefore we have

$$\begin{aligned}\Pr(A(a,b)) &= \int\cdots\int_{X_1,\ldots,X_{2a+b}\in[0,1]^d \text{ subject to (1)}} \big(1-\mathrm{vol}\langle X_1, \ldots, X_{2a+b}\rangle\big)^{n-2a-b}\,\mathrm{d}X_1\ldots\mathrm{d}X_{2a+b}\\ &= \sum_{m=0}^{n-2a-b}\binom{n-2a-b}{m}(-1)^m \int\cdots\int_{X_1,\ldots,X_{2a+b}\in[0,1]^d \text{ subject to (1)}} \big(\mathrm{vol}\langle X_1, \ldots, X_{2a+b}\rangle\big)^{m}\,\mathrm{d}X_1\ldots\mathrm{d}X_{2a+b}. \end{aligned}\tag{5}$$

For any fixed $X_1, \ldots, X_{2a+b}$ satisfying condition (1), the integrand

$$\big(\mathrm{vol}\langle X_1, \ldots, X_{2a+b}\rangle\big)^m$$

is equal to the probability that the m points $X_{2a+b+1}, \ldots, X_{2a+b+m}$ are all included in the box $\langle X_1, \ldots, X_{2a+b}\rangle$, that is,

$$x_{2j-1,j} < x_{i,j} < x_{2j,j} \text{ for all } i,j \text{ such that } 2a+b+1 \le i \le 2a+b+m,\ 1 \le j \le a.$$

and

$$x_{i,a+j} < x_{2a+j,a+j} \text{ for all } i,j \text{ such that } 2a+b+1 \le i \le 2a+b+m,\ 1 \le j \le b.$$

Thus

$$\int\cdots\int_{X_1,\ldots,X_{2a+b}\in[0,1]^d \text{ subject to (1)}} \left(\mathrm{vol}\langle X_1,\ldots,X_{2a+b}\rangle\right)^m \mathrm{d}X_1\ldots\mathrm{d}X_{2a+b}$$
$$= \int\cdots\int_{x_{2,1},\ldots,x_{2a,a},x_{2a+1,a+1},\ldots,x_{2a+b,a+b}\in[0,1]} \Pr(B_m^< \cap B_m^>)\,\mathrm{d}x_{2,1}\ldots\mathrm{d}x_{2a,a}\mathrm{d}x_{2a+1,a+1}\ldots\mathrm{d}x_{2a+b,a+b}. \quad (6)$$

where $B_m^<$ is the event that

$$x_{i,j} < x_{2j,j} \text{ for all } i,j \text{ such that } 1 \le i \le 2a+b+m,\ 1 \le j \le a,\ i \ne 2j$$

and

$$x_{i,a+j} < x_{2a+j,a+j} \text{ for all } i,j \text{ such that } 1 \le i \le 2a+b+m,\ 1 \le j \le b,\ i \ne 2a+j,$$

and $B_m^>$ is the event that

$$x_{i,j} > x_{2j-1,j} \text{ for all } i,j \text{ such that } 1 \le i \le 2a+b+m,\ 1 \le j \le a,\ i \ne 2j,\ i \ne 2j-1.$$

Observe that when $x_{2,1},\ldots,x_{2a,a},x_{2a+1,a+1},\ldots,x_{2a+b,a+b}$ are fixed, we have

1. $\Pr(B_m^<) = (x_{2,1}\ldots x_{2a,a}x_{2a+1,a+1}\ldots x_{2a+b,a+b})^{m+2a+b-1}$. This is because for each valid pair i,j in the definition of $B_m^<$, the probability that $x_{i,j} < x_{2j,j}$ is exactly $x_{2j,j}$, and the probability that $x_{i,a+j} < x_{2a+j,a+j}$ is exactly $x_{2a+j,a+j}$.
2. $\Pr(B_m^> \mid B_m^<) = (m+2a+b-1)^{-a}$. This is because for each j, $1 \le j \le a$, the probability that $x_{i,j} > x_{2j-1,j}$ for all i such that $1 \le i \le 2a+b+m$, $i \ne 2j$, $i \ne 2j-1$ is equal to the probability that $x_{2j-1,j} = \min\{x_{i,j} \mid 1 \le i \le 2a+b+m,\ i \ne 2j\}$. The latter is $1/(m+2a+b-1)$ since the $m+2a+b-1$ coordinates $x_{i,j}$ are all restricted to the same range by the event $B_m^<$.

Thus (6) gives

$$\int\cdots\int_{X_1,\ldots,X_{2a+b}\in[0,1]^d \text{ subject to (1)}} \left(\mathrm{vol}\langle X_1,\ldots,X_{2a+b}\rangle\right)^m \mathrm{d}X_1\ldots\mathrm{d}X_{2a+b}$$
$$= \int\cdots\int_{x_{2,1},\ldots,x_{2a,a},x_{2a+1,a+1},\ldots,x_{2a+b,a+b}\in[0,1]} \Pr(B_m^<)\Pr(B_m^> \mid B_m^<)\,\mathrm{d}x_{2,1}\ldots\mathrm{d}x_{2a,a}\mathrm{d}x_{2a+1,a+1}\ldots\mathrm{d}x_{2a+b,a+b}$$
$$= (m+2a+b-1)^{-a}\left(\int_{x\in[0,1]} x^{m+2a+b-1}\,\mathrm{d}x\right)^{a+b}$$
$$= (m+2a+b-1)^{-a}(m+2a+b)^{-(a+b)},$$

and (5) gives

$$\Pr(A(a,b)) = \sum_{m=0}^{n-2a-b}\binom{n-2a-b}{m}(-1)^m(m+2a+b-1)^{-a}(m+2a+b)^{-(a+b)}.$$

Substituting $m + 2a + b$ by k, we have

$$\Pr(A(a,b)) = (-1)^b \sum_{k=2a+b}^{n} \binom{n-2a-b}{k-2a-b} (-1)^k (k-1)^{-a} k^{-(a+b)}.$$

Recall (3) and (4). Thus

$$\begin{aligned} G(n,a,b) &= \binom{n}{2a+b} (2a+b)! \cdot \Pr(A(a,b)) \\ &= (-1)^b \sum_{k=2a+b}^{n} \frac{n!}{(n-k)!(k-2a-b)!} (-1)^k (k-1)^{-a} k^{-(a+b)} \\ &= (-1)^b \sum_{k=2a+b}^{n} \binom{n}{k} (-1)^k \frac{k!}{(k-2a-b)!(k-1)^a k^{a+b}}. \end{aligned} \tag{7}$$

In particular, the expected number of maximal empty boxes supported by one point in each face is

$$F(n,d) = G(n,d,0) = \sum_{k=2d}^{n} \binom{n}{k} (-1)^k \frac{k!}{(k-2d)!(k-1)^d k^d}. \tag{8}$$

It is easy to verify that $F(n,1) = n-1$. However for $d \geq 2$ it is not so easy to handle the alternating binomial sum in (8). The difficulty comes from the damping factors $1/((k-1)^d k^d)$; similar difficulties are present in the alternating binomial sum in (7).

3.2 Alternating Binomial Sums

For any $n \geq 0$, the following identity is well-known:

$$\sum_{k=0}^{n} \binom{n}{k} (-1)^k = (1-1)^n = 0. \tag{9}$$

We next derive a few other alternating binomial sums that we need. For any $d \geq 2$, define

$$R(n,d) = \sum_{k=0}^{n} \binom{n}{k} \frac{(-1)^k}{(k+1)^{d-1}}, \qquad \text{for } n \geq 0, \tag{10}$$

$$S(n,d) = \sum_{k=1}^{n} \binom{n}{k} \frac{(-1)^k}{k^{d-1}}, \qquad \text{for } n \geq 1, \tag{11}$$

$$T(n,d) = \sum_{k=2}^{n} \binom{n}{k} \frac{(-1)^k}{(k-1)^{d-1}}, \qquad \text{for } n \geq 2. \tag{12}$$

The following identity is also known; see e.g., [8, Exercise 27, p. 105].

Lemma 1. *For any* $n \geq 0$, $R(n,2) = \frac{1}{n+1}$.

We also have the following three lemmas relating $R(\cdot,\cdot)$, $S(\cdot,\cdot)$ and $T(\cdot,\cdot)$.

Lemma 2. *For any* $n \geq 1$ *and* $d \geq 2$, $S(n,d) = -\sum_{m=0}^{n-1} R(m,d)$.

Lemma 3. *For any* $n \geq 2$ *and* $d \geq 2$, $T(n,d) = -\sum_{m=1}^{n-1} S(m,d)$.

Lemma 4. *For any* $n \geq 0$ *and* $d \geq 3$, $R(n,d) = -S(n+1,d-1)/(n+1)$.

For any $n \geq 1$, let $H_n = 1 + \frac{1}{2} + \cdots + \frac{1}{n}$ denote the nth harmonic number. It is well-known that $H_n = \Theta(\log n)$. From Lemmas 1, 2, and 3, we immediately obtain the following corollaries:

Corollary 1. *For any* $n \geq 1$, $S(n,2) = -H_n$.

Corollary 2. *For any* $n \geq 2$, $T(n,2) = \sum_{m=1}^{n-1} H_m$.

By repeatedly applying Lemmas 2 and 4, we can determine the asymptotic growth rates of $R(n,d)$ and $S(n,d)$ when d is fixed:

Lemma 5. *For any fixed* $d \geq 2$, $R(n,d) = \Theta(n^{-1}\log^{d-2} n)$ *and* $-S(n,d) = \Theta(\log^{d-1} n)$.

We can now also determine the asymptotic growth rate of $T(n,d)$:

Lemma 6. *For any fixed* $d \geq 2$, $T(n,d) = \Theta(n\log^{d-1} n)$.

3.3 Base Cases for $G(n,a,b)$

We prove a sequence of lemmas of increasing difficulty, which we need later.

Lemma 7. $G(n,0,0) = 0$, $G(n,0,1) = 1$, $G(n,1,0) = n-1$.

Lemma 8. $G(n,0,2) = H_n - 1$. *In particular,* $G(n,0,2) = \Theta(\log n)$.

Lemma 9. $G(n,1,1) = n - 2H_n + 1$. *In particular,* $G(n,1,1) = \Theta(n)$.

3.4 Partial Fraction Decompositions

A rational fraction (i.e., the quotient of two polynomials with real coefficients) is *proper* if the degree of the numerator is less than the degree of the denominator. A proper rational fraction $\Phi(x)/\Psi(x)$ is called a *partial fraction* if its denominator $\Psi(x)$ is a power of an irreducible polynomial $P(x)$, that is, $\Psi(x) = P^h(x)$, $h \geq 1$. The following fundamental theorem holds [16] [17, Ch. 5]:

Any proper rational fraction $\Phi(x)/\Psi(x)$ *has a unique decomposition into a sum of partial fractions. Moreover, if all roots of* $\Psi(x)$ *are real, all numerators of the partial fractions in the decomposition are constants.*

To handle further values of $G(n,a,b)$ we need to obtain partial fraction decompositions of the damping factors in the alternating binomial sums.

Lemma 10. $G(n,2,0) = 2\sum_{m=1}^{n} H_m + 4H_n - 5n - 1$. *In particular,* $G(n,2,0) = \Theta(n \log n)$.

Proof. We use Equation (9) and Corollaries 1 and 2. According to its expression in (7),

$$\begin{aligned} G(n,2,0) &= \sum_{k=4}^{n} \binom{n}{k} (-1)^k \frac{k!}{(k-4)!(k-1)^2 k^2} \\ &= \sum_{k=4}^{n} \binom{n}{k} (-1)^k \frac{(k-3)(k-2)}{(k-1)k} \\ &= \sum_{k=2}^{n} \binom{n}{k} (-1)^k \frac{(k-3)(k-2)}{(k-1)k}. \end{aligned}$$

The partial fraction decomposition of $((k-3)(k-2))/((k-1)k)$ is $\frac{(k-3)(k-2)}{(k-1)k} = 1 - \frac{6}{k} + \frac{2}{k-1}$, hence

$$\begin{aligned} G(n,2,0) &= \sum_{k=2}^{n} \binom{n}{k} (-1)^k \left(1 - \frac{6}{k} + \frac{2}{k-1}\right) \\ &= \sum_{k=2}^{n} \binom{n}{k} (-1)^k - 6\sum_{k=2}^{n} \binom{n}{k} \frac{(-1)^k}{k} + 2\sum_{k=2}^{n} \binom{n}{k} \frac{(-1)^k}{k-1} \\ &= \left[\sum_{k=0}^{n} \binom{n}{k} (-1)^k - 6\sum_{k=1}^{n} \binom{n}{k} \frac{(-1)^k}{k} + 2\sum_{k=2}^{n} \binom{n}{k} \frac{(-1)^k}{k-1}\right] \\ &\quad - \left[\sum_{k=0}^{1} \binom{n}{k} (-1)^k - 6\sum_{k=1}^{1} \binom{n}{k} \frac{(-1)^k}{k} + 0\right] \\ &= [0 - 6S(n,2) + 2T(n,2)] - [(1-n) + 6n] \\ &= 2T(n,2) - 6S(n,2) - 5n - 1 \\ &= 2\sum_{m=1}^{n-1} H_m + 6H_n - 5n - 1 = 2\sum_{m=1}^{n} H_m + 4H_n - 5n - 1. \end{aligned}$$

Since $H_m = \Theta(\log m)$, we clearly have $G(n,2,0) = \Theta(n \log n)$. □

We are now in a position to give an exact formula for $E(n,2)$ (and $F(n,2)$):

Theorem 2. $F(n,2) = 2\sum_{m=1}^{n} H_m + 4H_n - 5n - 1$ *and* $E(n,2) = 2\sum_{m=1}^{n} H_m + n + 1$. *In particular,* $F(n,2) = \Theta(n \log n)$ *and* $E(n,2) = F(n,2) + \Theta(n) = \Theta(n \log n)$.

Proof. By Lemma 10, we have $F(n,2) = G(n,2,0) = 2\sum_{m=1}^{n} H_m + 4H_n - 5n - 1$. By Lemmas 7, 8, 9, and 10, and according to its definition in (2) we have

$$
\begin{aligned}
E(n,2) &= \sum_{a=0}^{2}\sum_{b=0}^{2-a}\binom{2}{a}\binom{2-a}{b}2^b\,G(n,a,b)\\
&= G(n,0,0)+4G(n,0,1)+2G(n,1,0)+4G(n,0,2)+4G(n,1,1)+G(n,2,0)\\
&= 0+4+2(n-1)+4(H_n-1)+4(n-2H_n+1)+2\sum_{m=1}^{n}H_m+4H_n-5n-1\\
&= 2\sum_{m=1}^{n}H_m+n+1. \qquad \square
\end{aligned}
$$

Analogous to Lemmas 7, 8, 9, and 10, we have the following general lemma:

Lemma 11. *For any fixed $a \geq 0$ and $b \geq 0$, $G(n,a,b) = \Theta(n\log^{a-1} n)$ if $a \geq 2$, and $G(n,a,b) = O(n)$ if $a = 0$ or 1.*

We are now in position to finalize our calculation for the expected number $E(n,d)$ of maximal empty boxes according to (2). By Lemma 11, $F(n,d) = G(n,d,0)$ is the dominating term. Thus we have the following theorem.

Theorem 3. *For any fixed $d \geq 2$, $F(n,d) = \Theta(n\log^{d-1} n)$ and $E(n,d) = F(n,d) + O(n\log^{d-2} n) = \Theta(n\log^{d-1} n)$.*

3.5 A More Precise Bound

We next derive a more precise bound on $E(n,d)$. For two functions f and g of n, we write $f(n) \sim g(n)$ if $f(n) = g(n)(1 \pm o(1))$. Then we have the following result.

Theorem 4. *For any fixed $d \geq 2$, $E(n,d) \sim \frac{(2d-2)!}{(d-1)!}\,n\ln^{d-1} n$.*

Proof of Theorem 4. The relation $\sim$ is clearly symmetric. Moreover, it is almost transitive, in the sense that for any fixed number r of functions $f_1,\ldots,f_r$ of n, if $f_i(n) \sim f_{i+1}(n)$ for all $i = 1,\ldots,r-1$, then $f_1(n) \sim f_r(n)$. For any fixed $d \geq 2$ we have already shown that $F(n,d) = \Theta(n\log^{d-1} n)$ and $E(n,d) = F(n,d) + O(n\log^{d-2} n)$. Thus $E(n,d) \sim F(n,d)$. A closer look at the proof of Lemma 11 shows that for any fixed $a \geq 2$ and $b \geq 0$, $G(n,a,b) \sim (2a+b-2)!\,T(n,a)$. In particular, for any fixed $d \geq 2$, $F(n,d) = G(n,d,0) \sim (2d-2)!\,T(n,d)$. To prove Theorem 4, it remains to show that $T(n,d) \sim \frac{1}{(d-1)!}\,n\ln^{d-1} n$. This is accomplished by the following two technical lemmas giving more precise bounds than Lemmas 5 and 6:

Lemma 12. *For any fixed $d \geq 2$, $R(n,d) \sim \frac{1}{(d-2)!}\,n^{-1}\ln^{d-2} n$ and $-S(n,d) \sim \frac{1}{(d-1)!}\ln^{d-1} n$.*

Lemma 13. *For any fixed $d \geq 2$, $T(n,d) \sim \frac{1}{(d-1)!}\,n\ln^{d-1} n$.*

In summary, for any fixed $d \geq 2$, we have $E(n,d) \sim F(n,d) = G(n,d,0) \sim (2d-2)!\,T(n,d) \sim \frac{(2d-2)!}{(d-1)!}\,n\ln^{d-1} n$. $\square$

Acknowledgement. We thank the anonymous reviewers for thoughtful comments.

References

1. Aggarwal, A., Suri, S.: Fast algorithms for computing the largest empty rectangle. In: Proceedings of the 3rd Annual Symposium on Computational Geometry, pp. 278–290 (1987)
2. Atallah, M., Frederickson, G.: A note on finding the maximum empty rectangle. Discrete Applied Mathematics 13, 87–91 (1986)
3. Atallah, M., Kosaraju, S.R.: An efficient algorithm for maxdominance, with applications. Algorithmica 4, 221–236 (1989)
4. Backer, J., Keil, M.: The bichromatic rectangle problem in high dimensions. In: Proceedings of the 21st Canadian Conference on Computational Geometry, pp. 157–160 (2009)
5. Backer, J., Keil, J.M.: The Mono- and Bichromatic Empty Rectangle and Square Problems in All Dimensions. In: López-Ortiz, A. (ed.) LATIN 2010. LNCS, vol. 6034, pp. 14–25. Springer, Heidelberg (2010)
6. Bentley, J.L., Kung, H.T., Schkolnick, M., Thompson, C.D.: On the average number of maxima in a set of vectors and applications. Journal of the ACM 25, 536–543 (1978)
7. Chazelle, B., Drysdale, R., Lee, D.T.: Computing the largest empty rectangle. SIAM Journal on Computing 15, 300–315 (1986)
8. Chuan-Chong, C., Khee-Meng, K.: Principles and Techniques in Combinatorics. World Scientific, Singapore (1996)
9. Datta, A.: Efficient algorithms for the largest empty rectangle problem. Information Sciences 64, 121–141 (1992)
10. Datta, A., Soundaralakshmi, S.: An efficient algorithm for computing the maximum empty rectangle in three dimensions. Information Sciences 128, 43–65 (2000)
11. Dumitrescu, A., Jiang, M.: On the largest empty axis-parallel box amidst n points. Algorithmica (2012), doi:10.1007/s00453-012-9635-5
12. Edmonds, J., Gryz, J., Liang, D., Miller, R.: Mining for empty spaces in large data sets. Theoretical Computer Science 296, 435–452 (2003)
13. Giannopoulos, P., Knauer, C., Wahlström, M., Werner, D.: Hardness of discrepancy computation and ε-net verification in high dimension. Journal of Complexity (2011), doi:10.1016/j.jco.2011.09.001
14. Kaplan, H., Rubin, N., Sharir, M., Verbin, E.: Efficient colored orthogonal range counting. SIAM Journal on Computing 38, 982–1011 (2008)
15. Klein, R.: Direct dominance of points. International Journal of Computer Mathematics 19, 225–244 (1986)
16. Kudryavtsev, L.D.: The method of undetermined coefficients. In: Hazewinkel, M. (ed.) Encyclopaedia of Mathematics. Springer (2001)
17. Kurosh, A.: Higher Algebra. Mir Publishers, Moscow (1975)
18. Marx, D.: Parameterized complexity and approximation algorithms. Computer Journal 51, 60–78 (2008)
19. McKenna, M., O'Rourke, J., Suri, S.: Finding the largest rectangle in an orthogonal polygon. In: Proceedings of the 23rd Annual Allerton Conference on Communication, Control and Computing, Urbana-Champaign, Illinois (October 1985)
20. Naamad, A., Lee, D.T., Hsu, W.-L.: On the maximum empty rectangle problem. Discrete Applied Mathematics 8, 267–277 (1984)
21. Orlowski, M.: A new algorithm for the largest empty rectangle problem. Algorithmica 5, 65–73 (1990)

Rainbow Connectivity of Sparse Random Graphs

Alan Frieze* and Charalampos E. Tsourakakis**

Department of Mathematical Sciences
Carnegie Mellon University
5000 Forbes Av., 15213
Pittsburgh, PA
U.S.A.
alan@random.math.cmu.edu, ctsourak@math.cmu.edu

Abstract. An edge colored graph G is rainbow edge connected if any two vertices are connected by a path whose edges have distinct colors. The rainbow connectivity of a connected graph G, denoted by $rc(G)$, is the smallest number of colors that are needed in order to make G rainbow connected.

In this work we study the rainbow connectivity of binomial random graphs at the connectivity threshold $p = \frac{\log n + \omega}{n}$ where $\omega = \omega(n) \to \infty$ and $\omega = o(\log n)$ and of random r-regular graphs where $r \geq 3$ is a fixed integer. Specifically, we prove that the rainbow connectivity $rc(G)$ of $G = G(n,p)$ satisfies $rc(G) \sim \max\{Z_1, diameter(G)\}$ with high probability (*whp*). Here Z_1 is the number of vertices in G whose degree equals 1 and the diameter of G is asymptotically equal to $\frac{\log n}{\log \log n}$ *whp*. Finally, we prove that the rainbow connectivity $rc(G)$ of the random r-regular graph $G = G(n,r)$ satisfies $rc(G) = O(\log^2 n)$ *whp*.

1 Introduction

Connectivity is a fundamental graph theoretic property. Recently, the concept of *rainbow connectivity* was introduced by Chartrand et al. in [5]. An edge colored graph G is rainbow edge connected if any two vertices are connected by a path whose edges have distinct colors. The rainbow connectivity $rc(G)$ of a connected graph G is the smallest number of colors that are needed in order to make G rainbow edge connected. Notice, that by definition a rainbow edge connected graph is also connected and furthermore any connected graph has a trivial edge coloring that makes it rainbow edge connected, since one may color the edges of a given spanning tree with distinct colors. Other basic facts established in [5] are that $rc(G) = 1$ if and only if G is a clique and $rc(G) = |V(G)| - 1$ if and only if G is a tree. Besides its theoretical interest, rainbow connectivity is also of interest in applied settings, such as securing sensitive information [10], transfer and networking [4].

The concept of rainbow connectivity has attracted the interest of various researchers. Chartrand et al. [5] determine the rainbow connectivity of several

* Research supported in part by NSF Grant ccf1013110.
** Research supported in part by NSF Grant ccf1013110.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 541–552, 2012.
© Springer-Verlag Berlin Heidelberg 2012

special classes of graphs, including multipartite graphs. Caro et al. [3] prove that for a connected graph G with n vertices and minimum degree δ, the rainbow connectivity satisfies $rc(G) \leq \frac{\log \delta}{\delta} n(1 + f(\delta))$, where $f(\delta)$ tends to zero as δ increases. The following simpler bound was also proved in [3], $rc(G) \leq n\frac{4 \log n+3}{\delta}$. Krivelevich and Yuster more recently [9] removed the logarithmic factor from the Caro et al. [3] upper bound. Specifically they proved that $rc(G) \leq \frac{20n}{\delta}$. It is worth noticing that due to a construction of a graph with minimum degree δ and diameter $\frac{3n}{\delta+1} - \frac{\delta+7}{\delta+1}$ by Caro et al. [3], the best upper bound one can hope for is $rc(G) \leq \frac{3n}{\delta}$.

As Caro et al. point out, the random graph setting poses several intriguing questions. Specifically, let $G = G(n, p)$ denote the binomial random graph on n vertices with edge probability p [6]. Caro et al. [3] proved that $p = \sqrt{\log n/n}$ is the sharp threshold for the property $rc(G(n, p)) \leq 2$. He and Liang [7] studied further the rainbow connectivity of random graphs. Specifically, they obtain the sharp threshold for the property $rc(G) \leq d$ where d is constant. For further results and references we refer the interested reader to the recent survey of Li and Sun [10]. In this work we look at the rainbow connectivity of the binomial graph at the connectivity threshold $p = \frac{\log n+\omega}{n}$ where $\omega = o(\log n)$. This range of values for p poses problems that cannot be tackled with the techniques developed in the aforementioned work. Rainbow connectivity has not been studied in random regular graphs to the best of our knowledge.

Let

$$L = \frac{\log n}{\log \log n} \tag{1}$$

and let $A \sim B$ denote $A = (1 + o(1))B$ as $n \to \infty$.

We establish the following theorems:

Theorem 1. *Let $G = G(n, p), p = \frac{\log n+\omega}{n}$, $\omega \to \infty, \omega = o(\log n)$. Also, let Z_1 be the number of vertices of degree 1 in G. Then, with high probability(whp),*[1]

$$rc(G) \sim \max\{Z_1, L\},$$

This theorem gives asymptotically optimal results. Our next theorem is not quite as precise.

Theorem 2. *Let $G = G(n, r)$ be a random r-regular graph where $r \geq 3$ is a fixed integer. Then, whp*

$$rc(G) = O(\log^2 n).$$

All logarithms whose base is omitted are natural. It will be clear from our proofs that the colorings in the above two theorems can be constructed in a low order polynomial time. The second theorem, while weaker, contains an unexpected use of a Markov Chain Monte-Carlo (MCMC) algorithm for randomly coloring a graph.

The paper is organized as follows: in Sections 2, 3 we prove Theorems 1, 2 respectively. Finally, in Section 4 we conclude by suggesting open problems.

[1] An event A_n holds with high probability (*whp*) if $\lim_{n\to+\infty} \mathbf{Pr}[A_n] = 1$.

2 Proof of Theorem 1

Observe first that $rc(G) \geq \max\{Z_1, diameter(G)\}$. First of all, each edge incident to a vertex of degree one must have a distinct color. Just consider a path joining two such vertices. Secondly, if the shortest distance between two vertices is ℓ then we need at least ℓ colors. Next observe that *whp* the diameter D is asymptotically equal to L, see for example [2]. We break the proof of Theorem 1 into several lemmas.

Let a vertex be *large* if $\deg(x) \geq \log n/100$ and *small* otherwise.

Lemma 1. Whp*, there do not exist two small vertices within distance at most* $3L/4$*.*

Proof. Omitted due to space considerations.

We use the notation $e[S]$ for the number of edges induced by a given set of vertices S. Notice that if a set S satisfies $e[S] \geq s + t$ where $t \geq 1$, the induced subgraph $G[S]$ has at least $t + 1$ cycles.

Lemma 2. *Fix* $t \in \mathbb{Z}^+$ *and* $0 < \alpha < 1$*. Then,* whp *there does not exist a subset* $S \subseteq [n]$*, such that* $|S| \leq \alpha t L$ *and* $e[S] \geq |S| + t$*.*

Proof. Omitted due to space considerations.

Remark 1. Let T be a rooted tree of depth at most $4L/7$ and let v be a vertex not in T, but with b neighbors in T. Let S consist of v, the neighbors of v in T plus the ancestors of these neighbors. Then $|S| \leq 4bL/7 + 1 \leq 3bL/5$ and $e[S] = |S| + b - 2$. It follows from Lemma 2 with $\alpha = 3/5$ and $t = 8$, that we must have $b \leq 10$ with probability $1 - o(n^{-3})$.

Now let

$$\epsilon = \epsilon(n) = o(1) \text{ be such that } \frac{\epsilon \log\log n}{\log 1/\epsilon} \to \infty \text{ and let } k = \epsilon L. \tag{2}$$

Here L is defined in (1) and we could take $\epsilon = 1/(\log\log n)^{1/2}$.

Lemma 3. *For all pairs of large vertices* $x, y \in [n]$ *there exists a whp subgraph* $G_{x,y}(V_{x,y}, E_{x,y})$ *with the following structure: The subgraph consists of two isomorphic vertex disjoint trees* T_x, T_y *rooted at* x, y *each of depth* k*.* T_x *and* T_y *both have a branching factor of* $\log n/101$*. If the leaves of* T_x *are* $x_1, x_2, \ldots, x_\tau, \tau \geq n^{4\epsilon/5}$ *then* $y_i = f(x_i)$ *where* f *is a natural isomporphism. Between each pair of leaves* $(x_i, y_i), i = 1, 2, \ldots, m$ *there is a path* P_i *of length* $(1 + 2\epsilon)L$*. The paths* $P_i, i = 1, 2, \ldots, m$ *are edge disjoint.*

Proof. Because we have to do this for all pairs x, y, we note without further comment that likely (resp. unlikely) events will be shown to occur with probability $1 - o(n^{-2})$ (resp. $o(n^{-2})$).

To find the subgraph we grow tree structures. Specifically, we first grow a tree from x using BFS until it reaches depth k. Then, we grow a tree starting from y

again using BFS until it reaches depth k. Finally, we grow trees from the leaves of T_x and T_y using BFS for depth $\gamma = (\frac{1}{2}+\epsilon)L$. Now we analyze these processes. Since the argument is the same we explain it in detail for T_x and we outline the differences for the other trees. We use the notation $D_i^{(\rho)}$ for the number of vertices at depth i of the BFS tree rooted at ρ.

First we grow T_x. As we grow the tree via BFS from a vertex v at depth i to vertices at depth $i+1$ certain *bad* edges from v may point to vertices already in T_x. Remark 1 shows with probability $1-o(n^{-3})$ there can be at most 10 bad edges emanating from v.

Furthermore, Lemma 1 implies that there exists at most one vertex of degree less than $\frac{\log n}{100}$ at each level *whp*. Hence, we obtain the recursion

$$D_{i+1}^{(x)} \geq \left(\frac{\log n}{100} - 10\right)\left(D_i^{(x)} - 1\right) \geq \frac{\log n}{101} D_i^{(x)}. \tag{3}$$

Therefore the number of leaves satisfies

$$D_k^{(x)} \geq \left(\frac{\log n}{101}\right)^{\epsilon L} \geq n^{4\epsilon/5}. \tag{4}$$

We can make the branching factor exactly $\frac{\log n}{101}$ by pruning. We do this so that the trees T_x are isomorphic to each other.

With a similar argument

$$D_k^{(y)} \geq n^{\frac{4}{5}\epsilon}. \tag{5}$$

The only difference is that now we also say an edge is bad if the other endpoint is in T_x. This immediately gives

$$D_{i+1}^{(y)} \geq \left(\frac{\log n}{100} - 20\right)\left(D_i^{(y)} - 1\right) \geq \frac{\log n}{101} D_i^{(y)}$$

and the required conclusion (5).

Similarly, from each leaf $x_i \in T_x$ and $y_i \in T_y$ we grow trees $\widehat{T}_{x_i}, \widehat{T}_{y_i}$ of depth $\gamma = \left(\frac{1}{2}+\epsilon\right)L$ using the same procedure and arguments as above. Remark 1 implies that there are at most 20 edges from the vertex v being explored to vertices in any of the trees already constructed. At most 10 to T_x plus any trees rooted at an x_i and another 10 for y. The numbers of leaves of each $\widehat{T}_{x_i}$ now satisfies

$$\widehat{D}_\gamma^{(x_i)} \geq \frac{\log n}{100}\left(\frac{\log n}{101}\right)^{\gamma} \geq n^{\frac{1}{2}+\frac{4}{5}\epsilon}.$$

Similarly for $\widehat{D}_\gamma^{(y_i)}$.

Observe next that BFS does not condition the edges between the leaves X_i, Y_i of the trees $\widehat{T}_{x_i}$ and $\widehat{T}_{y_i}$, i.e., we do not need to look at these edges in order to carry out our construction. On the other hand we have conditioned on the occurence of certain events to imply a certain growth rate. We handle this technicality as follows. We go through the above construction and halt if ever we find that we cannot expand by the required amount. Let **A** be the event that

we do not halt the construction i.e. we fail the conditions of Lemmas 1 or 2. We have $\mathbf{Pr}[\mathbf{A}] = 1 - o(1)$ and so,

$$\mathbf{Pr}[\exists i : e(X_i, Y_i) = 0 \mid \mathbf{A}] \leq \frac{\mathbf{Pr}[\exists i : e(X_i, Y_i) = 0]}{\mathbf{Pr}(\mathbf{A})} \leq 2n^{\frac{4\epsilon}{5}}(1-p)^{n^{1+\frac{8\epsilon}{5}}} \leq n^{-n^{\epsilon}}.$$

We conclude that *whp* there is always an edge between each X_i, Y_i and thus a path of length at most $(1+2\epsilon)L$ between each x_i, y_i.

Let $q = (1+5\epsilon)L$ be the number of available colors. We color the edges of G randomly. Specifically, we show that the probability of having a rainbow path between a pair of large vertices is at least $1 - \frac{1}{n^3}$.

Lemma 4. *Color each edge of G using one color at random from q available. Then, the probability of having at least one rainbow path between two fixed large vertices $x, y \in [n]$ is at least $1 - \frac{1}{n^3}$.*

Proof. We show that the subgraph $G_{x,y}$ contains such a path. We break our proof into two steps:

Before we proceed, we provide certain necessary definitions. Think of the process of coloring T_x, T_y as an evolutionary process that colors edges by starting from the two roots $x, f(x) = y$ until it reaches the leaves. In the following, we call a vertex u of T_x (T_y) *alive* if the path $P(x,u)$ ($P(y,u)$) from x (y) to u is rainbow, i.e., the edges have received distinct colors. We call a pair of vertices $\{u, f(u)\}$ alive, $u \in T_x, f(u) \in T_y$ if $u, f(u)$ are both *alive* and the paths $P(x,u), P(y,f(u))$ share no color. Define $A_j = |\{(u, f(u)) : (u, f(u)) \text{ is alive and depth}(u) = j\}|$ for $j = 1, .., k$.

• Step 1: Existence of at least $n^{\frac{4}{5}\epsilon}$ living pairs of leaves

Assume the pair of vertices $\{u, f(u)\}$ is *alive* where $u \in T_x, f(u) \in T_y$. It is worth noticing that $u, f(u)$ have the same depth in their trees. We are interested in the number of pairs of children $\{u_i, f(u_i)\}_{i=1,..,\log n/101}$ that will be alive after coloring the edges from depth(u) to depth$(u)+1$. A living pair $\{u_i, f(u_i)\}$ by definition has the following properties: edges $(u, u_i) \in E(T_x)$ and $(f(u), f(u_i)) \in E(T_y)$ receive two distinct colors, which are different from the set of colors used in paths $P(x,u)$ and $P(y, f(u))$. Notice the latter set of colors has cardinality $2 \times \text{depth}(u) \leq 2k$.

Let A_j be the number of living pairs at depth j. We first bound the size of A_1.

$$\mathbf{Pr}\left[A_1 \leq \frac{\log n}{200}\right] \leq 2^{\log n/200}\left(\frac{1}{q}\right)^{\log n/200} = O(n^{-\Omega(\log\log n)}). \tag{6}$$

For $j > 1$ we see that the random variable equal to the number of living pairs of children of $(u, f(u))$ stochastically dominates the random variable $X \sim \text{Bin}\left(\frac{\log n}{101}, p_0\right)$, where $p_0 = \left(1 - \frac{2k}{q}\right)^2 = \left(\frac{1+3\epsilon}{1+5\epsilon}\right)^2$. The colorings of the descendants of each live pair are independent and so we have using the Chernoff bounds

for $2 \le j \le k$,

$$\mathbf{Pr}\left[A_j < \left(\frac{\log n}{200}\right)^j p_0^{j-1} \middle| A_{j-1} \ge \left(\frac{\log n}{200}\right)^{j-1} p_0^{j-2}\right] \le \exp\left\{-\frac{1}{2}\cdot\left(\frac{99}{200}\right)^2 \cdot \frac{\log n}{101}\cdot\left(\frac{\log n}{200}\right)^{j-1} p_0^j\right\} = O(n^{-\Omega(\log\log n)}). \quad (7)$$

(6) and (7) justify assuming that $A_k \ge \left(\frac{\log n}{200}\right)^k \ge n^{\frac{4}{5}\epsilon}$.

• Step 2: Existence of rainbow paths between x, y in $G_{x,y}$

Assuming that there are $\ge n^{4\epsilon/5}$ living pairs of leaves (x_i, y_i) for vertices x, y,

$$\mathbf{Pr}(x, y \text{ are not rainbow connected}) \le \left(1 - \prod_{i=0}^{2\gamma-1}\left(1 - \frac{2k+i}{q}\right)\right)^{n^{4\epsilon/5}}.$$

But

$$\prod_{i=0}^{2\gamma-1}\left(1 - \frac{2k+i}{q}\right) \ge \left(1 - \frac{2k+2\gamma}{q}\right)^{2\gamma} = \left(\frac{\epsilon}{1+5\epsilon}\right)^{2\gamma}.$$

So

$$\mathbf{Pr}(x, y \text{ are not rainbow connected}) \le \exp\left\{-n^{4\epsilon/5}\left(\frac{\epsilon}{1+5\epsilon}\right)^{2\gamma}\right\} = \exp\left\{-n^{4\epsilon/5 - O(\log(1/\epsilon)/\log\log n)}\right\}. \quad (8)$$

Using (2) and the union bound taking (8) over all large x, y completes the proof of Lemma 4.

We now finish the proof of Theorem 1 i.e. take care of small vertices.

Proof. We showed in Lemma 4 that *whp* for any two large vertices, a random coloring results in a rainbow path joining them. We divide the small vertices into two sets: vertices of degree 1, V_1 and the vertices of degree at least 2, V_2. Suppose that our colors are $1, 2, \ldots, q$ and $V_1 = \{v_1, v_2, \ldots, v_s\}$. We begin by giving the edge incident with v_i the color i. Then we slightly modify the argument in Lemma 4. If x is the neighbor of $v_i \in V_1$ then color i cannot be used in Steps 1 and 2 of that procedure. In terms of analyis this replaces q by $(q-1)$ ($(q-2)$ if y is also a neighbor of V_1) and the argument is essentially unchanged i.e. *whp* there will be a rainbow path between each pair of large vertices. Furthermore, any path starting at v_i can only use color i once and so there will be rainbow paths between V_1 and V_1 and between V_1 and the set of large vertices.

The set V_2 is treated by using only two extra colors. Assume that Red and Blue have not been used in our coloring. Then we use Red and Blue to color

two of the edges incident to a vertex $u \in V_2$ (the remaining edges are colored arbitrarily). Suppose that $V_2 = \{w_1, w_2, \ldots, w_s\}$. Then if we want a rainbow path joining w_i, w_j where $i < j$ then we use the red edge to go to its neighbor w_i'. Then we take the already constructed rainbow path to w_j'', the neighbor of w_j via a blue edge. Then we can continue to w_j.

3 Proof of Theorem 2

We first observe that simply randomly coloring the edges of $G = G(n, r)$ with $q = n^{o(1)}$ colors will not do. This is because there will *whp* be $nq^{r-r^2} = n^{1-o(1)}$ vertices v where all edges at distance at most two from v have the same color.

We follow a similar somewhat strategy to the proof in Theorem 1. We grow small trees from each of a pair of vertices x, y and then try to connect these trees by a number of edge disjoint paths. The main difference will come from our procedure for coloring the edges. Because of the similarities, we will give a little less detail in our the common parts of our proofs.

We will use the configuration model [11] in our proofs. Let $W = [2m = rn]$ be our set of *configuration points* and let $W_i = [(i-1)r+1, ir]$, $i \in [n]$, partition W. The function $\phi : W \to [n]$ is defined by $w \in W_{\phi(w)}$. Given a pairing F (i.e. a partition of W into m pairs) we obtain a (multi-)graph G_F with vertex set $[n]$ and an edge $(\phi(u), \phi(v))$ for each $\{u, v\} \in F$. Choosing a pairing F uniformly at random from among all possible pairings Ω_W of the points of W produces a random (multi-)graph G_F. If $r = O(1)$ then any event that occurs whp in G_F will also occur whp in $G(n, r)$.

3.1 Tree Building

We will grow a Breadth First Search tree T_x from each vertex. We will grow each tree to depth

$$k = k_r = \begin{cases} 1 + \lceil \log_{r-2} \log n \rceil & r \geq 4. \\ \lceil 2 \log_2 \log n - 2 \log_2 \log_2 \log n \rceil & r = 3. \end{cases}$$

Observe that

$$T_x \text{ has at most } r(1 + (r-1) + (r-1)^2 + \cdots + (r-1)^{k-1}) = r\frac{(r-1)^k - 1}{r-1} \text{ edges.} \tag{9}$$

It is useful to observe that

Lemma 5. *Whp, no set of $s \leq \ell_1 = \frac{1}{10} \log_{r-1} n$ vertices contains more than s edges.*

Proof. Indeed,

$$\mathbf{Pr}(\exists S \subseteq [n], |S| \le \ell_1, e[S] \ge |S|+1) \le \sum_{s=3}^{\ell_1} \binom{n}{s}\binom{\binom{s}{2}}{s+1}\left(\frac{r^2}{rn-rs}\right)^{s+1} \tag{10}$$

$$\le \frac{2r\ell_1}{n}\sum_{s=3}^{\ell_1}\left(\frac{ne}{s}\cdot\frac{se}{2}\cdot\frac{2r}{n}\right)^s$$

$$\le \frac{2r\ell_1}{n}\cdot\ell_1\cdot(e^2r)^{\ell_1} = o(1). \tag{11}$$

Explanation of (10): The factor $\left(\frac{r^2}{rn-rs}\right)^{s+1}$ can be justified as follows. We can estimate

$$\mathbf{Pr}(e_1, e_2, \ldots, e_{s+1} \in E(G_F)) = \prod_{i=0}^{s}\mathbf{Pr}(e_{i+1} \in E(G_F) \mid e_1, e_2, \ldots, e_i \in E(G_F)) \le \left(\frac{r^2}{rn-rs}\right)^{s+1}$$

if we pair up the lowest index endpoint of each e_i in some arbitrary order. The fraction $\frac{r^2}{rn-rs}$ is an upper bound on the probability that this endpoint is paired with the other endpoint, regardless of previous pairings.

Denote the leaves of T_x by L_x.

Corollary 1. *Whp, $x \in [n]$ implies that $(r-1)^k \le |L_x| \le r(r-1)^{k-1}$.*

Proof. This follows from the fact that whp the vertices spanned by each T_x span at most one cycle. This in turn follows from Lemma 5.

Next let

$$V_1 = \{x : V(T_x) \text{ contains a cycle}\}.$$

Consider two vertices $x, y \in V(G)$ where $x \notin T_y$ and $y \notin T_x$. We will show that whp we can find a subgraph $G'(V', E'), V' \subseteq V, E' \subseteq E$ with similar structure to that found in Lemma 3. Here $k = k_r$ and $\gamma = \left(\frac{1}{2}+\epsilon\right)\log_{r-1} n$ for some small positive constant ϵ.

Suppose that we have constructed $i = O(\log n)$ such trees rooted at some of the of T_x. We grow the $(i+1)$st tree $\widehat{T}_z$ via BFS, ignoring edges that go into y or previously constructed trees. Let a leaf $z \in L_x$ be *bad* if we have to ignore a single edge as we construct the first ℓ_1 levels of $\widehat{T}_z$. The previously constructed trees plus y account for $O(n^{1/2+\epsilon})$ vertices, so the probability that z is bad is at most $O((r-1)^{\ell_1}n^{-1/2+\epsilon}) = O(n^{-1/3})$. This holds regardless of whichever other vertices are bad. So whp there will be at most 3 bad leaves on any T_x. Indeed, $\mathbf{Pr}(\exists x : x \text{ has } \ge 4 \text{ bad leaves}) \le n\binom{O(\log n)}{4}n^{-4/3} = o(1)$.

If a leaf is not bad then the first ℓ_1 levels produce $\Theta(n^{1/10})$ leaves. Given this, we see that whp the next $\gamma - \ell_1$ levels grow at a rate $r - 1 - o(n^{-1/25})$. Indeed, given that a level has L vertices where $n^{1/10} \le L \le n^{3/4}$, the number of vertices

in the next level dominates $Bin\left((r-1)L, 1-O\left(\frac{n^{3/4}}{n}\right)\right)$, after accounting for the configuration points used in building previous trees. We can thus assert that whp we will have that all but at most three of the leaves $L_x^* \subseteq L_x$ of T_x are roots of vertex disjoint trees $\widehat{T}_1, \widehat{T}_2, \ldots$, each with $\Theta(n^{1/2+\epsilon/2})$ leaves. The same analysis applies when we build trees $\widehat{T}'_1, \widehat{T}'_2, \ldots$, with roots at L_y.

Now the probability that there is no edge joining the leaves of $\widehat{T}_i$ to the leaves of $\widehat{T}'_j$ is at most

$$\left(1 - \frac{(r-1)\Theta(n^{1/2+\epsilon/2})}{rn}\right)^{(r-1)n^{1/2+\epsilon/2}} \leq e^{-\Omega(n^{\epsilon})}.$$

Thus whp we will succeed in finding in G_F and hence in $G = G(n,r)$, for all $x, y \in V(G_F)$, for all $u \in L_x^*, v \in L_y^*$, a path $P_{u,v}$ from u to v of length $O(\log n)$ such that if $u \neq u'$ and $v \neq v'$ then $P_{u,v}$ and $P_{u',v'}$ are edge disjoint.

3.2 Coloring the Edges

We now consider the problem of coloring the edges of G. Let H denote the line graph of G and let $\Gamma = H^{2k}$ denote the graph with the same vertex set as H and an edge between vertices e, f of Γ if there there is a path of length at most $2k$ between e and f in H. We construct a (near) random proper coloring of Γ using $q = 100\log^2 n$ colors. Since Γ has maximum degree $\Delta \leq 2(r-1)^{2k} < q/2$ we can easily achieve this in polynomial time by using Glauber Dynamics, see Jerrum [8].

Glauber Dynamics: Suppose that our color set is $[q]$ and that $Z_0, Z_1, \ldots, Z_t \in [q]^{V(\Gamma)}$ is a sequence of colorings of the vertices of graph Γ. Here Z_0 is an arbitrary coloring of Γ. Also, if $e \in V(\Gamma)$ and $N_\Gamma(e)$ is the set of neighbors of e in Γ then

$$A(Z_t, e) = \{c \in [q] : \nexists f \in N_\Gamma(e) \ s.t. \ Z_t(f) = c\}$$

is the set of colors available for re-coloring e. We obtain Z_{t+1} from Z_t as follows:

1. Choose e uniformly at random from $V(\Gamma)$, and c uniformly at random from $A(Z_t, e)$.

2. Set $Z_{t+1}(e) = c$ and for all $f \neq e$, set $Z_{t+1}(f) = Z_t(f)$.

Now $|V(\Gamma)| = |E(G)| = O(n)$ and so it follows from [8] that after $O(n\log n)$ steps we have the variation distance between Z_t and a uniform *proper* coloring of Γ is $O(n^{-10})$, say.

Suppose then that we color the edges of G using the above method. Fix a pair of vertices x, y of G. We see immediately, that no color appears twice in T_x and no color appears twice in T_y. This is because the distance between edges in T_x is at most $2k$. We have lots of paths joining x and y. We first show that we can find many paths where the set of $2k$ edges within distance k of an endpoint is rainbow colored.

Case 1: $r \geq 4$**:**
We argue now that we can find $\sigma = (r-2)^{k-1}$ leaves $u_1, u_2, \ldots, u_\tau \in T_x$ and σ leaves $v_1, v_2, \ldots, v_\tau \in T_y$ such for each i the T_x path from x to u_i and the T_y path from y to v_i do not share any colors.

Lemma 6. *Let T_1, T_2 be two vertex disjoint copies of an edge colored complete d-ary tree with ℓ levels, where $d \geq 3$. Suppose that the colorings of T_1, T_2 are both rainbow. Let $\kappa = (d-1)^\ell$. Then there exist leaves $u_1, u_2, \ldots, u_\kappa$ of T_1 and leaves $v_1, v_2, \ldots v_\kappa$ of T_2 such that the following is true: If P_i, P_i' are the paths from x to u_i in T_1 and from y to v_i in T_2 respectively, then $P_i \cup P_i'$ is rainbow colored for $i = 1, 2, \ldots, \kappa$.*

Proof. Let A_ℓ be the minimum number of rainbow path pairs that we can find. We prove that $A_\ell \geq (d-1)^\ell$ by induction on ℓ. This true trivially for $\ell = 0$. Suppose that x is incident with $x_1, x_2, \ldots, x_d$ and that the sub-tree rooted at x_i is $T_{1,i}$ for $i = 1, 2, \ldots, d$. Define y_i and $T_{2,i}$, $i = 1, 2, \ldots, d$ similarly w.r.t. y. Suppose that the color of the edge (x, x_i) is c_i for $i = 1, 2, \ldots, d$ and let $Q_x = \{c_1, c_2, \ldots, c_d\}$. Similarly, suppose that the color of the edge (y, y_i) is c_i' for $i = 1, 2, \ldots, d$ and let $Q_y = \{c_1', c_2', \ldots, c_d'\}$. Next suppose that Q_j is the set of colors in Q_x that appear on the edges $E(T_{2,j}) \cup \{(y, y_j)\}$. The sets $Q_1, Q_2, \ldots, Q_d$ are pair-wise disjoint. Similarly, suppose that Q_i' is the set of colors in Q_y that appear on the edges $E(T_{1,i}) \cup \{(x, x_i)\}$. The sets $Q_1', Q_2', \ldots, Q_d'$ are pair-wise disjoint.

Now define a bipartite graph H with vertex set $A + B = [d] + [d]$ and an edge (i, j) iff $c_i \notin Q_j$ and $c_j' \notin Q_i'$. We claim that if $S \subseteq A$ then its neighbor set $N_H(S)$ satisfies the inequality

$$d|S| - |N_H(S)| - |S| \leq |S| \cdot |N_H(S)|. \tag{12}$$

Here the LHS of (12) bounds the number of $S : N_H(S)$ edges from below. This is because there are at most $|S|$ edges missing from $S : N_H(S)$ due to $i \in S$ and $j \in N_H(S)$ and $c_i \in Q_j$. At most $|N_H(S)|$ edges are missing for similar reasons. On the other hand, $d|S|$ is the number there would be without these missing edges. The RHS of (12) is a trivial upper bound.

Re-arranging we get that

$$|N_H(S)| - |S| \geq \left\lceil \frac{(d - 2 - |S|)|S|}{|S| + 1} \right\rceil \geq -1.$$

(We get -1 when $|S| = d$).

Thus H contains a matching M of size $d-1$. Suppose without loss of generality that this matching is $(i, i), i = 1, 2, \ldots, d-1$. We know by induction that for each i we can find paths $(P_{i,j}, \widehat{P}_{i,j})$, $j = 1, 2, \ldots, (d-1)^{\ell-1}$ where $P_{i,j}$ is a root to leaf path in $T_{1,i}$ and $\widehat{P}_{i,j}$ is a root to leaf path in $T_{2,i}$ and that $P_{i,j} \cup \widehat{P}_{i,j}$ is rainbow for all i, j. Furthermore, (i, i) being an edge of H, means that the edge sets $\{(x, x_i)\} \cup E(P_{i,j}) \cup E(\widehat{P}_{i,j}) \cup \{(y, y_i\}$ are all rainbow.

When $x, y \notin V_1$ we apply this Lemma to T_x, T_y by deleting one of the r sub-trees attached to each of x, y and applying the lemma directly to the $(r-1)$-ary trees that remain. This will yield $(r-2)^k$ pairs of paths. If $x \in V_1$, we delete $r-2$ sub-trees attached to x leaving at least two $(r-1)$-ary trees of depth $k-1$ with roots adjacent to x. We can do the same at y. Let c_1, c_2 be the colors of the two edges from x to the roots of these two trees T_1, T_2. Similarly, let c_1', c_2' be the colors of the two analogous edges from y to the trees T_1', T_2'. If $c_1 \neq c_2$ and color c_1 does not appear in T_1' then we apply the lemma to T_1 and T_1'. Otherwise, we can apply the lemma to T_1 and T_2'. In both cases we obtain $(r-2)^{k-1}$ pairs of paths.

Putting $\sigma = (r-2)^{k-1} - 6$ we see that we can *whp* find σ paths $P_1, P_2, \ldots, P_\sigma$ of length $O(\log n)$ from x to y. Path P_i goes from x to a leaf $u_i \in L_x^*$ and then traverses $P(u_i, v_i)$ where $v_i \in L_y^*$.

Let $\widehat{P}_i$ be that part of P_i whose edges are not in $T_x \cup T_y$, $i = 1, 2, \ldots, \sigma$. Let X_i, Y_i be those parts of P_i that lie in T_x, T_y respectively. $X_i \cup Y_i$ is rainbow by construction. We have to argue that with sufficient probability, at least one $\widehat{P}_i$ is colored so that the whole of P_i is rainbow.

When an edge $e \in \widehat{P}_i$ is re-colored by Glauber, it has a choice of at least $q - \Delta$ colors, where $\Delta \leq (r-1)^{2k}$, regardless of the colors of the remaining edges, for any i. So, the probability that it is given the same color as any other edge of P_i^* is at most $\frac{2k+2\gamma}{q-\Delta}$, again regardless of the color of the remaining edges. So, by considering, in increasing time order, the color given at the last time each edge of W is recolored, we see that

$$\mathbf{Pr}(\nexists i : P_i \text{ is rainbow}) \leq \left(1 - \left(1 - \frac{2k+2\gamma}{q-\Delta}\right)^{2\gamma}\right)^{\sigma} \leq \left(\frac{2\gamma(2k+2\gamma)}{q-\Delta}\right)^{\sigma} = o(n^{-10}).$$

Case 2: $r = 3$**:**
When $r = 3$ we can't use $(r-2)^k$ to any effect. Instead of inducting on the trees at depth one from the roots x, y, we now induct on the trees at depth $s = 2$. The rest of the proof is omitted due to space considerations. □

4 Conclusion

In this work we have given an aymptotically tight result on the rainbow connectivity of $G = G(n,p)$ at the connectivity threshold. It is reasonable to conjecture that this could be tightened:

Conjecture: Whp, $rc(G) = \max\{Z_1, diameter(G(n,p))\}$. Our result on random regular graphs is not so tight. It is still reasonable to believe that the above conjecture also holds in this case. (Of course $Z_1 = 0$ here).

It is worth mentioning that if the degree r in Theorem 2 is allowed to grow as fast as $\log n$ then one can prove a result closer to that of Theorem 1.

References

1. Ananth, P., Nasre, M., Sarpatwar, K.: Rainbow Connectivity: Hardness and Tractability. In: IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), pp. 241–251 (2011)
2. Bollobás, B.: Random Graphs. Cambridge University Press (2001)
3. Caro, Y., Lev, A., Roditty, Y., Tuza, Z., Yuster, R.: On rainbow connection. Electronic Journal of Combinatorics 15 (2008), http://www.combinatorics.org/Volume_15/PDF/v15i1r57.pdf
4. Chakrabory, S., Fischer, E., Matsliah, A., Yuster, R.: Hardness and Algorithms for Rainbow Connection. Journal of Combinatorial Optimization 21(3) (2011)
5. Chartrand, G., Johns, G.L., McKeon, K.A., Zhang, P.: Rainbow connection in graphs. Mathematica Bohemica 133(1), 85–98 (2008), http://mb.math.cas.cz/mb133-1/8.html
6. Erdös, P., Rényi, A.: On Random Graphs I. Publicationes Mathematicae 6, 290–297 (1959)
7. He, J., Liang, H.: On rainbow-k-connectivity of random graphs. Arxiv 1012.1942v1 (2010), http://arxiv.org/abs/1012.1942v1
8. Jerrum, M.R.: A very simple algorithm for estimating the number of k-colourings of a low-degree graph. Random Structures and Algorithms 7(2), 157–165 (1995)
9. Krivelevich, M., Yuster, R.: The rainbow connection of a graph is (at most) reciprocal to its minimum degree. Journal of Graph Theory 63(3), 185–191 (2009)
10. Li, X., Sun, Y.: Rainbow connections of graphs - A survey. Arxiv 1101.5747v2 (2011), http://arxiv.org/abs/1101.5747
11. Wormald, N.C.: Models of random regular graphs. In: Surveys in Combinatorics. London Mathematical Society Lecture Note Series, vol. 276, pp. 239–298 (1999)

Invertible Zero-Error Dispersers and Defective Memory with Stuck-At Errors

Ariel Gabizon[1,⋆] and Ronen Shaltiel[2,⋆⋆]

[1] Department of Computer Science, Technion, Haifa, Israel
[2] Department of Computer Science, University of Haifa, Haifa, Israel

Abstract. Kuznetsov and Tsybakov [11] considered the problem of storing information in a memory where some cells are 'stuck' at certain values. More precisely, For $0 < r, p < 1$ we want to store a string $z \in \{0,1\}^{rn}$ in an n-bit memory $x = (x_1, \ldots, x_n)$ in which a subset $S \subseteq [n]$ of size pn are stuck at certain values $u_1, \ldots, u_{pn}$ and cannot be modified. The encoding procedure receives S, $u_1, \ldots, u_{pn}$ and z and can modify the cells outside of S. The decoding procedure should be able to recover z given x (*without having to know* S *or* $u_1, \ldots, u_{pn}$). This problem is related to, and harder than, the Write-Once-Memory (WOM) problem.

We give explicit schemes with rate $r \geq 1 - p - o(1)$ (trivially, $r \leq 1 - p$ is a lower bound). This is the first explicit scheme with asymptotically optimal rate. We are able to guarantee the same rate even if following the encoding, the memory x is corrupted in $o(\sqrt{n})$ adversarially chosen positions. This more general setup was first considered by Tsybakov [24] (see also [10,8]). and our scheme improves upon previous results.

We utilize a recent connection observed by Shpilka [21] between the WOM problem and linear seeded extractors for bit-fixing sources. We generalize this observation and show that memory schemes for stuck-at memory are equivalent to zero-error seedless dispersers for bit-fixing sources. We furthermore show that using zero-error seedless dispersers for affine sources (together with linear error correcting codes with large dual distance) allows the scheme to also handle adversarial errors.

It turns out that explicitness of the disperser is not sufficient for the explicitness of the memory scheme. We also need that the disperser is *efficiently invertible*, meaning that given an output z and the linear equations specifying a bit-fixing/affine source, one can efficiently produce a string x in the support of the source on which the disperser outputs z.

In order to construct our memory schemes, we give new constructions of zero-error seedless dispersers for bit-fixing sources and affine sources. These constructions improve upon previous work by [14,6,2,25,13] in that for sources with min-entropy k, they (i) achieve larger output length $m = (1 - o(1)) \cdot k$ whereas previous constructions did not, and (ii) are efficiently invertible, whereas previous constructions do not seem to be easily invertible.

⋆ The research leading to these results has received funding from the European Union's Seventh Framework Programme under grant agreement no. 259426 ERC Cryptography and Complexity.

⋆⋆ This research was supported by BSF grant 2010120, ISF grants 686/07 and 864/11, and ERC starting grant 279559.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 553–564, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1 Introduction

1.1 Background

Kuznetsov and Tsybakov [11] considered the problem of storing information on a defective memory with "stuck-at" errors. In this setup we have a memory $x = (x_1, \ldots, x_n)$ of n cells each storing a symbol in some finite alphabet (in this paper we will restrict attention to the Boolean alphabet). The problem is that a subset $S \subseteq [n]$ containing at most s out of the n cells are 'stuck' at a certain "defect pattern" (namely, $x|_S = u$ for some $u \in \{0,1\}^{|S|}$) and we cannot modify these cells. The goal is to store a string $z \in \{0,1\}^m$ in memory x, so that at a later point it would be possible to read x and retrieve z, *even without knowing which of the cells are stuck*. Naturally, we want m to be as large as possible (as a function of n and s). A precise definition follows.

Definition 1 (Recovering from stuck-at errors). *For positive integers $s < n$, an* (n, s)-stuck-at memory scheme *consists of*

- *a (possibly randomized) encoding function E such that given any $S \subset [n]$ with $|S| \leq s$, $u \in \{0,1\}^{|S|}$ and $z \in \{0,1\}^m$, E returns $x \in \{0,1\}^n$ with $x|_S = u$, and*
- *a decoding function $D : \{0,1\}^n \mapsto \{0,1\}^m$ such that for any $x \in \{0,1\}^n$ produced by E on inputs z, S and u as above, $D(x) = z$.*

The rate *of the scheme is defined as m/n. We say that a scheme (more precisely, a sequence of schemes with $n \mapsto \infty$) is* explicit *if D is computable in deterministic poly(n)-time and E is computable in randomized expected poly(n)-time.*[1]

Motivation for this model has come recently from Phase-Change-Memory where 'stuck-at' errors are common. It may be the case that discovering the 'stuck-at' cells will be time consuming, and that the process may ruin the written content. Thus, the assumption about only the encoder knowing the defect pattern (S, u) makes sense in such a scenario. See the introduction of [12] for more details.

Connection to the standard coding theoretic setup. It is trivial that a standard error correcting code which corrects s adversarial errors can be used to solve the case of stuck-at errors. The encoding function can simply ignore the knowledge that it has of the defect pattern (S, u) and start by encoding $z \in \{0,1\}^m$ as an n bit string x using the code, and then modify $x|_S$ so that $x|_S = u$. Decoding from stuck-at errors is then simply standard decoding.

The goal is to come up with better schemes (namely obtain schemes with better rate). We remark that an advantage of using error-correcting codes for s adversarial errors is that this approach immediately extends to handle combinations of adversarial and stuck-at errors. We will consider this combined setup later on in Section 1.6.

[1] In Definition 1 we allow the encoding function to be randomized. This makes sense in the application, and the solutions that we propose in this paper are indeed randomized. We stress that we do not assume that the decoding function has access to the coin tosses of the encoding function.

1.2 Previous Work and Our Results

It trivially holds that in any (n, s)-stuck-at memory scheme we have that $m \leq n - s$. In fact, this holds even if the decoding procedure knows the defect pattern. For simplicity, we will often set $s = pn$ for some constant p and measure the rate of a (family of) schemes as n grows. The trivial bound above gives that the rate of any (n, pn)-stuck-at memory scheme is at most $1 - p$. Kuznetsov and Tsybakov [11] showed schemes with rate approaching $1 - p$ exist by a non-constructive argument. Tsybakov [23], showed that given a linear code $C \subseteq \{0,1\}^n$ of rate r whose dual code has relative distance p, one can construct an (n, pn)-stuck-at memory scheme of rate $R = 1 - r$. Using current explicit constructions of codes, this gives a scheme of rate smaller than $1 - h(p) < 1 - p$, where h is the binary entropy function. Moreover, upper bounds on the rate of binary codes show that the method of [23] cannot give schemes with rate approaching $1 - p$, even given optimal code constructions. In this paper we construct a near optimal scheme, which comes within an additive term of $\log^{O(1)} n$ from the trivial bound.

Theorem 2 (Explicit scheme for stuck-at errors). *There exists a constant $c > 1$ such that for every function $s(n) \leq n - (\log n)^c$ we construct an explicit $(n, s(n))$-stuck-at memory scheme with $m(n) = n - s(n) - \log^c n$.*

In particular, when setting $s(n) = pn$ we obtain rate $1 - p - o(1)$ which is asymptotically optimal. This is the first explicit scheme with rate approaching $1 - p$.

Corollary 3 (Explicit asymptotically optimal scheme). *For every constant $0 < p < 1$ we construct an explicit (n, pn)-stuck-at memory scheme with rate $1 - p - o(1)$.*

1.3 Connection to Write Once Memory (WOM)

Another motivation for defective memory with stuck-at errors is the setting of "Write-Once-Memory" (abbreviated as WOM) introduced by Rivest and Shamir [16]. In this setting the memory cells $x_1, \ldots, x_n$ are initialized to the value '0', and it is possible to modify a cell from '0' to '1' but not vice-versa. The goal here is to come up with schemes that allow reusing the memory x many times, where in each round we dispose the old content and want to store new content. For concreteness let us consider the simplest two round setup: We first store some string $z_1 \in \{0,1\}^{m_1}$ in memory (by encoding it as an n bit string x) and later (when we no longer need to remember z_1) we wish to reuse the memory in order to store some string $z_2 \in \{0,1\}^{m_2}$. Note that at this phase the cells containing '1' are stuck, and we need to solve an instance of the defective memory with stuck-at errors problem.

An optimal solution to the problem of stuck-at errors immediately translates into an optimal solution to the WOM problem as follows: Identify the set of strings $z_1 \in \{0,1\}^{m_1}$ with the set of strings $x \in \{0,1\}^n$ of Hamming weight pn (by choosing p such that $m_1 = h(p) \cdot n$). At the first round, we store z_1 in memory

by storing the corresponding string $x \in \{0,1\}^n$ of Hamming weight pn (and note that we can indeed recover z_1 given x). This leaves us with an instance of the memory problem with pn stuck-at errors when we want to store $z_2 \in \{0,1\}^{m_2}$ in the second round. If we use a scheme with rate approaching $1-p$, then the induced WOM-scheme has rate approaching $h(p)+1-p$ which is known to be optimal [16] and this matches the best known explicit schemes [21].

The WOM problem seems easier than the problem of stuck-at errors in the sense that the locations of cells that are stuck at the beginning of the second round are not arbitrary (and can be chosen by how we implement the encoding in the first round). In particular, the WOM-scheme can choose a parameter $t = o(n)$ and decide not to write in the first t cells during the first round. This allows the encoding in the second round to use these cells to pass t bits of "control information" to the decoding procedure. We remark that this approach is used in many of the WOM schemes in the literature.

This approach seems less robust to changes in the model (such as added stuck-at errors or WOM with few adversarial errors) as the decoding procedure may critically depends on correctly receiving the control information. Consequently, it seems to us that the approach of this paper (and specifically the results presented later in Section 1.6 in a setup where both stuck-at and adversarial errors occur) can lead to more robust solutions to the WOM problem.

1.4 Decoding Using Zero-Error Dispersers for Bit-Fixing Sources

Shpilka's observation. The starting point for this work is a recent observation of Shpilka [21] which relates the problem of WOM to certain "linear seeded extractors for bit-fixing sources" (that we elaborate on in the full version). In addition to the WOM problem, Shpilka also considers the problem of defective memory with stuck-at errors. However, his approach is not directly suitable to this problem, and instead he solves a relaxed case in which the encoding function is allowed to transfer $t = O(\log^3 n)$ bits of "control information" to the decoding function. This can be realized if both encoding and decoding procedures have access to an additional t bit external memory which is guaranteed not to have stuck-at errors.

Loosely speaking, the reason for needing an external memory is that the encoding function needs to transfer control information (which is a "seed" for the seeded extractor) so that it will be available for the decoding procedure. As explained above, in the setup of WOM, an additional external memory is not necessary because the encoding scheme can reserve the first $t = O(\log^3 n)$ cells for passing control information to the decoding procedure.

Seedless extractors and dispersers for bit-fixing sources. We would like to use Shpilka's approach while avoiding the use of external memory. This suggests that we want to replace seeded extractors with *seedless* extractors (so that no control information needs to be passed). Indeed, our first step is to recast Shpilka's observation in the terminology of "seedless zero-error dispersers for bit-fixing

sources". We begin by defining seedless extractors and dispersers for a general class $\mathcal{C}$ of sources, and then define the class of bit-fixing sources.

Definition 4 (min-entropy and statistical distance). *Let X be a distribution over $\{0,1\}^n$. The* min-entropy *of X denoted by $H_\infty(X)$ is defined by $H_\infty(X) = min_{x\in\{0,1\}^n} \log(1/\Pr[X = x])$. Two distributions X, Y over $\{0,1\}^n$ are ϵ-*close *if for every $A \subseteq \{0,1\}^n$, $|\Pr[X \in A] - \Pr[Y \in A]| \leq \epsilon$.*

Definition 5 (Seedless extractors and dispersers). *Let $\mathcal{C}$ be a class of distributions over $\{0,1\}^n$. For $0 \leq k \leq n$ we use $\mathcal{C}_k$ to denote the class of distributions $X \in \mathcal{C}$ with $H_\infty(X) \geq k$.*

- *A function $E : \{0,1\}^n \mapsto \{0,1\}^m$ is an* extractor *for $\mathcal{C}$ with* entropy threshold *k and* error *$\epsilon \geq 0$ if for every $X \in \mathcal{C}_k$, $E(X)$ is ϵ-close to the uniform distribution on $\{0,1\}^m$.*
- *A function $D : \{0,1\}^n \mapsto \{0,1\}^m$ is a* disperser *for $\mathcal{C}$ with* entropy threshold *k and* error *$\epsilon \geq 0$ if for every $X \in \mathcal{C}_k$, $|\mathrm{Supp}(D(X))| \geq (1-\epsilon)2^m$ (where $\mathrm{Supp}(Z)$ denotes the support of the distribution Z). We say that D has zero-error if $\epsilon = 0$.*

We say that a (family of) extractors (or dispersers) is explicit if it runs in time poly(n).

The reader is referred to a survey article [19] for a tutorial on seedless extractors and dispersers. We will be interested in the family of bit-fixing sources.

Definition 6 (bit-fixing sources). *A bit-fixing source is a distribution X on $\{0,1\}^n$ such that there exists $S \subseteq [n]$ and $u \in \{0,1\}^{|S|}$ such that $X|_S$ is fixed to the value u and $X|_{[n]\setminus S}$ is uniformly distributed over $\{0,1\}^{n-|S|}$. Note that $H_\infty(X) = n - |S|$.*

Explicit memory schemes and efficiently invertible *zero-error dispersers.* We now recast Shplika's observation by noting that zero-error dispersers for bit-fixing sources with entropy threshold k that output m bits imply $(n, n-k)$-stuck-at memory schemes with rate m/n. In fact, zero-error dispersers for bit-fixing sources seem to completely capture the stuck-at problem in that the decoding procedure of any memory scheme can be shown to be a zero-error disperser. Before we state this connection, recall that our goal is to construct *explicit* schemes for stuck-at errors. Unfortunately, explicitness of the zero-error disperser is not sufficient for the induced scheme to be explicit. An additional property is needed: that the disperser is *efficiently invertible* in the sense defined below.

Definition 7 (Invertible zero-error dispersers). *Let $\mathcal{C}$ be a class of distributions over $\{0,1\}^n$. We say that $\mathcal{C}$ is* polynomially-specified *if each distribution $X \in \mathcal{C}$ is specified by a string of length poly(n). (For example, each bit-fixing source can be specified by the set $S \subseteq [n]$ and $u \in \{0,1\}^{|S|}$ that define the bit-fixing source).*

We say that a zero-error disperser D (for a polynomially specified class $\mathcal{C}$ with entropy threshold k) is efficiently invertible *if there is a randomized algorithm*

running in expected poly(n)-time that given $z \in \{0,1\}^m$ and (the specification of) a source $X \in \mathcal{C}_k$ returns $x \in \mathrm{Supp}(X)$ such that $D(x) = z$.

We now formally state a connection between zero-error dispersers for bit-fixing sources and schemes for stuck-at errors. This connection is completely straightforward (and the proof appears in the full version).

Theorem 8 (Equivalence between memory schemes and dispersers)

1. *Given a zero-error disperser $D : \{0,1\}^n \mapsto \{0,1\}^m$ for bit-fixing sources with entropy threshold k there exists an $(n, n-k)$-stuck-at memory scheme with rate m/n. Furthermore, if D is explicit and efficiently invertible then the scheme is explicit.*
2. *Given an $(n, n-k)$-stuck-at memory scheme with decoding function $D : \{0,1\}^n \mapsto \{0,1\}^m$, D is a zero-error disperser for bit-fixing sources with entropy threshold k.*

1.5 A New Construction of Invertible Zero-Error Dispersers for Bit-Fixing Sources

By Theorem 8 the problem of constructing explicit $(n, s(n))$-stuck-at memory scheme is reduced to the task of explicitly constructing an efficiently invertible zero-error disperser for bit-fixing sources with entropy threshold $k = n - s(n)$. In order to prove Theorem 2 and achieve asymptotically optimal rate, we need dispersers with output length $m = (1 - o(1)) \cdot k$. Unfortunately, no such explicit construction is known. There are two issues:

- The best explicit construction of zero-error dispersers for bit-fixing sources was given by Gabizon and Shaltiel [6], and it only achieves output length $m = \Omega(k)$. This yields schemes with very poor rate of $\Omega(1-p)$ which might be small even if $1-p$ is large. (Previous constructions [3,4,9,5,14] would also give poor[2] rate when viewed as zero-error dispersers.)
- The construction of [6] is quite complicated and do not seem to be easily invertible for large values of m.

In this paper, we give an improved explicit construction of zero-error dispersers for bit-fixing sources while handling the two issues above. Namely, whenever $k > \mathrm{polylog} n$ our construction achieves $m = (1 - o(1)) \cdot k$ and is efficiently invertible.

Theorem 9 (Zero-error disperser for bit-fixing sources). *There exists a constant $c > 1$ such that if n is large enough and $k \geq \log^c n$, there is an explicit and efficiently invertible zero-error disperser $D : \{0,1\}^n \mapsto \{0,1\}^{k - \log^c n}$ for bit-fixing sources with entropy threshold k.*

[2] An exception is the construction of Chor et. al [3] in the case of very large $k = n - o(n)$. Specifically, when $k = n - t$ [3] constructs zero-error *extractors* that output $n - O(t \cdot \log n)$ bits which is better than our Theorem 9 when $t = o(\log n)$. In fact, their construction, based on linear codes, is analogous to the scheme of [23] which is superior to our Theorem 2 when, for example, $s(n) = o(\log n)$.

Chor et al. [3] showed that zero-error extractors for bit-fixing sources do not exist in case $m > 1$ and $k < n/3$. In contrast, it is easy to show the existence of zero-error dispersers using the probabilistic method. Theorem 9 achieves output length that approaches the one given by the non-constructive argument (which gives $m = k - \log n - o(\log n)$). Plugging this construction in Theorem 8 yields Theorem 2.

1.6 Recovering from Stuck-At Errors and Adversarial Errors

Tsybakov [24] considered a more general model of defective memory where in addition to the 'stuck-at' errors, the memory can be corrupted at few (adversarially chosen) cells after the encoding. A formal definition follows.

Definition 10 (Stuck-at errors and adversarial errors). *An* (n, s, e)-stuck-at noisy memory scheme *consists of*

- *a (possibly randomized) encoding function* E *such that given any* $S \subset [n]$ *with* $|S| \leq s$, $u \in \{0,1\}^{|S|}$ *and* $z \in \{0,1\}^m$, E *returns* $x \in \{0,1\}^n$ *such that* $x|_S = u$, *and*
- *a decoding function* $D : \{0,1\}^n \mapsto \{0,1\}^m$ *such that for any* $x \in \{0,1\}^n$ *produced by* E *with input* z *(and any inputs* S *and* u *as above), and any 'noise vector'* $\xi \in \{0,1\}^n$ *of hamming weight at most* e, $D(x + \xi) = z$.

The rate *of the scheme is defined as* m/n. *We say a scheme (more precisely, a sequence of schemes with* $n \mapsto \infty$*) is* explicit *if* D *is computable in deterministic poly*(n)*-time and* E *is computable in randomized expected poly*(n)*-time.*

Note that by the discussion in Section 1.1 an error-correcting code that corrects $s + e$ adversarial errors can be used to solve this more general problem. This solution seems very expensive in case $e \ll s$ (as it treats the s stuck-at errors as adversarial) and it is possible to do better in this range.

1.7 Previous Work and Our Results

The solutions proposed in previous work (as well as our results) reduce the problem of defective memory with stuck-at and worst-case errors to constructing error-correcting codes that correct e adversarial errors. However, in all known schemes (including ours), it is required that the error correcting code has additional properties: It should be linear, and have dual distance s (meaning that the dual code should have distance at least s). We elaborate on why dual distance naturally comes up in the next section.

Let $e(\cdot), s(\cdot)$ be some integer functions and let $0 \leq r_{e,s} \leq 1$ denote the largest positive number such that there is an explicit family of linear binary codes with block length $n \to \infty$ such that: (i) the code corrects $e(n)$ adversarial errors (and in particular has distance at least $2e(n) + 1$), (ii) the dual code has distance $s(n)$, and (iii) the rate of the code approaches $r_{e,s}$ as $n \to \infty$. (Here, by explicit

family we mean that the code has encoding and decoding algorithms that run in poly(n)-time).

Kuzentsov, Kasami and Yamamura [10] proposed an $(n, s(n), e(n))$-stuck-at noisy memory scheme with rate $r_{e,s} - s(n)/n - o(1)$. However, their construction has a non-constructive component. Later, Heegard [8] made this component explicit by using partitioned linear block codes. Using current explicit constructions of binary codes, and setting $s(n) = pn$ for a constant p, an explicit $(n, pn, e(n))$-stuck-at noisy memory scheme using [8] would give rate smaller than $r_{e,pn} - h(p)$. Later work focused on improving the efficiency of the encoding and decoding procedures of [8], but not the rate [12]. In this work we give explicit schemes matching the rate guaranteed by the non-explicit argument of [10].

Theorem 11 (Explicit scheme that also handles adversarial errors). *Let $e(\cdot), s(\cdot)$ be integer functions. For sufficiently large n, we construct an explicit $(n, s(n), e(n))$-stuck-at noisy memory scheme with rate $r_{e,s} - \frac{s(n)}{n} - o(1)$.*

It may seem restricting that in addition to correcting e adversarial errors, the code needs to be linear and have large dual distance. Nevertheless, in some cases these additional properties come at no extra cost (when measuring the rate as a function of the number of adversarial errors). One such example is the Hamming code which is an explicit linear code with distance 3 that has best possible rate amongst all codes with such distance. The dual code (which is the Hadamard code) has distance $s(n) = n/2$. Altogether, this gives that for $p \leq 1/2$ and $s(n) \leq pn$, we get a scheme with rate $1 - p - o(1)$ that corrects $e(n) = 1$ adversarial errors.

We can do even better and allow $e(n) = o(\sqrt{n})$ adversarial errors for the same rate of $1-p-o(1)$ by using BCH codes. For any $e(n) = o(\sqrt{n})$, BCH codes give us an explicit family of linear codes with block length $n \mapsto \infty$ that have distance $2e(n)+1$ (which in turns allows correcting $e(n)$ adversarial errors) and dimension at least $n - \log n \cdot e(n) - 1$. The dual distance of this code is at least $n/2 - e(n) \cdot \sqrt{n}$. This translates into the following corollary.

Corollary 12. *Let $p < 1/2$ be a constant and let $e(n) = o(\sqrt{n})$ and $s(n) \leq pn$. For sufficiently large n, we construct an explicit $(n, s(n), e(n))$-stuck-at noisy memory scheme with rate $1 - p - o(1)$.*

This means that we can allow $e(n) = o(\sqrt{n})$ adversarial errors at the same rate given in Corollary 3 (except for the identity of the function hidden in the $o(1)$ term). Furthermore, as $n \to \infty$ this rate matches the trivial bound of $1 - p$ (and recall that this bound holds even without adversarial errors and when the decoding procedure knows the defect pattern).[3]

[3] The function hidden in the $o(1)$ term in Corollary 12 is $O(1/\sqrt{\log \log n})$. This is much larger than the $o(1)$ term in Corollary 3 which is $O(\log^{O(1)} n/n)$. We also mention that for our choice of parameters, $r_{e(n),pn} = 1 - O(e(n) \cdot \log n/n)$. We remark that our approach can potentially achieve rate approximately $r_{e(n),pn} - pn$, given improved explicit construction of zero-error dispersers for affine sources.

1.8 Decoding Using Zero-Error Dispersers for Affine Sources

Loosely speaking, the reason that dual distance comes up naturally in the results above is that a linear code $\mathcal{C} \subseteq \mathbb{F}_2^n$ with dual distance s has the property that for every $S \subseteq [n]$ of size s and every $u \in \{0,1\}^s$, $\mathcal{C}$ has a non-empty subset $\mathcal{C}_{S,u}$ of codewords x satisfying $x|_S = u$. This follows as the rate$(\mathcal{C}) \cdot n \times n$ generator matrix of such a code has the property that every s columns are linearly independent. Once we know that $\mathcal{C}_{S,u}$ is not empty, it follows by linearity that it is in fact quite large, as it forms an affine subspace of $\mathbb{F}_2^n$ with dimension at least rate$(\mathcal{C}) \cdot n - s$.

This suggests that given (S, u), it might be a good idea that the encoding procedure of the memory scheme encodes strings $z \in \{0,1\}^m$ by strings $x \in \mathcal{C}_{S,u}$. Such strings are consistent with the defect pattern, and they form an error correcting code (that can correct as many errors as $\mathcal{C}$ can). The advantage of this approach is that it is easy for the decoding procedure of the memory scheme to handle the adversarial errors: Upon receiving a corrupted string x' (that was derived from some $x \in \mathcal{C}_{S,u}$ by e adversarial errors) one can run the decoding algorithm of $\mathcal{C}$ to recover x. At this point, the decoding procedure of the memory scheme (which does not know S, u) needs to recover the original string z by applying some polynomial time function $f : \{0,1\}^n \to \{0,1\}^m$ on x. We would like f to have to have the following property: For every $z \in \{0,1\}^m$ and every defect pattern (S, u), there exists an $x \in \mathcal{C}_{S,u}$ such that $f(x) = z$, and furthermore that such an x can be found efficiently given S, u and z. This implies that we can use efficiently invertible zero-error dispersers for affine sources with entropy threshold rate$(\mathcal{C}) \cdot n - s$ (that we define next).

Definition 13 (Affine sources). *An affine source X is a distribution over $\{0,1\}^n$ (identified with $\mathbb{F}_2^n$) that is uniform over some affine subspace of $\mathbb{F}_2^n$.*

Note that every bit-fixing source is also an affine source, and that the class of affine sources is polynomially specified as every affine subspace can be specified by at most n affine constraints. The argument explained above gives the following theorem.

Theorem 14. *Given integers s, e and n, let $\mathcal{C} \subseteq \{0,1\}^n$ be a linear code that corrects e errors and has dual distance s. Given a zero-error disperser $f : \{0,1\}^n \to \{0,1\}^m$ for affine sources with entropy threshold rate$(\mathcal{C}) \cdot n - s$, there exists an (n, s, e)-stuck-at noisy memory scheme with rate m/n. Furthermore, if $\mathcal{C}$ has polynomial time encoding and decoding, and f is explicit and efficiently invertible, then the scheme is explicit.*

1.9 A New Construction of Invertible Zero-Error Dispersers for Affine Sources

By Theorem 14 the problem of constructing explicit $(n, s(n), e(n))$-stuck-at noisy memory scheme is reduced to the task of explicitly constructing an efficiently invertible zero-error disperser for affine sources with entropy threshold $k = r_{s,e} \cdot n - s$. Once again, we require dispersers with output length $m = (1 - o(1)) \cdot k$.

Similar to the situation for bit-fixing sources, no such explicit constructions are known. Moreover, for affine sources, there are no explicit constructions with $m = \omega(1)$ for $k = o(n/\sqrt{\log\log n})$. For $k = \Omega(n/\sqrt{\log\log n})$ there is an explicit construction that achieves $m = \Omega(k)$. This is due to by Bourgain [2] with improvements by Yehudayoff [25] and Li [13] (in fact, this construction gives an extractor which is a stronger object than a disperser). For smaller k, the best known constructions by Kopparty and Ben-Sasson [1] (which handles $k = \Omega(n^{4/5})$ and Shaltiel [18] (which handles $k \geq 2^{\log^{0.9} n}$) only achieve $m = 1$. Furthermore, as was the case for bit-fixing sources, these constructions do not seem to be easily invertible.

In this paper, we give an improved construction of zero-error dispersers for affine sources. Our construction (which uses the construction of [2,13,25]) achieves $m = (1 - o(1)) \cdot k$ and is efficiently invertible. Plugging this construction in Theorem 14 yields Theorem 11.

Theorem 15 (Zero-error disperser for affine sources). *There exists a constant $\beta > 0$ such that if n is large enough and $k \geq \frac{\beta n}{\sqrt{\log\log n}}$, there is an explicit and efficiently invertible zero-error disperser $D : \{0,1\}^n \mapsto \{0,1\}^{k - \frac{\beta n}{\sqrt{\log\log n}}}$ for affine sources with entropy threshold k.*

Note that in particular, this gives $m = (1 - o(1)) \cdot k$ for linear k. We stress that that Theorem 15 is incomparable to the results of [2,13,25]. It achieves better output length, but we only obtain a zero-error disperser and not an extractor. Obviously, the disperser of Theorem 15 is also good for bit-fixing sources. However, it is incomparable to Theorem 9 as it only works for entropy threshold $k = \Omega(n/\sqrt{\log\log n})$ whereas Theorem 9 allows entropy threshold $k = (\log n)^{O(1)}$. Moreover, the output length of Theorem 9 is superior even for large k. The inferior parameters of our zero-error disperser for affine sources (compared to the case of bit-fixing sources) is the cause for the less tight bounds that we obtain on schemes in the noisy case, as discussed in a footnote in the end of the previous section.

Organization of the paper. Due to space limitations the constructions and proofs of our zero error dispersers are deferred to the full version (which can be downloaded from the web pages of the authors).

2 Technique

In order to construct the zero-error dispersers of Theorem 9 and Theorem 15 we use the composition approach of Gabizon and Shaltiel [6] (see also [5,17]). Namely, we start with a zero-error disperser that output $m_0 = O(\log n)$ bits (for which there are explicit constructions by [14,2,25,13]) and compose it with some function $F : \{0,1\}^n \times \{0,1\}^{m_0} \to \{0,1\}^m$ to get $D : \{0,1\}^n \to \{0,1\}^m$ defined by $D(x) = F(x, D'(x))$. In the full version we give constructions of adequate functions F such that the resulting function is a zero-error disperser.

For this purpose, we compose several constructions of "linear seeded extractors" [22,15,20] and of "averaging samplers" [5] (see the survey article [7]). We also observe that if F is efficiently invertible (in a natural sense) and D' is explicit, then D is efficiently invertible. Exact details are given in the full version.

3 Conclusion and Open Problems

An interesting open problem is to improve the output length of our zero-error dispersers for affine sources. (Note that we get output length $m = k - \log^{O(1)} n$ for bit-fixing sources, and only $m = k - O(n/\sqrt{\log \log n})$ for affine sources).

Getting improvements in the case of affine sources will allow us to improve the bounds we get on $m(n)$ in the case where there are both stuck-at errors and adversarial errors. This matters especially in settings where $r_{e,s} = 1$. More specifically, for codes $\mathcal{C}$ with dimension $n - a(n)$ for $a(n) = o(n)$ (such as the Hamming and BCH codes that we use in our schemes), matching the output length we obtain for bit-fixing sources will allow us to show that $m(n) = n - a(n) - s(n) - \log^{O(1)} n$, whereas we currently achieve $m(n) = n - a(n) - s(n) - O(n/\sqrt{\log \log n})$.

If we plan to use our composition method to construct improved dispersers for affine sources, then we need to first solve the case of dispersers for affine sources with low entropy threshold. It suffices to output $m = \Theta(\log n)$ bits to "jump-start" our approach. Nevertheless, we remark that all known explicit constructions for small k [1,18] achieve only $m = 1$.

It may also be interesting to try and come up with schemes where the encoding procedure is deterministic rather than randomized.

Acknowledgements. We thank Amir Shpilka for many helpful conversations. We thank Eli Ben-Sasson and Simon Litsyn for answering our questions on coding theory. We thank the anonymous reviewers for helpful comments.

References

1. Ben-Sasson, E., Kopparty, S.: Affine dispersers from subspace polynomials. In: STOC, pp. 65–74 (2009)
2. Bourgain, J.: On the construction of affine extractors. Geometric and Functional Analysis 17(1), 33–57 (2007)
3. Chor, B., Goldreich, O., Håstad, J., Friedman, J., Rudich, S., Smolensky, R.: The bit extraction problem of t-resilient functions. In: 26th Annual Symposium on Foundations of Computer Science, pp. 396–407 (1985)
4. Cohen, A., Wigderson, A.: Dispersers, deterministic amplification, and weak random sources. In: 30th Annual Symposium on Foundations of Computer Science, pp. 14–19 (1989)
5. Gabizon, A., Raz, R., Shaltiel, R.: Deterministic extractors for bit-fixing sources by obtaining an independent seed. SICOMP: SIAM Journal on Computing 36(4), 1072–1094 (2006)

6. Gabizon, A., Shaltiel, R.: Increasing the output length of zero-error dispersers. Random Struct. Algorithms 40(1), 74–104 (2012)
7. Goldreich, O.: A Sample of Samplers: A Computational Perspective on Sampling. In: Goldreich, O. (ed.) Studies in Complexity and Cryptography. LNCS, vol. 6650, pp. 302–332. Springer, Heidelberg (2011)
8. Heegard, C.: Partitioned linear block codes for computer memory with 'stuck-at' defects. IEEE Transactions on Information Theory 29(6), 831–842 (1983)
9. Kamp, J., Zuckerman, D.: Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. SIAM J. Comput., 1231–1247 (2007)
10. Kuznetsov, A.V., Kasami, T., Yamamura, S.: An error correcting scheme for defective memory. IEEE Trans. Inform. Theory 24(6), 712–718 (1978)
11. Kuznetsov, A.V., Tsybakov, B.S.: Coding in a memory with defective cells. Probl. Peredachi Inf. 10, 52–60 (1974)
12. Lastras-Montaño, L.A., Jagmohan, A., Franceschini, M.: Algorithms for memories with stuck cells. In: ISIT, pp. 968–972 (2010)
13. Li, X.: A new approach to affine extractors and dispersers. In: IEEE Conference on Computational Complexity, pp. 137–147 (2011)
14. Rao, A.: Extractors for low-weight affine sources. In: IEEE Conference on Computational Complexity, pp. 95–101 (2009)
15. Raz, R., Reingold, O., Vadhan, S.P.: Extracting all the randomness and reducing the error in trevisan's extractors. J. Comput. Syst. Sci. 65(1), 97–128 (2002)
16. Rivest, R.L., Shamir, A.: How to reuse a "write-once" memory. Information and Control, 1–19 (1982)
17. Shaltiel, R.: How to get more mileage from randomness extractors. Random Struct. Algorithms, 157–186 (2008)
18. Shaltiel, R.: Dispersers for affine sources with sub-polynomial entropy. In: FOCS, pp. 247–256 (2011)
19. Shaltiel, R.: An Introduction to Randomness Extractors. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 21–41. Springer, Heidelberg (2011)
20. Shaltiel, R., Umans, C.: Simple extractors for all min-entropies and a new pseudorandom generator. J. ACM 52(2), 172–216 (2005)
21. Shpilka, A.: Capacity Achieving Two-Write WOM Codes. In: Fernández-Baca, D. (ed.) LATIN 2012. LNCS, vol. 7256, pp. 631–642. Springer, Heidelberg (2012)
22. Trevisan, L.: Extractors and pseudorandom generators. J. ACM 48(4), 860–879 (2001)
23. Tsybakov, B.S.: Additive group codes for defect correction. Prob. Peredachi Inf. 11(1), 111–113 (1975)
24. Tsybakov, B.S.: Defect and error correction. Prob. Peredachi Inf. 11(3), 21–30 (1975)
25. Yehudayoff, A.: Affine extractors over prime fields. Combinatorica 31(2), 245–256 (2011)

Two-Sided Error Proximity Oblivious Testing
(Extended Abstract)*

Oded Goldreich ** and Igor Shinkar

Department of Computer Science and Applied Mathematics
Weizmann Institute of Science, Rehovot, Israel
{oded.goldreich,igor.shinkar}@weizmann.ac.il

Abstract. Loosely speaking, a proximity-oblivious (property) tester is a randomized algorithm that makes a constant number of queries to a tested object and distinguishes objects that have a predetermined property from those that lack it. Specifically, for some threshold probability c, objects having the property are accepted with probability at least c, whereas objects that are ϵ-far from having the property are accepted with probability at most $c - F(\epsilon)$, where $F : (0,1] \to (0,1]$ is some fixed monotone function. (We stress that, in contrast to standard testers, a proximity-oblivious tester is not given the proximity parameter.)

The foregoing notion, introduced by Goldreich and Ron (STOC 2009), was originally defined with respect to $c = 1$, which corresponds to one-sided error (proximity-oblivious) testing. Here we study the two-sided error version of proximity-oblivious testers; that is, the (general) case of arbitrary $c \in (0,1]$. We show that, in many natural cases, two-sided error proximity-oblivious testers are more powerful than one-sided error proximity-oblivious testers; that is, many natural properties that have no one-sided error proximity-oblivious testers do have a two-sided error proximity-oblivious tester.

1 Introduction

In the last fifteen years, the area of property testing has attracted much attention (see, e.g., a couple of recent surveys [R1, R2]). Loosely speaking, property testing typically refers to sub-linear time probabilistic algorithms for deciding whether a given object has a predetermined property or is far from any object having this property. Such algorithms, called testers, obtain local views of the object by performing queries; that is, the object is seen as a function and the testers get oracle access to this function (and thus may be expected to work in time that is sub-linear in the length of the object).

The foregoing description refers to the notion of "far away" objects, which in turn presumes a notion of distance between objects as well as a parameter determining when two objects are considered to be far from one another. The

* The full version of this paper is available at http://eccc.hpi-web.de/report/2012/021

** Partially supported by the Israel Science Foundation (grant No. 1041/08).

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 565–578, 2012.
© Springer-Verlag Berlin Heidelberg 2012

latter parameter is called the proximity parameter, and is often denoted ϵ; that is, one typically requires the tester to reject with high probability any object that is ϵ-far from the property.

Needless to say, in order to satisfy the aforementioned requirement, any tester (of a reasonable property) must obtain the proximity parameter as auxiliary input (and determine its actions accordingly). A natural question, first addressed systematically by Goldreich and Ron [GR], is what does the tester do with this parameter (or how does the parameter affect the actions of the tester). A very minimal effect is exhibited by testers that, based on the value of the proximity parameter, determine the number of times that a basic test is invoked, *where the basic test is oblivious of the proximity parameter*. Such basic tests, called *proximity-oblivious testers*, are indeed at the focus of the study initiated in [GR].

Loosely speaking, a proximity-oblivious tester (POT) makes a number of queries that does not depend on the proximity parameter, but the quality of its ruling does depend on the actual distance of the tested object to the property. (A standard tester of constant error probability can be obtained by repeatedly invoking a POT for a number of times that depends on the proximity parameter.)[1]

The original presentation (in [GR]) focused on POTs that always accept objects having the property. Indeed, the setting of one-sided error probability is the most appealing and natural setting for the study of POT. Still, one can also define a meaningful notion of two-sided error probability proximity-oblivious testers (POTs) by generalizing the definition (i.e., [GR, Def. 2.2]) as follows:[2]

Definition 1.1 (POT, generalized): *Let* $\Pi = \bigcup_{n\in\mathbb{N}} \Pi_n$*, where* Π_n *contains functions defined over the domain* $[n] \stackrel{\text{def}}{=} \{1,...,n\}$*, and let* $\varrho : (0,1] \to (0,1]$ *be monotone. A* two-sided error POT with detection probability ϱ for Π *is a probabilistic oracle machine* T *that makes a constant number of queries and satisfies the following two conditions, with respect to some constant* $c \in (0,1]$*:*

1. *For every* $n \in \mathbb{N}$ *and* $f \in \Pi_n$*, it holds that* $\Pr[T^f(n)=1] \geq c$.
2. *For every* $n \in \mathbb{N}$ *and* $f : [n] \to \{0,1\}^*$ *not in* Π_n*, it holds that* $\Pr[T^f(n)=1] \leq c - \varrho(\delta_{\Pi_n}(f))$*, where* $\delta_{\Pi_n}(f) = \min_{g\in\Pi_n}\{\delta(f,g)\}$ *and* $\delta(f,g) \stackrel{\text{def}}{=} |\{x \in [n] : f(x)\neq g(x)\}|/n$.

The constant c *is called the* threshold probability (of T).

Indeed, one-sided error POTs (i.e., [GR, Def. 2.2]) are obtained as a special case by letting $c = 1$. Furthermore, for every $c \in (0,1]$, every property having a one-sided error POT also has a two-sided error POT of threshold probability c (e.g., consider a generalized POT that activates the one-sided error POT with probability c and rejects otherwise). Likewise, every property having a (two-sided error) POT, has a two-sided error POT of threshold probability 1/2.

[1] Specifically, referring to Definition 1.1, when given proximity parameter ϵ, the standard tester invokes the POT $O(1/\varrho(\epsilon)^2)$ times.

[2] For simplicity, we define POTs as making a constant number of queries, and this definition is used throughout the current work. However, as in [GR], the definition may be extended to allow the query complexity to depend on n.

Motivation. Property testing can be thought of as relating local views to global properties, where the local view is provided by the queries and the global property is the distance to a predetermined set. Proximity-oblivious testing takes this relation to an extreme by making the local view independent of the distance. In other words, it refers to the smallest local view that may provide information about the global property (i.e., the distance to a predetermined set). Hence, POTs are a natural context for the study of the relation between local views and global properties of various objects. In addition, a major concrete motivation for the study of POTs is that understanding a natural subclass of testers (i.e., those obtained via POTs) may shed light on property testing at large. This motivation was advocated in [GR], while referring to one-sided error POTs, but it extends to the generalized notion defined above.

The first question. The first question that arises is whether the latter generalization (i.e., from one-sided to two-sided error POTs) is a generalization at all (i.e., does it increase the power of POTs). This is not obvious, and for some time the first author implicitly assumed that the answer is negative. However, considering the issue seriously, one may realize that two-sided error POTs exist also for properties that have no one-sided error POT. A straightforward example is the property of Boolean functions that have at least a τ fraction of 1-values, for any constant $\tau \in (0,1)$. But this example is quite artificial and contrived, and the real question is whether there exist more natural examples. In this paper we provide a host of such examples.

Our results. The current work reports of several natural properties that have two-sided error POTs, although they have no one-sided error POTs. A partial list of such examples includes:

1. Properties of Boolean functions that refer to the fraction of 1-values (i.e., the density of the preimage of 1). Each such property is specified by a constant number of subintervals of $[0,1]$, and a function satisfies such a property if the fraction of 1-values (of the function) resides in one of these subintervals. Equivalently, this can be considered as a task of testing Boolean distributions. Namely, the class is specified by a constant number of subintervals of $[0,1]$, and consists of all 0-1 random variables X such that $\Pr[X=1]$ belongs to one of the subintervals. See Theorems 2.2 and 2.3.
2. More generally, we consider distributions with finite support. We give a characterization for classes of distributions that have a two sided-error POT. See Theorem 2.5.
3. Testing graph properties in the adjacency representation model. One class of properties refers to regular graphs of a prescribed degree and to subclasses of such regular graphs (e.g., regular graphs that consists of a collection of bicliques). Another class refers to graphs in which some fixed graph occurs for a bounded number of times (e.g., at most 1% of the vertex triplets form triangles). See Theorems 3.1 and 3.2.

It is evident that none of the foregoing properties has a one-sided error POTs. The point is showing that they all have two-sided error POTs.

The current version. This extended abstract presents only a small sample of the results that are reported in the full version of this work (which is available at `http://eccc.hpi-web.de/report/2012/021`). Likewise, due to space limitations, many of the proofs have been omitted too (and can be found in the full version of this paper).

2 Testing Properties of Distributions

As mentioned in the introduction, a simple example of a property of Boolean functions that has a (two-sided error) POT is provided by the set of all functions that have at least a τ fraction of 1-values, for any constant $\tau \in (0,1)$. In this case, the POT may query the function at a single uniformly chosen preimage and return the function's value. Indeed, every function in the foregoing set is accepted with probability at least τ, whereas every function that is ϵ-far from the set is accepted with probability at most $\tau - \epsilon$.

A more telling example refers to the set of Boolean function having a fraction of 1-values that is at least τ_1 but at most τ_2, for any $0 < \tau_1 < \tau_2 < 1$. This property has a two-sided error POT that selects uniformly two samples in the function's domain, obtains the function values on them, and accept with probability α_i if the sum of the answers equals i, where $(\alpha_0, \alpha_1, \alpha_2) = (0, 1, \frac{2(\tau_1+\tau_2-1)}{\tau_1+\tau_2})$ if $\tau_1 + \tau_2 \geq 1$, and $(\alpha_0, \alpha_1, \alpha_2) = (\frac{2(1-\tau_1-\tau_2)}{2-\tau_1-\tau_2}, 1, 0)$ otherwise.

In general, we consider properties that are each specified by a sequence of t density thresholds, denoted $\overline{\tau} = (\tau_1, ..., \tau_t) \in [0,1]^t$, such that t is even and $\tau_1 \leq \tau_2 < \cdots < \tau_{t-1} \leq \tau_t$. The corresponding property, denoted $\mathcal{B}_{\overline{\tau}}$, consists of all Boolean functions $f : [n] \to \{0,1\}$ such that for some $i \leq \lceil t/2 \rceil$ it holds that $\tau_{2i-1} \leq \Pr_{r \in [n]}[f(r)=1] \leq \tau_{2i}$.

We observe that the foregoing testing task, which refers to Boolean functions, can be reduced to testing 0-1 distributions when the tester is given several samples of the tested distribution (i.e., these samples are independently and identically distributed according to the tested distribution).[3] Specifically, the corresponding class of Boolean distributions, denoted $\mathcal{D}_{\overline{\tau}}$, consists of all 0-1 random variables X such that for some $i \leq \lceil t/2 \rceil$ it holds that $\tau_{2i-1} \leq \Pr[X=1] \leq \tau_{2i}$. Indeed, (uniformly selected) queries made to a Boolean function (when testing $\mathcal{B}_{\overline{\tau}}$) correspond to samples obtained from the tested distribution.

More generally, we will be interested in testing distributions over larger fixed-size domains. It turns out that POTs for properties of multi-valued distributions are more exceptional than their binary-valued analogues. Analogously to the case of binary distributions, where properties that correspond to intervals (bounding the probability that the outcome is 1) have POTs, it is tempting to hope that properties of ternary distributions that correspond to rectangles (bounding the probabilities of the outcomes 1 and 2 respectively) also have POTs. However, as shown in Section 2.5, this is *typically* not the case! In contrast, properties

[3] In this case, the distance between distributions is merely the standard notion of statistical distance.

of multi-valued distributions that corresponds to regions that are ellipsoids do have POTs.

2.1 A Generic Tester for Boolean Distributions and Its Analysis

A generic tester for $\mathcal{D}_{\overline{\tau}}$ obtains k (independent) samples from the tested distribution, where k may (but need not) equal t, and *outputs 1 with probability α_i if exactly i of the samples have value 1.* That is, this generic tester is parameterized by the sequence $\overline{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_k)$. The question, of course, is how many samples do we need (i.e., how is k related to t and/or to other parameters); in other words, whether it is possible to select a $(k+1)$-long sequence $\overline{\alpha}$ such that the resulting tester, denoted $T_{\overline{\alpha}}$, is a POT for $\mathcal{D}_{\overline{\tau}}$. (We shall show that $k = t$ is sufficient and necessary.) The key quantity to analyze is the probability that this tester (i.e., $T_{\overline{\alpha}}$) accepts a distribution that is 1 with probability q. This accepting probability, denoted $\mathtt{P}_{\overline{\alpha}}(q)$, satisfies

$$\mathtt{P}_{\overline{\alpha}}(q) = \sum_{i=0}^{k} \binom{k}{i} \cdot q^i (1-q)^{k-i} \cdot \alpha_i. \tag{1}$$

Indeed, the function $\mathtt{P}_{\overline{\alpha}}$ is a degree k polynomial. Noting that 0-1 distributions are determined by the probability that they assume the value 1, we associate these distributions with the corresponding probabilities (e.g., we may say that q is in $\mathcal{D}_{\overline{\tau}}$ and mean that the distribution that is 1 with probability q is in $\mathcal{D}_{\overline{\tau}}$). Thus, $T_{\overline{\alpha}}$ is a POT for $\mathcal{D}_{\overline{\tau}}$ if every distribution that is ϵ-far from $\mathcal{D}_{\overline{\tau}}$ is accepted with probability at most $c - \varrho(\epsilon)$, where $c \stackrel{\text{def}}{=} \min_{q \in \mathcal{D}_{\overline{\tau}}}\{\mathtt{P}_{\overline{\alpha}}(q)\}$ and $\varrho : (0,1] \to (0,1]$ is some monotone function.

One necessary condition for the foregoing condition to hold is that for every $i \in [t]$ it holds that $\mathtt{P}_{\overline{\alpha}}(\tau_i) = c$, because otherwise a tiny shift from some τ_i to outside $\mathcal{D}_{\overline{\tau}}$ will *not* reduce the value of $\mathtt{P}_{\overline{\alpha}}(\cdot)$ below c. Another necessary condition is that $\mathtt{P}_{\overline{\alpha}}(\cdot)$ is not a constant function. We first show that there exists a setting of $\overline{\alpha}$ for which both conditions hold (and, in particular, for $k = t$).

Proposition 2.1 (on the existence of $\overline{\tau}$ such that $\mathtt{P}_{\overline{\alpha}}$ is "good"): *For every sequence $\overline{\tau} = (\tau_1, ..., \tau_t)$ such that $0 < \tau_1 < \tau_2 < \cdots < \tau_t < 1$, there exists a sequence $\overline{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_t) \in [0,1]^{t+1}$ such that the following two conditions hold*

1. *For every $i \in [t]$, it holds that $\mathtt{P}_{\overline{\alpha}}(\tau_i) = \mathtt{P}_{\overline{\alpha}}(\tau_1)$.*
2. *The function $\mathtt{P}_{\overline{\alpha}}$ is not a constant function.*

Proof. Fixing any q, we view (1) as a linear expression in the α_i's. Thus, Condition 1 yields a system of $t-1$ linear equations in the $t+1$ variables $\alpha_0, \alpha_1, ..., \alpha_t$. This system is not contradictory, since the uniform vector, denoted $\overline{u}$, is a solution (i.e., $\overline{\alpha} = ((t+1)^{-1}, ..., (t+1)^{-1})$ satisfies $\mathtt{P}_{\overline{\alpha}}(\tau_i) = (t+1)^{-1}$). Thus, this ($t-1$ dimensional) system has also a solution that is linearly independent of $\overline{u}$. Denoting such a solution by $\overline{s}$, consider arbitrary $\beta \neq 0$ and γ such that

$\beta\overline{s} + \gamma\overline{u} \in [0,1]^{t+1} \setminus \{0^{t+1}\}$. Note that $\overline{\alpha} \stackrel{\text{def}}{=} \beta\overline{s} + \gamma\overline{u}$ satisfies the linear system and is not spanned by $\overline{u}$. To establish Condition 2, we show that only vectors $\overline{\alpha}$ that are spanned by $\overline{u}$ yield a constant function $\mathtt{P}_{\overline{\alpha}}$. To see this fact, write $\mathtt{P}_{\overline{\alpha}}(q)$ as a polynomial in q, obtaining:

$$\mathtt{P}_{\overline{\alpha}}(q) = \sum_{d=0}^{t} (-1)^d \binom{t}{d} \cdot \left(\sum_{i=0}^{d} (-1)^i \binom{d}{i} \cdot \alpha_i \right) \cdot q^d. \tag{2}$$

Hence, if $\mathtt{P}_{\overline{\alpha}}$ is a constant function, then for every $d \in [t]$ it holds that $\sum_{i=0}^{d} (-1)^i \binom{d}{i} \cdot \alpha_i = 0$, which yields a system of t linearly independent equation in $t+1$ unknowns. Thus, the only solutions to this system are vectors that are spanned by $\overline{u}$, and the claim follows. □

We next prove that the sequence $\overline{\alpha}$ guaranteed by Proposition 2.1 yields a POT for $D_{\overline{\tau}}$.

Theorem 2.2 (analysis of $T_{\overline{\alpha}}$): *For every sequence $\overline{\tau} = (\tau_1, ..., \tau_t)$ such that $0 < \tau_1 < \tau_2 < \cdots < \tau_t < 1$, there exists a sequence $\overline{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_t) \in [0,1]^{t+1}$ such that $T_{\overline{\alpha}}$ is a POT with linear detection probability for $D_{\overline{\tau}}$.*

Proof. Let $\overline{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_t) \in [0,1]^{t+1}$ be as guaranteed by Proposition 2.1. Then, the (degree t) polynomial $\mathtt{P}_{\overline{\alpha}}$ "oscillates" in $[0,1]$, while obtaining the value $\mathtt{P}_{\overline{\alpha}}(\tau_1)$ on the t points $\tau_1, \tau_2, ..., \tau_t$ (and only on these points). Thus, for every $i \in [t]$ and all sufficiently small $\epsilon > 0$, exactly one of the values $\mathtt{P}_{\overline{\alpha}}(\tau_i - \epsilon)$ and $\mathtt{P}_{\overline{\alpha}}(\tau_i + \epsilon)$ is larger than $\mathtt{P}_{\overline{\alpha}}(\tau_1)$ (and the other is smaller than it). Without loss of generality, it holds that $\mathtt{P}_{\overline{\alpha}}(q) \geq \mathtt{P}_{\overline{\alpha}}(\tau_1)$ for every q in $\mathcal{D}_{\overline{\tau}}$ and $\mathtt{P}_{\overline{\alpha}}(q) < \mathtt{P}_{\overline{\alpha}}(\tau_1)$ otherwise.[4] Furthermore, we claim that there exists a constant γ such that, for any probability q that is ϵ-far from $\mathcal{D}_{\overline{\tau}}$, it holds that $\mathtt{P}_{\overline{\alpha}}(q) \leq \mathtt{P}_{\overline{\alpha}}(\tau_1) - \gamma \cdot \epsilon$. This claim can be proved by considering the Taylor expansion of $\mathtt{P}_{\overline{\alpha}}$; specifically, expanding $\mathtt{P}_{\overline{\alpha}}(q)$ based on the value at τ_i yields

$$\mathtt{P}_{\overline{\alpha}}(q) = \mathtt{P}_{\overline{\alpha}}(\tau_i) + \mathtt{P}'_{\overline{\alpha}}(\tau_i) \cdot (q - \tau_i) + \sum_{j=2}^{t} \frac{\mathtt{P}^{(j)}_{\overline{\alpha}}(\tau_i)}{j!} \cdot (q - \tau_i)^j,$$

where $\mathtt{P}'_{\overline{\alpha}}$ is the derivative of $\mathtt{P}_{\overline{\alpha}}$ and $\mathtt{P}^{(j)}_{\overline{\alpha}}$ is the j^{th} derivative of $\mathtt{P}_{\overline{\alpha}}$. By the above, $\mathtt{P}'_{\overline{\alpha}}(\tau_i) \neq 0$ (for all $i \in [t]$). Let $v \stackrel{\text{def}}{=} \min_{i \in [t]} \{|\mathtt{P}'_{\overline{\alpha}}(\tau_i)|\} > 0$ and $w \stackrel{\text{def}}{=} \max_{i \in [t], j \geq 2} \{|\mathtt{P}^{(j)}_{\overline{\alpha}}(\tau_i)|/j!\}$. Then, for all sufficiently small $\epsilon > 0$ (say for $\epsilon \leq \min(1, v)/3w$), if $|q - \tau_i| = \epsilon$ then $\sum_{j=2}^{t} \frac{\mathtt{P}^{(j)}_{\overline{\alpha}}(\tau_i)}{j!} \cdot (q - \tau_i)^j$ is upper bounded by $\sum_{j \geq 2} w \cdot \epsilon(v/3w) \cdot (1/3)^{j-2} = v \cdot \epsilon/2$; and so, for every $i \leq \lceil t/2 \rceil$, it holds that $\mathtt{P}_{\overline{\alpha}}(\tau_{2i-1} - \epsilon) < \mathtt{P}_{\overline{\alpha}}(\tau_{2i-1}) - v \cdot \epsilon/2$ and $\mathtt{P}_{\overline{\alpha}}(\tau_{2i} + \epsilon) < \mathtt{P}_{\overline{\alpha}}(\tau_{2i}) - v \cdot \epsilon/2$. Using $\gamma = \min(1, v)/3tw$, the claim holds for all $\epsilon \leq 1$. □

[4] Otherwise, use $1 - \mathtt{P}_{\overline{\alpha}}$.

Sample optimality: We have analyzed a generic tester that uses $k = t$ samples for testing a property parameterized by t thresholds (i.e., $\overline{\tau} = (\tau_1, ..., \tau_t)$). The proof of Theorem 2.2 implies that using t samples (i.e., $k \geq t$) is necessary, because for $\overline{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_k)$ we need the (non-constant) degree k polynomial $\mathtt{P}_{\overline{\alpha}}$ to attain the same value on t points (i.e., the τ_i's).

2.2 Generalization of Theorem 2.2

So far we considered distribution classes $D_{\overline{\tau}}$ such that $\overline{\tau} = (\tau_1, ..., \tau_t)$ and $0 < \tau_1 < \tau_2 < \cdots < \tau_t < 1$. We now extend the treatment as follows.

Theorem 2.3 (Theorem 2.2, generalized): *For every sequence $\overline{\tau} = (\tau_1, ..., \tau_t) \in [0,1]^t$ such that $\tau_1 \leq \tau_2 < \tau_3 \leq \tau_4 < \cdots < \tau_{t-1} \leq \tau_t$, there exists a sequence $\overline{\alpha} = (\alpha_0, \alpha_1, ..., \alpha_t) \in [0,1]^{t+1}$ such that $T_{\overline{\alpha}}$ is a POT with quadratic detection probability for $D_{\overline{\tau}}$. Furthermore, if $\tau_{2i-1} = \tau_{2i}$ for every $i \in [\lceil t/2 \rceil]$, then $\mathtt{P}_{\overline{\alpha}}(q) = \mathtt{P}_{\overline{\alpha}}(\tau_1)$ for every q in $\mathcal{D}_{\overline{\tau}}$.*

2.3 POTs Can Test Only Intervals

In this section we show that the only testable classes of Boolean distributions are those defined by a finite collection of intervals in $[0,1]$, where intervals of length zero (i.e., points) are allowed. This means that the only properties of Boolean distribution that have a POT are those covered in Theorem 2.3.

Theorem 2.4 (characterization of Boolean distributions having a POT): *Let $\mathcal{D}_S$ be a property of Boolean distributions associated with a set $S \subseteq [0,1]$ such that $X \in \mathcal{D}_S$ if and only if $\Pr[X = 1] \in S$. Then, the property $\mathcal{D}_S$ has a POT if and only if S consists of a finite subset of subintervals of $[0,1]$.*

Proof. The "if" direction follows from Theorem 2.3. For the other direction, assume that $\mathcal{T}$ is POT for $\mathcal{D}_S$ that makes k queries. Then, for a view $\overline{b} = (b_1, \ldots, b_k) \in \{0,1\}^k$, the tester $\mathcal{T}$ accepts this view with some probability, denoted $\alpha_{\overline{b}} \in [0,1]$. Note that when testing a distribution X such that $\Pr[X = 1] = p$, the probability of seing this view is $p^{w(\overline{b})}(1-p)^{k-w(\overline{b})}$, where $w(\overline{b}) = \sum_j b_j$ denotes the number of 1's in $\overline{b}$. Hence, when given a distribution X such that $\Pr[X = 1] = p$, the acceptance probability of $\mathcal{T}$ on X is

$$\Pr[\mathcal{T} \text{ accepts } X] = \sum_{i=0}^{k} \left(\sum_{\overline{b} \in \{0,1\}^k : w(\overline{b}) = i} \alpha_{\overline{b}} \right) p^i (1-p)^{k-i},$$

which is a polynomial of degree k (in p). Thus, for every $r \in \mathbb{R}$, the set of points $p \in [0,1]$ on which the value of this polynomial is at least r equals a union of up to $\lceil (k+1)/2 \rceil$ intervals. In particular, this holds for $r = c$, where c denotes the threshold probability of $\mathcal{T}$, in which case this set of points equals the set S (because $\mathcal{T}$ is POT for $\mathcal{D}_S$). The theorem follows. □

2.4 Distributions over Larger Domains

We generalize the results from the previous section to distributions over larger (finite) domains. For $r \in \mathbb{N}$ we shall identify a distribution $\overline{q} = (q_1, \ldots, q_r)$ on $[r]$ with a point in $\Delta^{(r)}$, where

$$\Delta^{(r)} = \{(q_1, \ldots, q_r) \in [0,1]^r : \sum_{i \in [t]} q_i = 1\}. \tag{3}$$

Similarly, a class of distributions with domain $[r]$ will be identified with a subset of $\Delta^{(r)}$ in a natural way. The special case of Boolean distributions corresponds to $r = 2$, for which $\Delta^{(2)} = \{(p, 1-p) : p \in [0,1]\}$.

The following result asserts that a class of distributions has a POT if and only if there exists a polynomial that is non-negative exactly on the points that correspond to distributions in that class. Thus, the question of whether or not there exists a POT for $\Pi \subseteq \Delta^{(r)}$ reduces to whether or not some polynomial can be non-negative on Π and negative on $\Delta^{(r)} \setminus \Pi$.

Theorem 2.5 (POT and polynomials in the context of distribution testing): *Let Π be an arbitrary class of distributions $\overline{q} = (q_1, \ldots, q_r)$ with domain $[r]$; that is, $\Pi \subseteq \Delta^{(r)}$. Then, Π has a two-sided error POT if and only if there is a polynomial $\mathtt{P} : \Delta^{(r)} \to \mathbb{R}$ such that for every distribution $\overline{q} = (q_1, \ldots, q_r) \in \Delta^{(r)}$ it holds*

$$\mathtt{P}(q_1, \ldots, q_r) \geq 0 \iff \overline{q} \in \Pi. \tag{4}$$

If the total degree of $\mathtt{P}$ is t, then Π has a two-sided error POT $\mathcal{T}_\Pi$ that makes t queries and has polynomial detection probability $\varrho(\epsilon) = \Omega(\epsilon^C)$, where $C < t^{O(r)}$.[5] *Moreover, the acceptance probability of $\mathcal{T}_\Pi$ when testing $\overline{q} \in \Delta^{(r)}$ can be written as $\frac{1}{2} + \delta \cdot \mathtt{P}(q_1, \ldots, q_r)$ for some constant $\delta > 0$ that depends only on the degree of $\mathtt{P}$ and on an upper bound of the absolute value of all coefficients of $\mathtt{P}$.*

Proof. The "only if" direction is proved by using the independence of samples of the given distribution. Consider a POT $\mathcal{T}_\Pi$ for Π, that makes t sampling queries and accepts each distribution in Π with probability at least c. When testing $\overline{q} = (q_1, \ldots, q_r)$, for every view $\overline{v} = (v_1, \ldots, v_t) \in [r]^t$, the probability of seeing this view is $\prod_{i=1}^t q_{v_i}$. Denoting by $\alpha_{\overline{v}}$ the probability that the tester accepts the view $\overline{v} = (v_1, \ldots, v_t)$, we have

$$\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}] = \sum_{\overline{v} = (v_1, \ldots, v_t) \in [r]^t} \left(\prod_{i=1}^t q_{v_i} \right) \cdot \alpha_{\overline{v}}. \tag{5}$$

Define a polynomial $\mathtt{P}$ such that $\mathtt{P}(q_1, \ldots, q_r)$ equals the r.h.s of (5) minus c. Then, by definition of the tester, $\mathtt{P}$ satisfies (4).

[5] The constant in the $\Omega()$ notation depends on $\mathtt{P}$, while the $O()$ notation hides some absolute constant.

For the other direction, let $\mathtt{P} : \Delta^{(r)} \to \mathbb{R}$ be a polynomial of degree t. We show that the class

$$\Pi = \{(q_1, \ldots, q_r) \in \Delta^{(r)} : \mathtt{P}(q_1, \ldots, q_r) \geq 0\} \tag{6}$$

has a POT, that makes t queries, and has polynomial detection probability.

In order to simplify the proof, we shall slightly modify $\mathtt{P}$, while making sure that the modifications of $\mathtt{P}$ does not affect Π in (6). Specifically, we multiply each monomial of degree $d < t$ (of $\mathtt{P}$) by $(\sum_{i\in[r]} q_i)^{t-d}$. This does not change the value of $\mathtt{P}$ in $\Delta^{(r)}$, and hence does not affect Π.[6] Henceforth we shall assume that $\mathtt{P}$ is a homogeneous polynomial of degree t, and therefore can be written as

$$\mathtt{P}(q_1, \ldots, q_r) = \sum_{\overline{v}\in[r]^t} \alpha_{\overline{v}} \prod_{i=1}^{t} q_{v_i} \tag{7}$$

for some coefficients $\alpha_{\overline{v}} \in \mathbb{R}$.

Assume that Π is non trivial. This implies that not all coefficients $\alpha_{\overline{v}}$ are zeros. Given (7), we define a POT $\mathcal{T}_\Pi$ for Π as follows. The tester makes t queries to a given distribution, gets t samples, denoted by $\overline{v} = (v_1, \ldots, v_t)$, and accepts with probability $\beta_{\overline{v}} = \frac{1}{2} + \delta \cdot \alpha_{\overline{v}}$, where we choose $\delta = \frac{1}{2 \cdot \max\{|\alpha_{\overline{v}}| : \overline{v}\in[r]^t\}} > 0$, in order to assure that $\beta_{\overline{v}} \in [0, 1]$ for all $\overline{v}$. Therefore, when testing $\overline{q} = (q_1, \ldots, q_r)$ the acceptance probability of the test is

$$\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}] = \sum_{\overline{v}\in[r]^t} \beta_{\overline{v}} \prod_{i=1}^{t} q_{v_i} = \frac{1}{2} + \delta \cdot \left(\sum_{\overline{v}\in[r]^t} \alpha_{\overline{v}} \prod_{i=1}^{t} q_{v_i} \right),$$

and hence, by (7), the equality above becomes

$$\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}] = \frac{1}{2} + \delta \cdot \mathtt{P}(q_1, \ldots, q_r). \tag{8}$$

Next, we analyze the acceptance probability in (8). If $\overline{q} \in \Pi$, then, by (6), we have $\mathtt{P}(q_1, \ldots, q_r) \geq 0$, and therefore $\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}] \geq \frac{1}{2}$. Lastly, assume $\overline{q}$ is ϵ-far from Π. Then, in particular $\overline{q} \notin \Pi$, and hence $\mathtt{P}(q_1, \ldots, q_r) < 0$. Thus, using (8), we have $\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}] < \frac{1}{2}$. In order to prove that $\mathcal{T}_\Pi$ is a POT, we need to show that $\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}]$ is bounded below $\frac{1}{2}$ by some function that depends on ϵ. This type of result is known in real algebraic geometry as the Łojasiewicz inequality (see [BCR, Chapter 2.6]). Specifically, we use the following theorem of Solernó [Sol].

Theorem 2.6 (Effective Łojasiewicz inequality): *Let* $\mathtt{P} : \Delta^{(r)} \to \mathbb{R}$ *be a polynomial, and let*

$$\Pi = \{(p_1, \ldots, p_r) \in \Delta^{(r)} : \mathtt{P}(p_1, \ldots, p_r) \geq 0\}.$$

[6] This grouping of monomials to homogeneous monomials maps at most 2^t monomials to a single homogeneous monomials, and thus the coefficients in the $\mathtt{P}$ may grow by a factor of at most 2^t.

Assume that for $\overline{q} = (q_1, \ldots, q_r) \in \Delta^{(r)}$ *it holds* $\mathrm{dist}(\overline{q}, \Pi) = \inf\{\frac{1}{2}\sum_{i\in[r]} |q_i - p_i| : (p_1, \ldots, p_r) \in \Pi\} > \epsilon$. *Then,* $\mathtt{P}(q_1, \ldots, q_r) < -\Omega(\epsilon^C)$ *for some constant* $C < \deg(\mathtt{P})^{O(r)}$, *where the constant in the* $\Omega()$ *notation depends on* $\mathtt{P}$, *and the* $O()$ *notation hides some absolute constant.*

By applying Theorem 2.6 on (8), we conclude that if $\overline{q} \in \Delta^{(r)}$ is ϵ-far from Π, then $\Pr[\mathcal{T}_\Pi \text{ accepts } \overline{q}] < \frac{1}{2} - \Omega(\epsilon^C)$, where $C < \deg(\mathtt{P})^{O(r)}$. This completes the proof of Theorem 2.5. □

2.5 Corollaries to Theorem 2.5

As hinted upfront, Theorem 2.5 provides a tool towards proving both positive and negative results regarding the existence of POTs for various properties. We state several such corollaries for some concrete properties of interest. As suggested by Theorem 2.5 in order to show that a some property $\Pi \subseteq \Delta^{(r)}$ has a POT it is enough to construct a polynomial that is non-negative on Π and is negative in $\Delta^{(r)} \setminus \Pi$. We omit the proofs due to space limitations.

Closure under disjoint union. Recall that in the standard property testing model, as well as in one-sided error POT model, testable properties are closed under union. However, for properties of distributions with two-sided error POT, the closure under union does not hold in general. Nevertheless, if two *disjoint* classes of distributions have two-sided error POTs, then so does their union.

Corollary 2.7 (closure under disjoint union): *Let* $\Pi_1, \ldots \Pi_k$ *be disjoint classes of distributions with domain* $[r]$, *and suppose that each of the classes* Π_i *has a two-sided error POT. Then, their union* $\Pi = \cup_{i=1}^k \Pi_i$ *also has a two-sided error POT. Moreover, suppose that for each* $i \in [k]$ *the class* Π_i *has a two-sided error POT that makes* t_i *queries and has detection probability* ϱ_i. *Then, their union* $\Pi = \cup_{i\in[k]} \Pi_i$ *has a two-sided error POT that makes* $\sum_{i\in[k]} t_i$ *queries and has detection probability* $\Omega(\min\{\varrho_i : i \in [k]\})$.

Positive corollaries. Corollary 2.8 says that a property Π consisting of a single point, or, more generally, of finitely many points has a POT whose query complexity depends on the size of Π. Corollary 2.9 gives an example of infinite classes of distributions that have a POT. The example corresponds to properties whose regions are ellipsoids in $\Delta^{(r)}$.

Corollary 2.8 (finite classes of distributions have POTs): *Fix* $r \geq 2$ *and* $k \geq 2$, *and let* Π *be a property that contains exactly* k *distributions with domain* $[r]$. *Then,* Π *has a POT that makes* $2k$ *queries and has quadratic detection probability.*

Corollary 2.9 (some infinite classes of distributions that have POTs): *Let* $\overline{p} = (p_1, \ldots, p_r)$ *be a distribution, and let* $\overline{B} = (B_0; B_1, \ldots, B_r) \in \mathbb{R}^{r+1}$ *such that* $B_i > 0$ *for all* $i \geq 0$. *Define* $\Pi^{(\overline{p},\overline{B})}$ *to be a class of distributions that lie within*

an ellipsoid centered at $\overline{p} = (p_1, \ldots, p_r)$ *with radii* $(\sqrt{\frac{B_0}{B_1}}, \ldots \sqrt{\frac{B_0}{B_r}})$. *That is,* $\Pi^{(\overline{p},\overline{B})} = \{\overline{q} = (q_1, \ldots, q_r) : \sum_{i\in[r]} B_i(q_i - p_i)^2 \leq B_0\}$. *Then, the property* $\Pi^{(\overline{p},\overline{B})}$ *has a two-sided error POT that makes two queries and has linear detection probability.*

Negative corollaries. On a negative side, we show that classes of distributions that correspond to polytopes in general do not have POTs. The proof goes by showing that there is no polynomial $\mathtt{P} : \Delta^{(r)} \to \mathbb{R}$ satisfying the condition specified in Theorem 2.5.

Corollary 2.10 (in general, polytopes have no POT): *Let* $r \geq 3$. *Let* $\Pi \subset \Delta^{(r)}$ *be a non-trivial polytope*[7] *that has a vertex* $\overline{v}$ *that is internal to* $\Delta^{(r)}$ (i.e., $\overline{v}$ is not a convex combination of $\Pi \setminus \{\overline{v}\}$ and all coordinates of $\overline{v}$ are positive). *Then,* Π *does not have a POT.*

3 Graph Properties (in the Adjacency Representation Model)

Symmetric properties of Boolean functions induce graph properties (in the adjacency representation model of [GGR]), and so the statistical properties of the previous section yield analogous properties that refer to the edge densities of graphs. The question addressed in this section is whether the study of two-sided error POT can be extended to "genuine" graph properties. The first property that we consider is degree regularity.

Recall that, in the adjacency matrix model, an N-vertex graph $G = ([N], E)$ is represented by the Boolean function $g : [N] \times [N] \to \{0, 1\}$ such that $g(u, v) = 1$ if and only if u and v are adjacent in G (i.e., $\{u, v\} \in E$). Distance between graphs is measured in terms of their aforementioned representation (i.e., as the fraction of (the number of) different matrix entries (over N^2)), but occasionally we shall use the more intuitive notion of the fraction of (the number of) edges over $\binom{N}{2}$.

3.1 The Class of k-Regular Graphs

For every function $k : \mathbb{N} \to \mathbb{N}$, we consider the set $\mathcal{R}^{(k)} = \cup_{N\in\mathbb{N}} \mathcal{R}^{(k)}_N$ such that $\mathcal{R}^{(k)}_N$ is the set of all $k(N)$-regular N-vertex graphs. That is, $G \in \mathcal{R}^{(k)}_N$ if and only if G is a simple N-vertex graph in which each vertex has degree $k(N)$. Clearly, $\mathcal{R}^{(k)}$ has no one-sided error POT, provided that $0 < k(N) < N - 1$ (cf. [GR]). In contrast, we show that it has a two-sided error POT.

[7] A polytope Π is defined as an intersection of t half-spaces (corresponding to linear conditions), such that the i-th half-space is given by $H_i = \{(q_1, \ldots, q_r) \in \mathbb{R}^r : \sum_{j\in[r]} \alpha^{(i)}_j q_j \leq \beta_i\}$. A non-trivial polytope Π is a set in $\mathbb{R}^r$ of more than a single point (i.e., $|\Pi| > 1$) that satisfy a system of linear inequalities.

Theorem 3.1 (a POT for $\mathcal{R}^{(k)}$): *For every function $k : \mathbb{N} \to \mathbb{N}$ such that $k(N) = \kappa N$ for some fixed constant $\kappa \in (0,1)$, the property $\mathcal{R}^{(k)}$ has a two-sided error POT. Furthermore, all graphs in $\mathcal{R}^{(k)}$ are accepted with equal probability.*

Proof. We may assume that $N \cdot k(N)$ is an even integer (since otherwise the test may reject without making any queries). On input N and oracle access to an N-vertex graph $G = ([N], E)$, the tester sets $\tau = k(N)/N = \kappa$ and proceeds as follows.

1. Selects uniformly a vertex $s \in [N]$ and consider the Boolean function $f_s : [N] \to \{0,1\}$ such that $f_s(v) = 1$ if and only if $\{s, v\} \in E$.
2. Invokes the POT of Theorem 2.3 to test whether the function f_s has density τ; that is, it tests whether the random variable X_s defined uniformly over $[N]$ such that $X_s(v) = f_s(v)$ is in the class $\mathcal{D}_{\tau,\tau}$.
   Recall that this POT takes two samples of X_s and accepts with probability α_i when seeing i values of 1. (The values of $(\alpha_0, \alpha_1, \alpha_2)$ are set based on τ.)

The implementation of Step 2 calls for taking two samples of X_s, which amounts to selecting uniformly two vertices and checking whether or not each of them neighbors s. Thus, we make two queries to the graph G.

Turning to the analysis of the foregoing test, let $\mathtt{P}(q)$ denote the probability that the POT invoked in Step 2 accepts a random variable X such that $\Pr[X = x] = q$. Then, the probability that our graph tester accepts the graph G equals $\frac{1}{N} \cdot \sum_{s \in [N]} \mathtt{P}(d_G(s)/N)$, where $d_G(v)$ denotes the degree of vertex v in G. Thus, every $k(N)$-regular N-vertex graph G is accepted with probability $\mathtt{P}(\tau)$. As we shall show, the following claim (which improves over a similar claim in [GGR, Apdx D]) implies that every graph that is ϵ-far from $\mathcal{R}^{(k)}_N$ is accepted with probability $\mathtt{P}(\tau) - \Omega(\epsilon^2)$.

Claim 3.1.1. *If $\sum_{v \in [N]} |d_G(v) - k(N)| \leq \epsilon' \cdot N^2$, then G is $6\epsilon'$-close to $\mathcal{R}^{(k)}_N$.*

The proof of the claim is omitted here, and can be found in the full version of this paper. Note that the claim is non-trivial, since it asserts that small local discrepancies (in the vertex degrees) imply small distance to regularity. The converse is indeed trivial.

Using the claim above, we infer that if G is ϵ-far from $\mathcal{R}^{(k)}_N$, then $\sum_{v \in [N]} |d_G(v) - k(N)| > \epsilon \cdot N^2/6$. On the other hand, by Theorem 2.3, we have, for some $\gamma > 0$,

$$\begin{aligned}\frac{1}{N} \cdot \sum_{s \in [N]} \mathtt{P}(d_G(s)/N) &\leq \frac{1}{N} \cdot \sum_{s \in [N]} \left(\mathtt{P}(\tau) - \gamma \cdot ((d_G(s) - k(N))/N)^2\right) \\ &\leq \mathtt{P}(\tau) - \frac{\gamma}{N^2} \cdot \left(\frac{\sum_{s \in [N]} |d_G(s) - k(N)|}{N}\right)^2\end{aligned}$$

where the last inequality follows by the Cauchy-Schwarz inequality. Now, using $\sum_{v \in [N]} |d_G(v) - k(N)| > \epsilon \cdot N^2/6$, we conclude that G is accepted with probability at most $\mathtt{P}(\tau) - \gamma \cdot (\epsilon/6)^2$. The theorem follows. □

3.2 Bounded Density of Induced Copies

Fixing any n-vertex graph H, denote by $\rho_H(G)$ the density of H as a subgraph in G; that is, $\rho_H(G)$ is the probability that a random sample of n vertices in G induces the subgraph H. For any graph H and $\tau \in [0,1]$, we consider the graph property $\Pi_{H,\tau} \stackrel{\text{def}}{=} \{G : \rho_H(G) \leq \tau\}$; in particular, $\Pi_{H,0}$ is the class of H-free graphs. Alon *et al.* [AFKS] showed that, for some monotone function $F_n : (0,1] \to (0,1]$ if G is δ-far from the class of H-free graphs, then $\rho_H(G) > F_n(\delta)$. Here we provide a much sharper bound for the case of $\tau > 0$ (while using an elementary proof).[8]

Theorem 3.2 (distance from $\Pi_{H,\tau}$ yields $\rho_H > \tau$): *For every n-vertex graph H and $\tau > 0$, if $G = ([N], E)$ is δ-far from $\Pi_{H,\tau}$, then $\rho_H(G) > (1 + (\delta n/3)) \cdot \tau$, provided that $\delta > 6/N$.*

It follows that $\Pi_{H,\tau}$ has a two-sided error POT, which just inspects a random sample of n vertices and checks whether the induced subgraph is isomorphic to H. This POT accepts a graph in $\Pi_{H,\tau}$ with probability at least $1-\tau$, whereas it accepts any graph that is δ-far from $\Pi_{H,\tau}$ with probability at most $1-\tau-(\tau n/3)\cdot\delta$ (if $\delta > 6/N$, and with probability at most $1-\tau-(\delta/6)^n$ otherwise).

Proof. Let us consider first the case that H contains no isolated vertices. Setting $G_0 = G$, we proceed in iterations while preserving the invariant that G_i is $(\delta - 2i/N)$-far from $\Pi_{H,\tau}$. In particular, we enter the i^{th} iteration with a graph G_{i-1} not in $\Pi_{H,\tau}$, and infer that G_{i-1} contains a vertex, denoted v_i, that participates in at least $M \stackrel{\text{def}}{=} \tau \cdot \binom{N-1}{n-1}$ copies of H. We obtain a graph G_i that is $(N-1)/\binom{N}{2}$-close to G_{i-1} by omitting from G_{i-1} all edges incident at v_i. We stress that the M copies of H counted in the i^{th} iterations are different from the copies counted in the prior $i-1$ iterations, because all copies counted in the i^{th} iteration touch the vertex v_i and do not touch the vertices $v_1, ..., v_{i-1}$, since the latter vertices are isolated in G_{i-1} (whereas H contains no isolated vertices). Also note that the copies of H counted in the i^{th} iteration also occur in G, since they contain no vertex pair on which G_{i-1} differs from G. Thus, after $t \stackrel{\text{def}}{=} \lfloor \delta N/2 \rfloor$ iterations, we obtain a graph $G_t \notin \Pi_{H,\tau}$, which contain $\tau \cdot \binom{N}{n}$ copies of H that are disjoint from the $t \cdot M$ copies of H counted in the t iterations. It follows that

$$\rho_H(G) \;\geq\; \tau + t \cdot \frac{M}{\binom{N}{n}} \;=\; \tau + \lfloor \delta N/2 \rfloor \cdot \frac{n \cdot \tau}{N} \;>\; \tau + \left(\frac{\delta n}{2} - \frac{n}{N}\right) \cdot \tau$$

and the claim follows (using $\delta > 6/N$). Recall, however, that the foregoing relies on the hypothesis that H has no isolated vertices. If this hypothesis does not hold, then the complement graph of H has no isolated vertices, and we can proceed analogously. In other words, if H has an isolated vertex, then no vertex in H is connected to all the other vertices. In this case, we consider the graph G_i

[8] In contrast, the proof of Alon *et al.* [AFKS] relies on Szemeredy's Regularity Lemma.

obtained from G_{i-1} by connecting the vertex v_i to all other vertices in the graph. Also in this case, H-copies in G_i cannot touch $v_1, ..., v_{i-1}$ (this time because each vertex in $v_1, ..., v_{i-1}$ is connected to all vertices in G_{i-1}), and we can proceed as before. □

Acknowledgments. We are grateful to Dana Ron for collaboration in early stages of this research.

References

[AFKS] Alon, N., Fischer, E., Krivelevich, M., Szegedy, M.: Efficient Testing of Large Graphs. Combinatorica 20, 451–476 (2000)

[BCR] Bochnak, J., Coste, M., Roy, M.: Real Algebraic Geometry. Springer (1998)

[GGR] Goldreich, O., Goldwasser, S., Ron, D.: Property testing and its connection to learning and approximation. Journal of the ACM, 653–750 (July 1998); Extended abstract in 37th FOCS (1996)

[GR] Goldreich, O., Ron, D.: On Proximity Oblivious Testing. SIAM Journal on Computing 40(2), 534–566 (2011); Extended abstract in 41st STOC (2009)

[R1] Ron, D.: Property testing: A learning theory perspective. Foundations and Trends in Machine Learning 1(3), 307–402 (2008)

[R2] Ron, D.: Algorithmic and analysis techniques in property testing. Foundations and Trends in TCS 5(2), 73–205 (2009)

[RS] Rubinfeld, R., Sudan, M.: Robust characterization of polynomials with applications to program testing. SIAM Journal on Computing 25(2), 252–271 (1996)

[Sol] Solernó, P.: Effective Łojasiewicz Inequalities in Semialgebraic Geometry. Applicable Algebra in Engineering, Communication and Computing 2(1), 1–14 (1990)

Mirror Descent Based Database Privacy

Prateek Jain[1] and Abhradeep Thakurta[2]

[1] Microsoft Research India
prajain@microsoft.com
[2] Pennsylvania State University
azg161@cse.psu.edu

Abstract. In this paper, we focus on the problem of private database release in the interactive setting: a trusted database curator receives queries in an online manner for which it needs to respond with accurate but privacy preserving answers. To this end, we generalize the IDC (*Iterative Database Construction*) framework of [15,13] that maintains a differentially private artificial dataset and answers incoming *linear* queries using the artificial dataset. In particular, we formulate a generic IDC framework based on the Mirror Descent algorithm, a popular convex optimization algorithm [1]. We then present two concrete applications, namely, cut queries over a bipartite graph and linear queries over low-rank matrices, and provide significantly tighter error bounds than the ones by [15,13].

1 Introduction

Statistical analysis is extensively used to mine interesting information/patterns from the data. However, releasing such information can potentially compromise privacy of the individual records in the data [8,11,4], hence risk leaking sensitive information, e.g., health/financial records of a person/company.

Existing literature on privacy preserving statistical analysis studies the problem in two different settings: *interactive* and *non-interactive*. In the interactive setting, a database curator who owns a dataset (e.g. a hospital/bank) tries to answer queries about the dataset accurately (i.e., with small error), while preserving privacy of each individual in the dataset. In the non-interactive setting, the curator releases a "sanitized" version of the dataset that accurately answers all the queries in a given query class [2,9]. While non-interactive setting has been extensively explored in the literature[2,18,14,9,12], interactive-setting is relatively less-explored with most results being fairly recent [19,15,13].

In this paper, we focus on the interactive setting mentioned above, where the queries can be adaptively (and even adversarially) chosen according to past queries and their responses. For privacy, we use the notion of *differential privacy* [7,6] which is one of the most successful and theoretically sound notions, and is now being accepted as a benchmark. Intuitively, the output of a differentially private algorithm running on a dataset should be almost independent of the inclusion (or exclusion) of any individual data record. It is trivial to achieve

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 579–590, 2012.
© Springer-Verlag Berlin Heidelberg 2012

privacy by giving a response that is completely independent of the underlying data. However, such a response will have large error and hence low *utility*.

A slightly better solution is to independently add enough noise to each query response such that it nullifies the effect of any particular record in the dataset. However, to preserve privacy for k queries with this scheme, naïve analysis suggests that the error in each query scales as $O(\sqrt{k})$. In the pursuit of obtaining a better error bound than $O(\sqrt{k})$, [19] proposed an algorithm called the *median mechanism* that improves over the naïve solution, and guarantees $O(\frac{\text{poly}(\log k)}{N^{1/3}})$ error in each query response for *adaptive* queries over normalized histograms. Here N is the number of records in the database. While their result reduced the dependence on the number of queries to $\log k$, the error bound was still higher than the sampling error of $1/\sqrt{N}$. Furthermore, in general the algorithm is super-polynomial in both N and k.

Recently, [15] proposed a multiplicative weights update (MW) based algorithm that can guarantee $O(\sqrt{(\log k)/N})$ error for *linear queries* over normalized histograms. Their method maintains a differentially private artificial dataset at each step. For a given query, if the existing artificial dataset provides an answer close to the true response then the artificial dataset is not updated. Otherwise, the artificial dataset is updated so that it gets "closer" to the true dataset. [15] show that a multiplicative update to the dataset requires a small number of updates and hence only a small amount of noise needs to be added at each step.

Subsequently, [13] proposed a more generic framework which, given an update mechanism or Iterative Database Construction scheme (IDC) for maintaining artificial dataset, can guarantee privacy as well as *utility* (i.e., bound on the error in each query response). Utility guarantee by [13] depends on the number of updates that the given IDC might require in the worst case. Moreover, [13] also proposed an update scheme based on Frieze/Kannan (FK) cut decomposition method and provided utility guarantee for the same.

In this paper, we use the framework of [13] and provide a generic Iterative Database Construction scheme (IDC) based on the Mirror Descent algorithm, a popular convex optimization method [1]. Our Mirror Descent based IDC (MD-IDC) scheme can be adapted for any strongly convex potential function. Further, we provide a bound on the number of updates required by our MD-IDC scheme and thus obtain privacy and utility guarantees using framework of [13]. We show that the MW update based IDC (MW-IDC) and the FK algorithm based IDC (FK-IDC) are special cases of our generic MD-IDC and their utility guarantees follows directly from our generalized analysis.

Depending on the structure of the set of queries as well as the geometry of the dataset, MD-IDC can provide different utility guarantees for different potential functions. We provide examples where, by selecting different potential function than the ones used by [15,13], we can obtain tighter error bounds.

Next, we apply our framework to the problem of releasing cut values in a bipartite graph and propose an algorithm that guarantees smaller error than both [15] and [13]. For this problem, we use a *group*-norm based potential function that is known to exploit sparsity structure in the data [22]. Similarly, we apply

our framework to the problem of releasing linear queries over a dataset that is a low-rank matrix. We show that by using spectral structure of both the underlying matrix and the queries, our method guarantees smaller error than the methods of [13] and [15].

Our Contributions:

1. **Unify and generalize MW-IDC and FK-IDC [15,13]:** We propose a generic Mirror Descent based IDC (MD-IDC) which is a generalized update rule from which MW-IDC and FK-IDC can be derived as special cases.
2. **Exploit geometry of true dataset and queries:** Using our Mirror Descent-IDC, we can capture a wider class of structural properties on the underlying dataset $\boldsymbol{x}^*$ and the set of linear queries $\mathcal{F}$. Specifically, we provide potential functions for MD-IDC that can directly exploit bound on any arbitrary L_p-norm of $\boldsymbol{x}^*$ and the L_q-norm over $\mathcal{F}$. In contrast, [15,13] are limited to (L_1, L_∞) and (L_2, L_2) norm pair, respectively.
3. **Application to graph cuts release and linear query release over low-rank matrices:** We compare our utility bounds against the ones provided by MW-IDC and FK-IDC on three practically relevant applications: i) interactive cut query release for *imbalanced* bi-partite graphs (i.e., bi-partite graphs with large degree variations), ii) interactive cut query release for *power-law distributed* bipartite graphs (i.e., bipartite graphs where degree distribution follows a power-law), and iii) private matrix sensing where goal is to release responses to linear queries over a *low-rank* matrix. In each case, we show that our error bounds are significantly tighter than the ones obtained by the existing methods [15,13].

Paper Outline: In Section 3, we formulate our problem and discuss the framework of [13] in Section 3.1. Then, in Section 3.2, we propose our Mirror Descent based IDC algorithm and provide utility guarantees for the same. In Section 4, we provide two applications of our MD-IDC framework, namely, 1) releasing cut-queries in bi-partite graph, 2) releasing linear queries over low-rank matrices.

2 Notation and Preliminaries

Let $\boldsymbol{x}^* \in \mathbb{R}^d$ denote the private dataset, $\mathcal{F} = \{f_1, \ldots, f_k\}$ denote the function sequence provided to the online query response algorithm. For every query function $f \in \mathcal{F}$, we assume $f : \mathbb{R}^d \to \mathbb{R}$ to be a linear function, denoted by $f(\boldsymbol{x}) = \langle f, \boldsymbol{x} \rangle$. Vectors are denoted by bold-face symbols (e.g., $\boldsymbol{x}$), matrices are represented by capital letters (e.g., M). X_i denotes i-th row of X. $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ denotes the inner product between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$. Similarly, $\mathrm{Tr}(X^T Y) = \langle X, Y \rangle$ denotes the inner product between X and Y. L_p norm or p-norm of a vector $\boldsymbol{x} \in \mathbb{R}^d$ is denoted as $\|\boldsymbol{x}\|_p = \left(\sum_i^d x_i^p\right)^{1/p}$ and $p^* = \frac{p}{p-1}$ denotes the dual norm of L_p. For a matrix X, $\|X\|_p$ represents L_p-norm of vectorized X. $\|X\|_F = \sqrt{\sum_{ij} X_{ij}^2}$ denotes the Frobenius norm of X.

Definition 1 ((p,q)-group norm of matrix X). $\|X\|_{p,q} = (\sum_{i=1}^{m} \|X_i\|_p^q)^{1/q}$, *where* $X \in \mathbb{R}^{m \times n}$. *Hence,* (p,q)*-group norm is equivalent to* L_q *norm of a vector of* L_p *norms of rows of* X.

Definition 2. *Let* $X = U\Sigma V^T$ *be the singular value decomposition of* X. *Then Schatten p-norm of* X *is given by:* $\|X\|_{S_p} = \left(\sum_i \sigma_i^p\right)^{1/p}$, *where* σ_i *is the* i*-th singular value of* X *and* $\sigma_1 \geq \sigma_2 \ldots$.

Definition 3 (Uniform convexity). *A function* $\Psi : \mathbb{R}^d \to \mathbb{R}$ *is* s*-uniformly convex (for* $s \geq 1$*) with respect to* $\|\cdot\|_r$ *iff:* $\forall \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d, \forall \alpha \in [0,1], \Psi_r(\alpha \boldsymbol{x} + (1-\alpha)\boldsymbol{y}) \leq \alpha \Psi_r(\boldsymbol{x}) + (1-\alpha)\Psi_r(\boldsymbol{y}) - \frac{\alpha(1-\alpha)}{s}\|\boldsymbol{x} - \boldsymbol{y}\|_r^s$

Note that the definition above is a generalization of the conventional strong convexity definition where s is set to be two.

Definition 4. *Let* $\Psi : \mathbb{R}^d \to \mathbb{R}$ *be a continuously differentiable strictly convex potential function. Then, the Bregman's divergence (generated by* Ψ*) between any two vectors* $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$ *is defined as:*
$\Delta_\Psi(\boldsymbol{x}_1; \boldsymbol{x}_2) = \Psi(\boldsymbol{x}_1) - \Psi(\boldsymbol{x}_2) - \langle \bigtriangledown \Psi(\boldsymbol{x}_2), \boldsymbol{x}_1 - \boldsymbol{x}_2 \rangle$.

3 Problem Definition and Overview

Given a private dataset $\boldsymbol{x}^* \in \mathbb{R}^d$ and a set of queries $\mathcal{F} = \{f_1, \ldots, f_i \ldots f_k\}$, $f_i : \mathbb{R}^d \to \mathbb{R}, \forall i$, the goal is to answer each query f_i accurately (w.r.t $\boldsymbol{x}^*$) while preserving privacy of $\boldsymbol{x}^*$. That is, if a_i is the response to query f_i, then we want:

$$|a_i - f_i(\boldsymbol{x}^*)| \leq T, \forall 1 \leq i \leq k,$$

while preserving privacy of $\boldsymbol{x}^*$; $T > 0$ is an error parameter.

The above mentioned problem is known as the *interactive dataset release* problem [19,15]. In this setting, the queries can be adversarial, that is the adversary can select f_i depending on responses to previous queries. Hence, the privacy of each response a_i has to be argued w.r.t. complete query set $\mathcal{F}$.

For privacy, we use the notion of differential privacy which is now a benchmark notion [7,6]. Intuitively, an algorithm is differential private if addition (removal) of an entry to (from) the dataset does not significantly alter the output. In the context of *interactive dataset release*, it requires a guarantee that *none* of the query response a_i change significantly, if one entry of the dataset $\boldsymbol{x}^*$ is modified. Below, we provide a formal definition of $(\epsilon, \delta, \gamma)$-differential privacy adapted for the problem of interactive dataset release.

Definition 5 (Differential privacy [7,6]). *An algorithm* $\mathcal{A}$ *is* $(\epsilon, \delta, \gamma)$*-differentially private if for any two datasets* $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$ *s.t.* $\|\boldsymbol{x} - \boldsymbol{x}'\|_1 \leq \gamma$, *and for all measurable sets* $\mathcal{O} \subseteq Range(\mathcal{A})$, *the following holds:*
$\Pr[\mathcal{A}(\boldsymbol{x}) \in \mathcal{O}] \leq e^\epsilon \Pr[\mathcal{A}(\boldsymbol{x}') \in \mathcal{O}] + \delta$.

Algorithm 1. Online Query Response Mechanism (OQR) [15,13]

Require: Dataset: $\boldsymbol{x}^*$, privacy parameters: $(\epsilon, \delta, \gamma)$, query set $\mathcal{F} = \{f_1, \cdots, f_k\}$, failure probability β,
U_{IDC}: IDC algorithm, B: bound on number of updates by U_{IDC}

Set noise parameter: $\epsilon_0 \leftarrow \frac{\epsilon}{100\gamma\sqrt{B\log(4/\delta)}}$, Set threshold $T \leftarrow \frac{4}{\epsilon_0}\log(2k/\beta)$
$\boldsymbol{x}_0 = U_{IDC}(\text{NULL}, \text{NULL}, \text{NULL})$, counter $= 0$.
for $t \in \{1, \cdots, k\}$ and counter $< B$ **do**
  $A_t \sim \text{Lap}(\frac{1}{\epsilon_0})$
  True response: $a_t = f_t(\boldsymbol{x}^*)$, Noisy response: $\hat{a}_t \leftarrow f_t(\boldsymbol{x}^*) + A_t$, Noisy difference: $\hat{d}_t \leftarrow \hat{a}_t - f_t(\boldsymbol{x}_{t-1})$
  if $|\hat{d}_t| > T$ **then**
    $\boldsymbol{x}_t \leftarrow U_{IDC}(\boldsymbol{x}_{t-1}, f_t, \hat{d}_t)$, counter $\leftarrow$ counter $+ 1$
    Output query response: $\hat{a}_t = f_t(\boldsymbol{x}^*) + A_t$
  else
    No update, i.e., $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1}$
    Output query response: $\hat{a}_t = f_t(\boldsymbol{x}_t)$
  end if
end for

Now, a special case of the above mentioned problem is when each query f_i is linear, i.e., $f_i(\boldsymbol{x}) = \langle f_i, \boldsymbol{x}\rangle$, $f_i \in \mathbb{R}^d$. Most of the existing results are for the case of linear queries only. For rest of the paper, we assume f_i to be a linear query; we discuss extension to the nonlinear case in the full version of this paper [16].

Recently, [15] provided a multiplicative weights update based differentially private algorithm for the problem of *interactive dataset release* (with linear queries) that guarantees at most $O(\log^{1/4} k \log d)$ error in each query. Subsequently, [13] proposed a more general framework that uses *Iterative Database Construction (IDC)* algorithms to provide differentially private versions of dataset $\boldsymbol{x}^*$. [13] provided a tighter analysis of the multiplicative weights based algorithm (MW-IDC) of [15]. They also proposed a novel IDC algorithm based on Frieze/Kannan cut-decomposition algorithm (FK-IDC) [10] and apply their method to the problem of releasing graph cuts.

In the next section, we introduce the above mentioned *online query release mechanism* of [13] and state the generic utility and privacy guarantee of [13]. Then, in section 3.2, we present our generic Mirror Descent based IDC (MD-IDC) and show that both MW-IDC and FK-IDC form special cases of our MD-IDC algorithm. Further, their error bounds follow directly from our generic analysis for MD-IDC. We also provide two applications where different instantiations of our MD-IDC provide better error bounds than MW-IDC and FK-IDC.

3.1 Online Query Release Mechanism

[15,13] introduced a generic online query release mechanism where at each step t, a differentially private (or "public") version of the dataset $\boldsymbol{x}_{t-1}$ is maintained. Now, for a given query f_t (that can be adversarially chosen according to $\boldsymbol{x}_{t-1}$ and past query responses), the algorithm tries to answer the query using $\boldsymbol{x}_{t-1}$.

However, if query response $f_t(\boldsymbol{x}_{t-1})$ is "too far" from the true response $f_t(\boldsymbol{x}^*)$, then the algorithm answers the query based on the true dataset $\boldsymbol{x}^*$. Also, as the dataset $\boldsymbol{x}_{t-1}$ is "inaccurate", hence it is *updated* so that it gets closer to $\boldsymbol{x}^*$. The *update* algorithm is called *Iterative Database Construction (IDC)* algorithm, and should produce next iterate $\boldsymbol{x}_t$ using the previous iterate $\boldsymbol{x}_{t-1}$, current query f_t, and the response provided for f_t. That is, $U_{IDC} : \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R} \to \mathbb{R}^d$, where U_{IDC} is the given IDC algorithm. See Algorithm 1 for a pseudo-code.

Now, [15] observed that, for iterations where iterate $\boldsymbol{x}_{t-1}$ is not updated, Algorithm 1 is $(\epsilon = 0)$-differentially private with high probability over the randomness of the algorithm. Also, $\boldsymbol{x}_{t-1}$ is updated only for a small number of steps. Using these observations, [15,13] show that the noise parameter set in Step 1 of Algorithm 1 is enough to guarantee privacy of $\boldsymbol{x}^*$.

Theorem 1 (Privacy (Theorem 4.1, [13])). *Assuming each query $f \in \mathcal{F}$, $\|f\|_\infty \leq 1$, Algorithm 1 is $(\epsilon, \delta, \gamma)$-differentially private.*

Similar to [15,13], utility (i.e., maximum error in any query response) can be guaranteed easily by bounding the magnitude of the noise added using tail bounds for Laplace distribution.

Theorem 2 (Utility). *If the variable counter (defined in Step 2 of Algorithm 1 (Algorithm* OQR *)) is less than B after all the k-query responses, then with probability $\geq 1 - \frac{\beta}{2}$, Algorithm* OQR *incurs at most $2T$ error in each query response, i.e.,*

$$|\hat{a}_t - f_t(\boldsymbol{x}^*)| \leq 2T = \frac{800\gamma\sqrt{B}\log(4/\delta)\log(2k/\beta)}{\epsilon}, \forall 1 \leq t \leq k,$$

where B is the bound on number of updates using U_{IDC}.

Note that privacy guarantee of Algorithm 1 is independent of the IDC algorithm (U_{IDC}), while the utility guarantee depends on U_{IDC} only through a bound on the number of updates (B). Hence, the most critical aspect of Algorithm 1 is the design of U_{IDC} and provide a tight upper bound on B for the given application. In next section, we present a generic Mirror Descent algorithm based IDC algorithm that can be adapted according to the underlying application to obtain better bound on B (and hence the utility guarantee).

3.2 Mirror Descent Based IDC

In this section, we introduce our Mirror Descent based IDC. Mirror descent is a popular optimization algorithm [1], that is also extensively used in the context of online learning [20]. Suppose, the goal is to minimize a function $\ell(\boldsymbol{x})$ s.t. $\boldsymbol{x} \in \mathcal{C}$ where $\mathcal{C}$ is a convex set. Then, mirror descent uses the following *exploration-exploitation* based update (with $\Delta(\cdot;\cdot)$ being the distance function):

$$\boldsymbol{x}_t = \arg\min_{\boldsymbol{x} \in \mathcal{C}} \left(\Delta(\boldsymbol{x}; \boldsymbol{x}_{t-1}) - \eta_t \langle \bigtriangledown \ell(\boldsymbol{x}_{t-1}), \boldsymbol{x} \rangle\right). \tag{1}$$

For online query release mechanism (Algorithm OQR (Algorithm 1)), we use a similar MD-based update to design IDC. Specifically, we set $\ell_t(\boldsymbol{x}) = |f_t(\boldsymbol{x}^*) -$

Algorithm 2. Mirror Descent based IDC (MD-IDC)

Require: Previous iterate: $\boldsymbol{x}_{t-1}$, Linear query: $f_t \in \mathcal{F}$, Norm parameters: p, q, Noisy difference in response: $\hat{d}_t = \langle f_t, \boldsymbol{x}^* \rangle + A_t - \langle f_t, \boldsymbol{x}_{t-1} \rangle$, Threshold: T, Privacy parameters: $(\epsilon, \delta, \gamma)$, $\zeta_q = \max_{f \in \mathcal{F}} \|f\|_q$, Potential function: Ψ that is s-uniformly convex w.r.t. $\|\cdot\|_r$, $r = \frac{q}{q-1}$

1: Define $\mathcal{C} = \{\boldsymbol{x} \ s.t. \ \|\boldsymbol{x}\|_p \leq \|\boldsymbol{x}^*\|_p\}$

2: Set step size $\eta = \frac{(s-1)^{s-1}(T/2)^{s-1}}{s^s \zeta_q^s}$ and update bound $B = \frac{2^{s-1} s^s \zeta_q^s}{T^s (s-1)^{s-1}} \max_{\boldsymbol{x} \in \mathcal{C}} \Psi(\boldsymbol{x})$

3: **if** $\boldsymbol{x}_{t-1} = \phi$ (i.e., $t = 1$) **then**

4: Output: $\boldsymbol{x}_0 = \operatorname{argmin}_{\boldsymbol{x} \in \mathcal{C}} \Psi(\boldsymbol{x})$

5: **else**

6: Output: $\boldsymbol{x}_t \leftarrow \operatorname{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left(\Delta_\Psi(\boldsymbol{x}; \boldsymbol{x}_{t-1}) - \eta \cdot \operatorname{sgn}(\hat{d}_t) \langle \bigtriangledown f_t(\boldsymbol{x}_{t-1}), \boldsymbol{x} - \boldsymbol{x}_{t-1} \rangle \right)$

7: **end if**

$f_t(\boldsymbol{x})|$. Note that, we want to update $\boldsymbol{x}_{t-1}$ so that $\ell_t(\boldsymbol{x})$ is small, i.e., $\boldsymbol{x}_t$ does not make mistakes on queries similar to f_t. But at the same time, we want $\boldsymbol{x}_t$ to be close to $\boldsymbol{x}_{t-1}$, as it contains information learned from previous queries.

As $\ell_t(\boldsymbol{x}) = |f_t(\boldsymbol{x}^*) - f_t(\boldsymbol{x})|$ is not a differentiable function, we use the following sub-gradient of ℓ_t: $\partial \ell_t(\boldsymbol{x}_{t-1}) = -\operatorname{sgn}(f_t(\boldsymbol{x}^*) - f_t(\boldsymbol{x}_{t-1})) \bigtriangledown f_t(\boldsymbol{x}_{t-1})$. Also, as each function f_t is linear, i.e., $f_t(\boldsymbol{x}) = \langle f_t, \boldsymbol{x} \rangle$, $f_t \in \mathbb{R}^d$: $\bigtriangledown f_t(\boldsymbol{x}) = f_t$.

Finally, we use Bregman's divergence as the distance function $\Delta(\cdot;\cdot)$. Given a continuously differentiable strictly convex function Ψ, the corresponding Bregman's divergence is given by: $\Delta_\Psi(\boldsymbol{x}_1; \boldsymbol{x}_2) = \Psi(\boldsymbol{x}_1) - \Psi(\boldsymbol{x}_2) - \langle \bigtriangledown \Psi(\boldsymbol{x}_2), \boldsymbol{x}_1 - \boldsymbol{x}_2 \rangle$.

Hence, for a given potential function Ψ and $d_t = f_t(\boldsymbol{x}^*) - f_t(\boldsymbol{x}_{t-1})$, our MD-IDC update for linear queries is given by:

$$\boldsymbol{x}_t = \operatorname*{argmin}_{\boldsymbol{x} \in \mathcal{C}} \left(\Delta_\Psi(\boldsymbol{x}; \boldsymbol{x}_{t-1}) - \eta \cdot \operatorname{sgn}(d_t) \langle f_t, \boldsymbol{x} - \boldsymbol{x}_{t-1} \rangle \right),$$

where η is selected appropriately. See Algorithm 2 for a pseudo-code of our MD-IDC algorithm. In the following, we provide the utility guarantees for our MD-IDC based Online Query Response Mechanism (Algorithm 1).

Theorem 3 (Utility). *Let $f_t \in \mathcal{F}, 1 \leq t \leq k$ be a linear query, let q be the norm chosen for the query set $\mathcal{F}$ and let $\mathcal{C} = \{\boldsymbol{x} \ s.t. \ \|\boldsymbol{x}\|_p \leq \|\boldsymbol{x}^*\|_p\}$. Furthermore, let $\Psi(\cdot)$ be a s-strongly convex function w.r.t. $\|\cdot\|_r$, where $r = \frac{q}{q-1}$. Then, w.p. at least $1 - \beta$, for each query response, the error incurred by MD-IDC (Algorithm 2) based* OQR *algorithm (Algorithm 1) is bounded by:*

$$|\hat{a}_t - f_t(\boldsymbol{x}^*)| = O\left(\frac{\log(k/\beta)^{2/(s+2)} (\gamma \zeta_q)^{s/(s+2)} \log^2(1/\delta)}{\epsilon^{s/(s+2)}} \left(\max_{\boldsymbol{x} \in \mathcal{C}} \Psi(\boldsymbol{x}) \right)^{1/(s+2)} \right),$$

where, $1 \leq t \leq k$, $\zeta_q \leq \max_{f \in \mathcal{F}} \|f\|_q$ and $(\epsilon, \delta, \gamma)$ are the privacy parameters.

A detailed proof of the above theorem is provided in the full version [16].

Special Cases: MW-IDC & FK-IDC: Above we described our generic MD-IDC algorithm which given any *potential function*, provides bound on the error in

each query's response. Our algorithm has the flexibility of selecting the potential function for different problem settings.Recall that the potential function should be strongly convex w.r.t. $\|\cdot\|_{\frac{q}{q-1}}$-norm over set $\mathcal{C} = \{\boldsymbol{x}\ s.t.\ \|\boldsymbol{x}\|_p \leq \|\boldsymbol{x}^*\|_p\}$, while $\max_{\boldsymbol{x}\in\mathcal{C}} \Psi(\boldsymbol{x})$ should be small. Note that, here we assume that $\|\boldsymbol{x}^*\|_p$ is is known *publicly* or an approximate version of the same can released in differentially private manner, by adding appropriate amount of noise.

Now, it is known that for $1 < p \leq 2$, $\Psi_p(\boldsymbol{x}) = \frac{1}{p-1}\|\boldsymbol{x}\|_p^2$ is 2-uniformly convex w.r.t. $\|\cdot\|_p$. Selecting $p = q^*$ (i.e., p, q are *dual* pairs) and ignoring privacy parameters (ϵ, δ) and failure probability β, we get the following error bound: $\mathsf{Err}_p = O(\sqrt{\gamma}\|f\|_{p^*}\|\boldsymbol{x}^*\|_p \log k)$. Now, if $\boldsymbol{x}^*$ is a histogram over a database with N records, then $\gamma = \frac{1}{N}$. Hence, $\mathsf{Err}_p = O(\sqrt{\frac{1}{N}\|f\|_{p^*}\|\boldsymbol{x}^*\|_p \log k})$.

Interestingly, selecting $p = 2$, our MD-IDC reduces to Frieze/Kannan IDC (FK-IDC) of [13]. Further, the error bound is also *exactly* the same as the one obtained by [13]. Similarly, selecting $p = \frac{\log d}{\log d - 1}$, we get the matching error bound for MW-IDC [13]. However, the algorithm is different than that of MW-IDC and is in fact more general, as it can be applied to any real-valued $\boldsymbol{x}^*$, while MW-IDC applies to positive vectors only. Further, selecting $\Psi_H(\boldsymbol{x}) = \sum_i x_i \log x_i$, we obtain exact MW-IDC algorithm. Note that, $\Psi_H(\boldsymbol{x})$ is 2-uniformly convex w.r.t. L_1 norm and hence can be applied directly in our framework.

Above, we assume p and q to be dual pairs, i.e., $q = p^*$. However, similar to [20], selecting non-dual (p, q) pair can lead to tighter bounds for certain settings. We defer the details for non-dual (p, q) pairs to the full version of the paper [16].

4 Applications

In this section, we discuss some of the applications of our MD-IDC, and show that by selecting an appropriate potential function Ψ for a given application, we can obtain significantly more accurate answers than [13,15]. In particular, we provide two concrete applications and show that we can devise problem specific potential functions to outperform the existing methods of [13,15].

4.1 Online Cut-Query Release

In this section, we consider the problem of releasing cut-queries over a private *bi-partite* graph. Specifically, let $G = (V_1, V_2, E)$ be an undirected bi-partite graph and let $S \subseteq V_2$ be a subset of nodes. The goal here is to release cut $(S, \bar{S})$ while preserving privacy. The cut query answers the following question: how "well-connected" are the nodes of V_1 are to $S \subseteq V_2$. For simplicity of exposition we assume $S \subseteq V_2$; for $S \subseteq V_2 \cup V_1$, similar results can be obtained easily.

For online cut-query release, the "dataset" is given by the adjacency matrix of G, i.e., $X^* \in \mathbb{R}^{|V_1|\times|V_2|}$. $X^*_{ij} = 1, \forall (i,j) \in E, 1 \leq i \leq |V_1|, 1 \leq j \leq |V_2|$ and is zero otherwise. Similarly, a cut query is given by $F \in \mathbb{R}^{|V_1|\times|V_2|}$, where $F_{ij} = 1, \forall i \in S, j \in \bar{S}$. Hence, the cut size is given by $C(S, G) = \langle X^*, F\rangle$.

Note that, we want to guarantee privacy for each edge in the graph. Hence, removing or adding an edge from X leads to an "adjacent" dataset X'. Also,

$\gamma = \|X - X'\|_1 \leq 1$. We seek an algorithm that answers queries accurately while providing $(\epsilon, \delta, \gamma = 1)$-differential privacy. For this problem, we use Algorithm 1 (Algorithm OQR) with our generic MD-IDC.

For MW-IDC [15], using Theorem 3 and $k = O(|V_2|^{|S|})$, (ignoring privacy parameters (ϵ, δ) and failure probability β) the error in each query is given by:

$$\mathsf{Err}_{MW} = O\left(\sqrt{\zeta_\infty |E||S| \log(|V_1||V_2|)}\right), \quad \zeta_\infty = \max_t \|F_t\|_\infty = 1. \tag{2}$$

Now, for FK-IDC [13], Theorem 3 provides the following bound:

$$\mathsf{Err}_{FK} = O\left(\sqrt{\zeta_2 |E|^{1/2} |S| \log(|V_2|)}\right), \quad \zeta_2 = \max_t \|F_t\|_2. \tag{3}$$

Similar to the previous section, we can select a different L_p-norm potential function for our MD-IDC, than the one used by MW-IDC, FK-IDC. However, for this problem, that does not lead to an improvement over MW-IDC and FK-IDC. Instead, with the intent of exploiting the structure of the adjacency matrix, we select group-norm based potential functions (see Definition 1). Of particular interest is the $(2, p)$-norm, where $p \approx 1$. Similar to L_p norms, it can be shown that $\Psi_{2,p}(X) = \frac{1}{p-1}\|\boldsymbol{x}\|_{2,p}^2$ is 2-uniformly convex w.r.t. $\|\cdot\|_{2,p}, 1 < p \leq 2$. Note that, this function is same as the "Group Lasso" regularizer [22] and is known to be useful for recovering vectors with shared sparsity. For our problem, this function is useful for the case where degrees of nodes in the graph have heavy variation.

Using Theorem 3, error incurred by MD-IDC with $(2, p)$-norm function is:

$$\mathsf{Err}_{MD-IDC} = O\left(\sqrt{\zeta_{2,p^*} \|X^*\|_{2,p} |S| \log(|V_2|)}\right), \tag{4}$$

where $\zeta_{2,p^*} = \max_t \|F_t\|_{2,p^*}$ and $p^* = p/(p-1)$. Note that, the error bound for our group-norm based MD-IDC is in general incomparable to the corresponding bounds by MW-IDC or FK-IDC. However for several specific problems, group-norm based MD-IDC outperforms both MW-IDC and FK-IDC. Below, we provide two such examples.

Imbalanced Bi-partite Graph: Consider a bi-partite graph where the node sets V_1 and V_2 are of equal cardinality, i.e., $|V_1| = |V_2| = V$. Let V_1 be divided into two sets $V_1 = \{A, B\}$. Let $|A| = |V|^{3/4}$ and let each node of A be connected to every node of V_2, while each node of B is connected to only $|V|^{1/2}$ nodes of B. That is a small number of nodes are highly connected, while the remaining nodes are sparsely connected. Recall that the cut-queries are over a set $S \subseteq V_2$.

Note, that for the above mentioned family of graph $|E| = O(|V|^{7/4})$. Hence, bounds for MW-IDC and FK-IDC are given by:

$$\mathsf{Err}_{MW} = \tilde{O}(|V|^{7/8}|S|^{1/2}), \qquad \mathsf{Err}_{FK} = \tilde{O}(|V|^{11/16}|S|^{3/4}) \tag{5}$$

Similarly, the error incurred by $(2, p = \frac{\log|V|}{\log|V|-1})$-norm based MD-IDC is:

$$\mathsf{Err}_{2,\frac{\log|V|}{\log|V|-1}} = \tilde{O}(|V|^{5/8}|S|^{3/4}).$$

Hence, if $|S| = o(|V|)$, then:

$$\mathsf{Err}_{2,\frac{\log|V|}{\log|V|-1}} = o(1)\mathsf{Err}_{MW}, \quad \mathsf{Err}_{2,\frac{\log|V|}{\log|V|-1}} = o(1)\mathsf{Err}_{FK}.$$

Also, note that the error incurred by a trivial response of 0 for each query is bounded by: $|V||S|$. Similarly, standard randomized response leads to $O(|V|^{3/2})$ error. Hence, our error guarantees are better than the trivial baselines as well.

Power-Law Distributed Bi-partite Graph: Next, we consider a more practical scenario where degrees of nodes in V_1 follow a power-law distribution. Several graphs that arise in practice have been shown to follow a power-law distribution. For simplicity, we assume $|V_1| = |V_2| = |V|$. Now, power-law distribution assumption implies: $\mathbb{E}[\text{Number of nodes with degree } i] = \frac{i^{-\beta}}{\sum_{i=j}^{|V|} j^{-\beta}}|V|$, where $\beta > 0$ is a parameter of the distribution. For simplicity, we drop expectation from the above statement and assume the following *deterministic* statement:

$$\text{Number of nodes with degree } i = \frac{i^{-\beta}}{\sum_{i=j}^{|V|} j^{-\beta}}|V|.$$

If $1 < \beta < 2$, it can be shown that: $|E| = O(|V|^{3-\beta})$. Hence, using (2), (3):

$$\mathsf{Err}_{MW} = \tilde{O}(|V|^{3/2-\beta/2}|S|^{1/2}), \quad \mathsf{Err}_{FK} = \tilde{O}(|V|^{1-\beta/4}|S|^{3/4}).$$

Similarly, using (4), for $1 < \beta < 3/2$: $\mathsf{Err}_{MD-IDC} = \tilde{O}(|V|^{3/4-\beta/2}|S|^{3/4})$. Hence, using the fact that $|S| \leq |V|$ and assuming $1 < \beta < 3/2$:

$$\mathsf{Err}_{2,\frac{\log|V|}{\log|V|-1}} = o(1)\mathsf{Err}_{MW}, \quad \mathsf{Err}_{2,\frac{\log|V|}{\log|V|-1}} = o(1)\mathsf{Err}_{FK}.$$

Finally, we compare the above mentioned error bounds with the error incurred by a trivial response of 0. For this trivial response, the error is bounded by: $\min\{|S||V|, |E|\} = \min\{|S||V|, |V|^{3-\beta}\}$. Hence, if $|S| \geq |V|^{1-\frac{2}{3}\beta}$, then $\mathsf{Err}_{2,\frac{\log V}{\log V-1}}$ is smaller than the error incurred by the trivial response. Similarly, randomized response incurs $O(|V|^{3/2})$ error. Hence, if $|S| = o(|V|)$, then our proposed MD-IDC obtains better error bounds.

Finally, we note that while our results are for online cut-queries, they can also be used for releasing *sanitized* differentially private graphs which are accurate for cut queries. However, our algorithm would require to process all $O(|V|^{|S|})$ cut-queries. We leave further investigation of our MD-IDC method for release of sanitized differentially-private graphs as future work.

4.2 Online Query Release over Low-Rank Matrix

In this section, we consider the problem of releasing response to linear queries where the dataset is a low-rank matrix. Let $X^* \in \mathbb{R}^{m\times n}$ be a rank-r matrix and let $F_t \in \mathbb{R}^{m\times n}$ be a linear query. Then, the response to the query is: $\langle F_t, X^*\rangle$.

A Practical Scenario: let X^* be a user-movie rating matrix, i.e., $X^*_{i,j}$ is the rating user i provides for movie j. And the queries answer questions of the form: "what is the average rating for comedy movies for users from Seattle".

We can directly apply Algorithm OQR (Algorithm 1) to release response to these queries, while providing privacy guarantees for each individual entry in X^*. Assuming $\|X^*\|_\infty = 1$, for any adjacent dataset X', $\|X^* - X'\|_1 \leq 1 = \gamma$. Recall that, $\|X\|_p$ represents L_p norm of vectorized X.

Similar to the previous section, we provide a potential function for our MD-IDC that provides better error guarantees than MW-IDC and FK-IDC. Note that, the matrix X^* can have negative entries as well, hence multiplicative weight based algorithm from [15] cannot be applied directly.

Using Theorem 3 with FK-IDC, we obtain the following error bound for answer k-queries (ignoring privacy parameters (ϵ, δ) and failure probability β):

$$\mathsf{Err}_{FK} = \tilde{O}\left(\sqrt{\zeta_2 \|X^*\|_F \log k}\right), \text{ where } \zeta_2 \leq \max_t \|F_t\|_F.$$

In the previous section, we used group-norm based potential functions as they are more well-suited for exploiting degree structure of the graphs. In this section, we use another popular class of potential functions based on the Schatten-p norm (see Definition 2) that is more well-suited to exploit the spectral structure of X^*.

Similar to L_p norm, it is known that $\Psi_{S_p}(X) = \frac{1}{p-1}\|X\|_{S_p}^2, \forall 1 < p \leq 2$, is 2-strongly convex w.r.t. $\|\cdot\|_{S_p}$ [17]. Hence using Theorem 3, the error incurred by Algorithm 1 with MD-IDC and with potential function Ψ_{S_p} is bounded by:

$$\mathsf{Err}_{S_p} = \tilde{O}\left(\sqrt{\zeta_{S_p^*} \|X^*\|_{S_p} \log(k)}\right), \tag{6}$$

where $p^* = \frac{p}{p-1}$ and $\zeta_{S_p^*} = \max_t \|F_t\|_{S_{p^*}}$. Note that, for $p = 2$, S_2 is the Frobenius norm and hence in that case, the above bound is same as Err_{FK}.

Now, for the case of low-rank matrices, Schatten-1 norm (or "trace" norm) is a popular regularization as it generally preserves the low-rank structure. Below, we show for a large class of queries using trace norm based MD-IDC indeed achieves better error bounds than both MW-IDC and FK-IDC. Specifically, let $p = \frac{\log mn}{\log mn - 1} \approx 1$. Then, using (6) and $\|X^*\|_{S_1} \leq \sqrt{r}\|X^*\|_F$ we get:

$$\mathsf{Err}_{S_1} = \tilde{O}\left(\sqrt{\sqrt{r}\zeta_{S_p^*} \|X^*\|_F \log(k)}\right), \tag{7}$$

where $p^* = \log(mn)$. Hence, if $\sqrt{r}\|F_t\|_{S_{\log mn}} < \|F_t\|_F, \forall t$, then $\mathsf{Err}_{S_1} < \mathsf{Err}_{FK}$. Now, $\|F_t\|_{S_{\log mn}} \leq e\sigma_1^{F_t}$, where $\sigma_1^{F_t}$ is the largest singular value of F_t. Similarly, $\|F_t\|_F = \sqrt{\sum_i (\sigma_i^{F_t})^2}$, where $\sigma_i^{F_t}$ is the i-th singular value of F_t.

Now, if each query F_t is a rank-1 query, then, $\sqrt{r}\|F_t\|_{S_{\log mn}} > \|F_t\|_F$ for $r > 1$. Hence, in this case, Frobenius-norm based potential function leads to tighter bounds. However, if the queries have "spread-out" spectrum, then trace-norm based potential function is more accurate.

A concrete example of such a case is when each element of F_t is sampled uniformly from a standard Gaussian, i.e., $F_t(i,j) \sim N(0,1)$. In this case, using Corollary 5.35 of [21] and assuming $m > 4n$, we get (w.h.p.): $\sqrt{n} \leq \sigma_n^{F_t} \leq \sigma_1^{F_t} \leq 3\sqrt{n}$. Hence, $r(\sigma_1^{F_t})^2 \leq 9rn \leq 9r\frac{1}{n}(\sigma_n^{F_t})^2$. That is, $\sqrt{r}\|F_t\|_{S_{\log mn}} \leq 3e\sqrt{r/n}\|F_t\|_F$. Hence, for $r = o(n)$, $\mathsf{Err}_{S_1} = o(1)\mathsf{Err}_{FK}$. In typical applications, r is a constant. Hence, Err_{S_1} is a factor of $\sqrt{n}$ *smaller than* Err_{FK}.

Note that, random queries F_t are used extensively in the domain of compressed sensing [3] and can be used to recover low-rank matrix X^* accurately. Hence, our result provides a method to recover matrix X^* approximately (with bounded error) without compromising accuracy of any single entry.

References

1. Beck, A., Teboulle, M.: Mirror descent and nonlinear projected subgradient methods for convex optimization. Oper. Res. Lett. 31(3), 167–175 (2003)
2. Blum, A., Ligett, K., Roth, A.: A learning theory approach to non-interactive database privacy. In: STOC (2008)
3. Candes, E.J., Tao, T.: Near-optimal signal recovery from random projections: universal encoding strategies? IEEE Transactions on Information Theory (2006)
4. Dinur, I., Dwork, C., Nissim, K.: Revealing information while preserving privacy, full version of [5] (2010) (in preparation)
5. Dinur, I., Nissim, K.: Revealing information while preserving privacy. In: PODS, pp. 202–210. ACM (2003)
6. Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., Naor, M.: Our Data, Ourselves: Privacy Via Distributed Noise Generation. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 486–503. Springer, Heidelberg (2006)
7. Dwork, C., McSherry, F., Nissim, K., Smith, A.: Calibrating Noise to Sensitivity in Private Data Analysis. In: Halevi, S., Rabin, T. (eds.) TCC 2006. LNCS, vol. 3876, pp. 265–284. Springer, Heidelberg (2006)
8. Dwork, C., McSherry, F., Talwar, K.: The price of privacy and the limits of LP decoding. In: STOC, pp. 85–94. ACM (2007)
9. Dwork, C., Rothblum, G.N., Vadhan, S.P.: Boosting and differential privacy. In: FOCS (2010)
10. Frieze, A.M., Kannan, R.: A simple algorithm for constructing szemere'di's regularity partition. Electr. J. Comb. (1999)
11. Ganta, S.R., Kasiviswanathan, S.P., Smith, A.: Composition attacks and auxiliary information in data privacy. In: KDD 2008: Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 265–273. ACM (2008)
12. Gupta, A., Hardt, M., Roth, A., Ullman, J.: Privately releasing conjunctions and the statistical query barrier. In: STOC (2011)
13. Gupta, A., Roth, A., Ullman, J.: Iterative constructions and private data release. CoRR, abs/1107.3731 (2011)
14. Hardt, M., Ligett, K., McSherry, F.: A simple and practical algorithm for differentially private data release. CoRR, abs/1012.4763 (2010)
15. Hardt, M., Rothblum, G.N.: A multiplicative weights mechanism for privacy-preserving data analysis. In: FOCS (2010)
16. Jain, P., Thakurta, A.: Mirror descent based database privacy. Technical Report NAS-TR-0159-2012, Pennsylvania State University (April 2012)
17. Kakade, S.M., Shalev-Shwartz, S., Tewari, A.: On the duality of strong convexity and strong smoothness: Learning applications and matrix regularization. Informal Publication (2009)
18. Kasiviswanathan, S.P., Lee, H.K., Nissim, K., Raskhodnikova, S., Smith, A.: What can we learn privately? In: FOCS (2008)
19. Roth, A., Roughgarden, T.: Interactive privacy via the median mechanism. In: STOC (2010)
20. Srebro, N., Sridharan, K., Tewari, A.: On the universality of online mirror descent. CoRR, abs/1107.4080 (2011)
21. Vershynin, R.: Introduction to the non-asymptotic analysis of random matrices. CoRR, abs/1011.3027 (2010)
22. Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. Journal of the Royal Statistical Society, Series B 68, 49–67 (2007)

Analysis of k-Means++ for Separable Data

Ragesh Jaiswal and Nitin Garg

Department of Computer Science and Engineering,
IIT Delhi, New Delhi, India
{cs5070222,rjaiswal}@cse.iitd.ac.in

Abstract. k-means++ [5] seeding procedure is a simple sampling based algorithm that is used to quickly find k centers which may then be used to start the Lloyd's method. There has been some progress recently on understanding this sampling algorithm. Ostrovsky et al. [10] showed that if the data satisfies the separation condition that $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq c$ ($\Delta_i(P)$ is the optimal cost w.r.t. i centers, $c > 1$ is a constant, and P is the point set), then the sampling algorithm gives an $O(1)$-approximation for the k-means problem with probability that is exponentially small in k. Here, the distance measure is the squared Euclidean distance. Ackermann and Blömer [2] showed the same result when the distance measure is any μ-similar Bregman divergence. Arthur and Vassilvitskii [5] showed that the k-means++ seeding gives an $O(\log k)$ approximation in expectation for the k-means problem. They also give an instance where k-means++ seeding gives $\Omega(\log k)$ approximation in expectation. However, it was unresolved whether the seeding procedure gives an $O(1)$ approximation with probability $\Omega\left(\frac{1}{poly(k)}\right)$, even when the data satisfies the above-mentioned separation condition. Brunsch and Röglin [8] addressed this question and gave an instances on which k-means++ achieves an approximation ratio of $(2/3-\epsilon)\cdot\log k$ only with exponentially small probability. However, the instances that they give satisfy $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} = 1 + o(1)$. In this work, we show that the sampling algorithm gives an $O(1)$ approximation with probability $\Omega\left(\frac{1}{k}\right)$ for any k-means problem instance where the point set satisfy separation condition $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq 1+\gamma$, for some fixed constant γ. Our results hold for any distance measure that is a metric in an approximate sense. For point sets that do not satisfy the above separation condition, we show $O(1)$ approximation with probability $\Omega(2^{-2k})$.

1 Introduction

The k-median problem with respect to a point domain $\mathcal{X}$ and a distance measure $D : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, is defined as follows:

> Given a set $P \subseteq \mathcal{X}$ of n points, find a subset $C \subseteq \mathcal{X}$ of k points (these are called *centers*) such that the objective function

$$\phi_C(P) = \sum_{p \in P} \min_{c \in C} D(p, c)$$

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 591–602, 2012.
© Springer-Verlag Berlin Heidelberg 2012

is minimized. For $\mathcal{X} = \mathbb{R}^d$ and $D(x, y) = ||x - y||^2$, the problem is called the k-means problem.

k-means++ seeding is a simple sampling algorithm that is used to quickly find k centers that is then used to start the Lloyd's method. This sampling procedure is extremely simple and can be described as follows:

(**SampAlg**) Pick the first center uniformly at random from P. Choose a point $p \in P$ to be the i^{th} center for $i > 1$ with probability proportional to the distance of p from the nearest previously chosen centers, i.e., with probability $\frac{\min_{c \in C} D(p,c)}{\phi_C(P)}$.

There has been some recent progress in understanding the above sampling procedure. However, even this simple procedure is not fully understood. There are a number of important questions that are unresolved. Next, we give the current state of understanding and discuss some of the unresolved questions.

Previous work. The non-uniform sampling technique defined above was first analysed by Ostrovsky et al. [10] for the k-means problem. They showed that if the given data is separable in the sense that $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq c > 1$, for some fixed constant c, then the sampling algorithm gives an $O(1)$ approximation with probability exponentially small in k. After this, Arthur and Vassilvitskii [5] showed that the algorithm gives an $O(\log k)$ approximation in expectation for *any* data set. They also give a problem instance where the algorithm gives an approximation of $\Omega(\log k)$ in expectation. However, for the instance that they construct, the sampling algorithm gives an $O(1)$ approximation with constant probability. The sampling algorithm may be regarded as useful as long as we can show that it gives an $O(1)$ approximation with probability $\Omega\left(\frac{1}{poly(k)}\right)$. This is because we may repeat $O(poly(k))$ times and take the best answer. Some initial progress towards this question was by Aggarwal [3] et al. and Ailon et al. [4] who showed that sampling more than k centers gives an $O(1)$ *pseudo*-approximation with constant probability. However, the basic question whether we can get an $O(1)$ approximation with probability $\Omega\left(\frac{1}{poly(k)}\right)$ remained unresolved. In a recent paper, Brunsch and Röglin [8] gave a problem instance where the sampling algorithm gives a $(2/3 - \epsilon) \cdot \log k$ approximation with probability exponentially small in k. This resolves the question for the case when the data is not assumed to be separable in the sense of Ostrovsky et al. [10]. However, the example that they construct satisfies $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \leq 1 + o(1)$ and hence does not satisfy the separability condition in the spirit of Ostrovsky et al.

Most of the above-mentioned results are for the k-means problem where the data set consists of points in $\mathbb{R}^d$ and the distance measure is the squared Euclidean distance. There are multiple instances in Machine Learning where the goal is to solve the problem with respect to other distance measures. Some examples include the Kullback-Leibler divergence, Mahalanobis distance, Itakura-Saito divergence. We can ask the same questions for the k-median problem with

respect to these distance measures. Ackermann and Blömer [2] analysed the sampling algorithm, **SampAlg**, with respect to a general class of distance measures called the μ-similar Bregman divergences. They show that if the data set satisfies the separation condition in the spirit of Ostrovsky et al., (that is $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq c > 1$), then **SampAlg** gives an $O(1)$-approximation with probability $\Omega(2^{-2k})$.

In our work, we analyse the sampling algorithm for the case that the data is separable, i.e., $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq c$ for some constant $c > 1$. This separability condition has been argued to be reasonable when using k-means objective to cluster data since the condition implies that the data is "well-clusterable".

Our contribution. We show that given a data set that is separable, i.e., $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq 1 + \gamma$, for some constant γ, **SampAlg** gives an $O(1)$ approximation with probability $\Omega(1/k)$. Our analysis works for the k-median problem with respect to *any* distance measure that is a metric in some approximate sense. We will look at some conditions that the distance measure needs to satisfy below.

Definition 1 (α-approximate symmetry). *Let $0 < \alpha \leq 1$. Let $\mathcal{X}$ be some data domain and D be a distance measure with respect to $\mathcal{X}$. D is said to satisfy the α-approximate symmetry property if the following holds:*

$$\forall x, y \in \mathcal{X}, \alpha \cdot D(y, x) \leq D(x, y) \leq (1/\alpha) \cdot D(y, x). \tag{1}$$

Definition 2 (β-approximate triangle inequality). *Let $0 < \beta \leq 1$. Let $\mathcal{X}$ be some data domain and D be a distance measure with respect to $\mathcal{X}$. D is said to satisfy the β-approximate triangle inequality if the following holds:*

$$\forall x, y, z \in \mathcal{X}, D(x, z) \leq (1/\beta) \cdot (D(x, y) + D(y, z)). \tag{2}$$

Definition 3 (Centroid property). *A distance measure D over space $\mathcal{X}$ is said to satisfy the centroid property if for any subset $P \subseteq \mathcal{X}$ and any point $c \in \mathcal{X}$, we have:*

$$\sum_{p \in P} D(p, c) = \Delta_1(P) + |P| \cdot D(m(P), c),$$

where $m(P) = \frac{\sum_{p \in P} p}{|P|}$ denotes the mean of the points in P. Also, as mentioned earlier, $\Delta_1(P)$ denote the optimal cost with respect to 1 center.

Note that in the k-means problem, $\mathcal{X} = \mathbb{R}^d$ and $D(x, y) = ||x - y||^2$. This distance measure satisfies α-approximate symmetry and β-approximate triangle inequality for $\alpha = 1$ and $\beta = 1/2$. The squared Euclidean distance also satisfies the Centroid property. Note that the squared Euclidean distance is not the only distance measure, used for clustering in practice, that satisfies these properties. *Mahalanobis distance* also satisfies the above properties. A class of distance measures called *Bregman divergences* that are used frequently in Machine Learning is known to satisfy the Centroid property. Furthermore, an important sub-class of Bregman divergences, called μ-similar Bregman divergences, is known to satisfy

all of the above properties (see [1] for an overview of Bregman divergences). We can now give our main result using the above definitions:

Theorem 1 (Main Theorem). *Let $0 < \alpha, \beta \leq 1$ and $\gamma = \frac{32}{(\alpha\beta)^4}$ be constants. Let D be a distance measure over space $\mathcal{X}$ such that D satisfies α-approximate symmetry, β-approximate trianlge inequality, and the Centroid property. Let $P \subseteq \mathcal{X}$ be any set of n points from the space $\mathcal{X}$ such that the following holds:*

$$\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq 1 + \gamma, \tag{3}$$

where $\Delta_i(P)$ is defined to be the optimal value of the objective function with i centers, i.e., $\Delta_i(P) = \min_{C,|C|=i} \left[\sum_{p\in P} \min_{c\in C} D(p,c)\right]$. Then **SampAlg** *gives an $O(1)$-approximation with probability $\Omega(1/k)$.*

We also show that when the data is not given to be separable, then **SampAlg** gives an $O(1)$ approximation with probability $\Omega(2^{-2k})$. Note that this is for any k-median instance with respect to any distance measure that satisfy α-symmetry and β-triangle inequality[1]. This is an improvement over the result by Ackermann Blömer [2] who get a similar result though for separable data, i.e. $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq c$ for some fixed constant c. We discuss this result in Section 3.

Techniques. Here is an outline of the proof of our Main Theorem. Let $\{A_1, ..., A_k\}$ denote the points in the optimal clustering. From the Centroid property, we know that the centroids $\{c_1, ..., c_k\}$ of $\{A_1, ..., A_k\}$ are the optimal centers. Let $d_{ij} = D(c_i, c_j)$ and let $T_{min} = \min_{i\neq j} [|A_i| \cdot d_{ij}]$. Let C' denote any set of i points chosen by the first i iterations of the algorithm. Let j be the index of an optimal cluster such that no point in C' belongs to A_j. We will first argue that $\phi_{C'}(A_j) \geq d \cdot T_{min}$ (for some constant d) by showing that if this were not the case, then the separability condition is violated. Let X_i denote the points in those optimal clusters such that C' has a point from that cluster and let $\bar{X}_i$ denote the remaining points. From the previous argument, we know that $\phi_{C'}(\bar{X}_i) \geq (k-i) \cdot d \cdot T_{min}$. On the other hand, we can argue that the expected cost of the centers C' w.r.t. X_i is at most $d' \cdot \Delta_k(P)$ (for some constant d'). Then we show that the probability of picking the $(i+1)^{th}$ point from $\bar{X}_i$ is at least $\frac{k-i}{k-i+1}$. Note that this probability is proportional to $\frac{\phi_{C'}(\bar{X}_i)}{\phi_{C'}(P)}$ and if this were smaller than $\frac{k-i}{k-i+1}$, then $\frac{T_{min}}{\Delta_k(P)} \leq d''$ (for some constant d'') but this contradicts with the separability condition. So, the probability that we pick points from each optimal cluster is $\Omega(1/k)$ (using telescoping product). Conditioned on this event, we will argue that the expected cost is at most some constant times the optimal.

We now focus on the proof of our Main Theorem

2 Proof of Theorem 1

Let $A_1, ..., A_k$ denote the optimal clusters, i.e., the point set P is partitioned into subsets $A_1, ..., A_k$ such that all points in A_i are in the i^{th} cluster as per

[1] The Centroid property is not required for this result.

the optimal k-median clustering. Let $C_{OPT} = \{c_1, ..., c_k\}$ be the optimal cluster centers. So, $\forall i \neq j, p \in A_j, D(p, c_i) \geq D(p, c_j)$. For any set of centers C, we denote the distance of a point p to it nearest center in C with $D(p, C)$. For any optimal cluster A_i, let $r_i = \frac{\sum_{p \in A_i} D(p,c_i)}{|A_i|}$.

We will need the following two basic lemmas. These are generalizations of Lemmas 3.1 and 3.2 in [5].

Lemma 1. *Consider any optimal cluster A_i. Let c be a point chosen from A_i uniformly at random. Then we have* $\mathbf{Exp}[\phi_{\{c\}}(A_i)] \leq \frac{2}{\alpha\beta} \cdot \phi_{\{c_i\}}(A_i)$.

Proof. The expected cost may be written as:

$$\begin{aligned}
\mathbf{Exp}[\phi_{\{c\}}(A_i)] &= \sum_{p \in A_i} \frac{1}{|A_i|} \cdot \sum_{q \in A_i} D(q, p) \\
&\leq \sum_{p \in A_i} \frac{1}{|A_i|} \cdot \sum_{q \in A_i} (1/\beta) \cdot (D(q, c_i) + D(c_i, p)) \\
&\leq \sum_{p \in A_i} \frac{1}{|A_i|} \cdot \sum_{q \in A_i} (1/\beta) \cdot (D(q, c_i) + (1/\alpha) \cdot D(p, c_i)) \\
&= \sum_{p \in A_i} \frac{1}{|A_i|} \cdot \left[\frac{\phi_{\{c_i\}}(A_i)}{\beta} + \frac{|A_i|}{\alpha\beta} \cdot D(p, c_i)) \right] \leq \frac{2}{\alpha\beta} \cdot \phi_{\{c_i\}}(A_i)
\end{aligned}$$

□

Lemma 2. *Let C be any set of centers. Consider any optimal cluster A_i. Let c be a center chosen using non-uniform sampling with respect to the set C and let $C' = C \cup \{c\}$. The we have* $\mathbf{Exp}[\phi_{C'}(A_i) | c \in A_i] \leq \frac{4}{(\alpha\beta)^2} \cdot \phi_{\{c_i\}}(A_i)$.

Proof. The probability that we choose a point $p \in A_i$ to be c, conditioned on the fact that c is chosen from A_i is given by $\frac{D(p,C)}{\sum_{q \in A_i} D(q,C)}$. Once we choose p to be c, then any point $q' \in A_i$ contributes $\min(D(q', C), D(q', c))$ to the cost. Using these two observations, we get the following:

$$\mathbf{Exp}[\phi_{C'}(A_i) | c \in A_i] = \sum_{p \in A_i} \frac{D(p, C)}{\sum_{q \in A_i} D(q, C)} \cdot \sum_{q' \in A_i} \min(D(q', C), D(q', p)) \quad (4)$$

From β-approximate triangle inequality, we have that $D(p, C) \leq (1/\beta) \cdot (D(p, q'') + D(q'', C))$ for all $q'' \in A_i$. So, we have

$$D(p, C) \leq \frac{1}{\beta |A_i|} \cdot \left(\sum_{q'' \in A_i} D(p, q'') + \sum_{q'' \in A_i} D(q'', C) \right) \quad (5)$$

Using above in (4), we get the following:

$$\begin{aligned}
\mathbf{Exp}[\phi_{C'}(A_i)|c \in A_i] &\leq \frac{1}{\beta|A_i|} \cdot \sum_{p \in A_i} \frac{\sum_{q'' \in A_i} D(p,q'')}{\sum_{q \in A_i} D(q,C)} \cdot \sum_{q' \in A_i} D(q',C) + \\
&\quad \frac{1}{\beta|A_i|} \cdot \sum_{p \in A_i} \frac{\sum_{q'' \in A_i} D(q'',C)}{\sum_{q \in A_i} D(q,C)} \cdot \sum_{q' \in A_i} D(q',p) \\
&= \frac{1}{\beta|A_i|} \cdot \sum_{p \in A_i} \sum_{q'' \in A_i} D(p,q'') + \frac{1}{\beta|A_i|} \cdot \sum_{p \in A_i} \sum_{q' \in A_i} D(q',p) \\
&\leq \frac{1}{\beta|A_i|} \cdot \sum_{p \in A_i} \sum_{q'' \in A_i} D(p,q'') + \frac{1}{\alpha\beta|A_i|} \cdot \sum_{p \in A_i} \sum_{q' \in A_i} D(p,q') \\
&\qquad \text{(using (1))} \\
&\leq \frac{2}{\alpha\beta} \cdot \frac{1}{|A_i|} \sum_{p \in A_i} \sum_{q \in A_i} D(p,q) \\
&\leq \frac{2}{\alpha\beta^2} \cdot \frac{1}{|A_i|} \sum_{p \in A_i} \sum_{q \in A_i} (D(p,c_i) + D(c_i,q)) \quad \text{(using (2))} \\
&\leq \frac{2}{(\alpha\beta)^2} \cdot \frac{1}{|A_i|} \sum_{p \in A_i} \sum_{q \in A_i} (D(p,c_i) + D(q,c_i)) \quad \text{(using (1))} \\
&= \frac{4}{(\alpha\beta)^2} \cdot \phi_{\{c_i\}}(A_i)
\end{aligned}$$

□

The above lemma says that conditioned on picking the next center from a cluster A_i, the expected cost of this cluster with respect to the currently chosen centers is within $O(1)$ factor of the optimal. So, in general once we pick a center from an optimal cluster, there is good chance that we may be able to "forget" about this cluster in the future as we already have a constant approximation with respect to this cluster. The issue might be that the given a current set of centers C, the probability of sampling the next center from a given cluster might be very small. We show that if this happens, then the separation condition is violated.

Let $C_i = \{c'_{j_1}, ..., c'_{j_i}\}$ be the centers chosen in the first i steps of the sampling algorithm, where $J_i = \{j_1, ..., j_i\}$ denotes the subset of indices of the optimal cluster to which the centers belongs. Let $X_i = \cup_{j \in J_i} A_j$. Let E_i be the event that J_i contains i distinct indices, i.e., the cardinality of J_i is i. We will later show that $\forall i, Pr[E_i] \geq \frac{k-i+1}{k}$.

First, we show that the expected cost of C_i with respect to the point set X_i is at most some constant times the cost of C_{OPT} with respect to X_i.

Lemma 3. $\forall i, \mathbf{Exp}[\phi_{C_i}(X_i) \mid E_i] \leq \frac{4}{(\alpha\beta)^2} \cdot \phi_{C_{OPT}}(X_i)$.

Proof. The proof follows from Lemmas 1 and 2. □

In the next Lemma, we get a lower bound on the probability that the cost of the solution given by the sampling algorithm is at most some constant times the cost of the optimal solution.

Lemma 4. $\mathbf{Pr}\left[\phi_{C_k}(P) \leq \frac{8}{(\alpha\beta)^2} \cdot \phi_{C_{OPT}}(P)\right] \geq (1/2) \cdot \mathbf{Pr}[E_k]$.

Proof. Given that event E_k happens, we have $X_k = P$ and from Lemma 3, we get that $\mathbf{Exp}[\phi_{C_k}(P) \mid E_k] \leq \frac{4}{(\alpha\beta)^2} \cdot \phi_{C_{OPT}}(P)$. By Markov, we get that

$$\mathbf{Pr}\left[\phi_{C_k}(P) > (8/(\alpha\beta)^2) \cdot \phi_{C_{OPT}}(P) \mid E_k\right] \leq 1/2.$$

Removing the conditioning on E_k we get the desired Lemma. □

Now, all we need to show is that $\mathbf{Pr}[E_k] \geq 1/k$. This trivially follows from Lemma 6 that shows that $\mathbf{Pr}[E_{i+1} \mid E_i] \geq \frac{k-i}{k-i+1}$.

We will need the some additional definitions. Let $\bar{X}_i = P \setminus X_i$. Let $\bar{J}_i = [k] \setminus J_i$. Note that conditioned on E_i happening, $|\bar{J}_i| = k - i$. For any $s \in \bar{J}_i$ let I_s denote the index $t \in J_i$ such that $D(c_s, c'_t)$ is minimized. Let $V_s = D(c_s, c_{I_s})$. We know that

$$D(c'_{I_s}, c_{I_s}) \leq D(c'_{I_s}, c_s) \quad \text{and} \quad V_s \leq (1/\beta) \cdot (D(c_s, c'_{I_s}) + D(c'_{I_s}, c_{I_s}))$$

The first inequality is due to the fact that $c'_{I_s} \in A_{I_s}$ (hence is c'_{I_s} is closer to the center of A_{I_s} than of A_s). The above inequality gives us the following:

$$V_s \leq (1/\beta) \cdot (D(c_s, c'_{I_s}) + (1/\alpha) \cdot D(c_s, c'_{I_s})) \leq \frac{2}{\alpha\beta} \cdot D(c_s, c'_{I_s}) \tag{6}$$

Let $T_s = |A_s| \cdot V_s$. Let $T_{min} = \min_{i \neq j} |A_i| \cdot D(c_i, c_j)$. Note that

$$\forall s \in \bar{J}_i, T_s \geq T_{min}. \tag{7}$$

Using the above definitions we can show the following Lemma.

Lemma 5. $\phi_{C_i}(\bar{X}_i) \geq (k - i) \cdot \frac{(\alpha\beta)^2}{8} \cdot T_{min}$

Proof. For any s, let A_s^{in} denote those data points that are closer to the center c_s than any data point that does not belong to A_s, i.e.,

$$A_s^{in} = \{p | p \in A_s \text{ and } \forall q \notin A_s, D(p, c_s) \leq D(p, q)\}$$

Let the remaining points in A_s be denoted by A_s^{out}, i.e., $A_s^{out} = A_s \setminus A_s^{in}$. Next, we will argue that if the data is separable,i.e., $\Delta_{k-1}(P)/\Delta_k(P) \geq 1 + \gamma$, then $|A_s^{in}| \geq |A_s^{out}|$.

Claim. Let $\gamma = \frac{32}{(\alpha\beta)^4}$. If $\Delta_{k-1}(P)/\Delta_k(P) > 1 + \gamma$, then $\forall s, |A_s^{in}| \geq |A_s^{out}|$.

Proof. Consider any point $p \in A_s^{out}$. Let $N[p]$ denote the point $\notin A_s$ that is nearest to p and let $I[p]$ denote the index of the cluster to which $N[p]$ belongs. We note that the following inequalities hold:

$$\begin{aligned}
D(p, c_{I[p]}) &\leq \frac{1}{\beta}\left(D(p, N[p]) + D(N[p], c_{I[p]})\right) \quad \text{(using (2))} \\
&\leq \frac{1}{\beta}\left(D(p, c_s) + D(N[p], c_{I[p]})\right) \quad \text{(since } D(p, N[p]) \leq D(p, c_s)\text{)} \\
&\leq \frac{1}{\beta}\left(D(p, c_s) + D(N[p], c_s)\right) \quad \text{(since } D(N[p], c_{I[p]}) \leq D(N[p], c_s)\text{)} \\
&\leq \frac{1}{\beta}\left(D(p, c_s) + \frac{1}{\beta}(D(N[p], p) + D(p, c_s))\right) \quad \text{(using (2))} \\
&\leq \frac{1}{\beta}\left((1 + \frac{1}{\beta})D(p, c_s) + \frac{1}{\alpha\beta}D(p, N[p])\right) \quad \text{(using (1))} \\
&\leq \frac{1}{\beta}\left((1 + \frac{1}{\beta})D(p, c_s) + \frac{1}{\alpha\beta}D(p, c_s)\right) \quad \text{(since } D(p, N[p]) \leq D(p, c_s)\text{)} \\
&\leq \frac{3}{\alpha\beta^2}D(p, c_s) \qquad (8)
\end{aligned}$$

For the sake of contradiction, let us assume that $|A_s^{in}| < |A_s^{out}|$. Let f be any one-one function that maps data points in A_s^{in} to data points in A_s^{out}.

For any point $p \in A_s^{in}$, the following inequalities hold:

$$\begin{aligned}
D(p, c_{I[f(p)]}) &\leq \frac{1}{\beta}\left(D(p, f(p)) + D(f(p), c_{I[f(p)]})\right) \quad \text{(using (2))} \\
&\leq \frac{1}{\beta}\left(\frac{1}{\beta}\left(D(p, c_s) + D(c_s, f(p))\right) + D(f(p), c_{I[f(p)]})\right) \quad \text{(using (2))} \\
&\leq \frac{1}{\beta}\left(\frac{1}{\beta}\left(D(p, c_s) + \frac{1}{\alpha}D(f(p), c_s)\right) + D(f(p), c_{I[f(p)]})\right) \quad \text{(using (1))} \\
&\leq \frac{1}{\beta}\left(\frac{1}{\beta}\left(D(p, c_s) + \frac{1}{\alpha}D(f(p), c_s)\right) + \frac{3}{\alpha\beta^2}D(f(p), c_s)\right) \quad \text{(using (8))} \\
&= \left(\frac{1}{\beta^2}\, D(p, c_s) + \frac{4}{\alpha\beta^2}\, D(f(p), c_s)\right) \qquad (9)
\end{aligned}$$

Using (8) and (9), we get the following:

$$\begin{aligned}
\sum_{p \in A_s^{in}} D(p, c_{I[f(p)]}) + \sum_{p \in A_s^{out}} D(p, c_{I[p]}) &\leq \frac{1}{\beta^2}\sum_{p \in A_s^{in}} D(p, c_s) + \frac{4}{\alpha\beta^2}\sum_{p \in A_s^{in}} D(f(p), c_s) \\
&\quad + \frac{3}{\alpha\beta^2}\sum_{p \in A_s^{out}} D(p, c_s) \\
&\leq \frac{1}{\beta^2}\sum_{p \in A_s^{in}} D(p, c_s) + \frac{7}{\alpha\beta^2}\sum_{p \in A_s^{out}} D(p, c_s) \\
&\leq \frac{8}{\alpha\beta^2}\sum_{p \in A_s} D(p, c_s) = \frac{8}{\alpha\beta^2}|A_s|r_s \qquad (10)
\end{aligned}$$

The second inequality above is due to the fact that f is one-one. Using (10), we get that

$$\frac{\phi_{\{c_1,\ldots,c_k\}\setminus c_s}(P)}{\phi_{\{c_1,\ldots,c_k\}}(P)} = \frac{\sum_{t\in[k]\setminus\{s\}} |A_t|\cdot r_t + \frac{8}{\alpha\beta^2}\cdot|A_s|\cdot r_s}{\sum_{t\in[k]}|A_t|\cdot r_t} \leq \frac{8}{\alpha\beta^2}$$

This contradicts with the fact that $\Delta_{k-1}(P)/\Delta_k(P) \geq 1+\gamma = 1+\frac{32}{(\alpha\beta)^4}$. This concludes the proof of the claim. □

We use the above claim to prove the Lemma. For any $s \in \bar{J}_i$ and $p \in A_s^{in}$ we have

$$\begin{aligned}
&\frac{1}{\beta}\left(D(p,C_i)+D(c_s,p)\right) \geq D(c_s,C_i) \quad \text{(using (2))}\\
&\Rightarrow \frac{1}{\beta}\left(D(p,C_i)+\frac{1}{\alpha}D(p,c_s)\right) \geq D(c_s,C_i) \quad \text{(using (1))}\\
&\Rightarrow \frac{1}{\beta}\left(D(p,C_i)+\frac{1}{\alpha}D(p,C_i)\right) \geq D(c_s,C_i) \quad \text{(using definition of } A_s^{in})\\
&\Rightarrow \frac{2}{\alpha\beta}D(p,C_i) \geq D(c_s,C_i)\\
&\Rightarrow D(p,C_i) \geq \frac{\alpha\beta}{2}D(c_s,C_i)\\
&\Rightarrow D(p,C_i) \geq \frac{\alpha\beta}{2}D(c_s,c'_{I_s})\\
&\Rightarrow D(p,C_i) \geq \frac{(\alpha\beta)^2}{4}D(c_s,c_{I_s}) \quad \text{(using (6))}\\
&\Rightarrow D(p,C_i) \geq \frac{(\alpha\beta)^2}{4}V_s
\end{aligned}$$

From this we get the following:

$$\begin{aligned}
&\sum_{p\in A_s^{in}} D(p,C_i) \geq \frac{(\alpha\beta)^2}{4}\cdot\frac{|A_s|}{2}\cdot V_s \quad \text{(since } |A_s^{in}| \geq |A_s|/2 \text{ from previous claim)}\\
&\Rightarrow \sum_{p\in A_s} D(p,C_i) \geq \frac{(\alpha\beta)^2}{8}\cdot T_{min} \quad \text{(using (7))}\\
&\Rightarrow \sum_{s\in\bar{J}_i}\sum_{p\in A_s} D(p,C_i) \geq (k-i)\cdot\frac{(\alpha\beta)^2}{8}\cdot T_{min} \quad \text{(since } |\bar{J}_i| \geq (k-i))\\
&\Rightarrow \phi_{C_i}(\bar{X}_i) \geq (k-i)\cdot\frac{(\alpha\beta)^2}{8}\cdot T_{min}
\end{aligned}$$

This concludes the proof of Lemma 5. □

Lemma 6. $\forall i, \mathbf{Pr}[E_{i+1} \mid E_i] \geq \frac{k-i}{k-i+1}$

Proof. $\mathbf{Pr}[E_{i+1} \mid E_i]$ is just the conditional probability that the $(i+1)^{th}$ center is chosen from the set $\bar{X}_i$ given that the first i centers are chosen from i different optimal clusters. This probability can be expressed as

$$\mathbf{Pr}[E_{i+1} \mid E_i] = \mathbf{Exp}\left[\frac{\phi_{C_i}(\bar{X}_i)}{\phi_{C_i}(P)} \mid E_i\right] \tag{11}$$

For the sake of contradiction, let us assume that

$$\mathbf{Exp}\left[\frac{\phi_{C_i}(\bar{X}_i)}{\phi_{C_i}(P)} \mid E_i\right] = \mathbf{Pr}[E_{i+1} \mid E_i] < \frac{k-i}{k-i+1} \tag{12}$$

Applying Jensen's inequality, we get the following:

$$\frac{1}{\mathbf{Exp}\left[\frac{\phi_{C_i}(P)}{\phi_{C_i}(\bar{X}_i)} \mid E_i\right]} \leq \mathbf{Exp}\left[\frac{\phi_{C_i}(\bar{X}_i)}{\phi_{C_i}(P)} \mid E_i\right] < \frac{k-i}{k-i+1}$$

This gives the following:

$$\begin{aligned}
1 + \frac{1}{k-i} &< \mathbf{Exp}\left[\frac{\phi_{C_i}(P)}{\phi_{C_i}(\bar{X}_i)} \mid E_i\right] \\
&= \mathbf{Exp}\left[\frac{\phi_{C_i}(X_i) + \phi_{C_i}(\bar{X}_i)}{\phi_{C_i}(\bar{X}_i)} \mid E_i\right] \\
&= 1 + \mathbf{Exp}\left[\frac{\phi_{C_i}(X_i)}{\phi_{C_i}(\bar{X}_i)} \mid E_i\right] \\
\Rightarrow \frac{1}{k-i} &\leq \mathbf{Exp}\left[\frac{\phi_{C_i}(X_i)}{\frac{(\alpha\beta)^2}{8} \cdot (k-i) \cdot T_{min}} \mid E_i\right] \quad \text{(using Lemma 5)} \\
&\leq \frac{\mathbf{Exp}[\phi_{C_i}(X_i) \mid E_i]}{\frac{(\alpha\beta)^2}{8} \cdot (k-i) \cdot T_{min}} \\
&\leq \frac{\frac{4}{(\alpha\beta)^2} \cdot \phi_{C_{OPT}}(P)}{\frac{(\alpha\beta)^2}{8} \cdot (k-i) \cdot T_{min}} \quad \text{(using Lemma 3)} \\
\Rightarrow \frac{T_{min}}{\phi_{C_{OPT}}(P)} &\leq \frac{32}{(\alpha\beta)^4}
\end{aligned} \tag{13}$$

Let I_{min} be the index for which $\min_{j \neq I_{min}}(|A_{I_{min}}| D(c_{I_{min}}, c_j))$ is minimized. Note that $T_{min} = \min_{j \neq I_{min}}(|A_{I_{min}}| D(c_{I_{min}}, c_j))$. Consider the set $C' = \cup_{s \neq I_{min}}\{c_s\}$, i.e., all centers except the center of the I^{th}_{min} cluster. We will compute the cost of C' with respect to P:

$$\begin{aligned}
\frac{\phi_{C'}(P)}{\phi_{C_{OPT}}(P)} &\leq \frac{\phi_{C_{OPT}}(P) + T_{min}}{\phi_{C_{OPT}}(P)} \quad \text{(using Centroid property)} \\
&\leq 1 + \frac{32}{(\alpha\beta)^4} \quad \text{(using (13))}
\end{aligned}$$

This contradicts with the fact that P satisfies $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} > 1 + \frac{32}{(\alpha\beta)^4}$. □

3 Analysis of SampAlg without Separation Condition

In this section, we will show that **SampAlg** gives an $O(1)$ approximation with probability $\Omega(2^{-2k})$ for any data set. This holds with respect to any distance measure that satisfies the α-symmetry and β-triangle inequality. Note that the Centroid property is not required. This is stated more formally in the next Theorem.

Theorem 2. *Let $0 < \alpha, \beta \leq 1$ be constants. Let D be a distance measure over space $\mathcal{X}$ such that D satisfies α-approximate symmetry and β-approximate triangle inequality. Let $P \subseteq \mathcal{X}$ be any set of n points from the space $\mathcal{X}$. Then* **SampAlg** *gives an $O(1)$-approximation with probability $\Omega(2^{-2k})$.*

Proof. We will use the definitions and notations from the previous Section. Given a set of centers C_i, we say that an optimal cluster A_j is "covered" if there exists a center $c' \in C$ such that $\phi_{\{c'\}}(A_j) \leq \frac{8}{(\alpha\beta)^2}\phi_{\{c_j\}}(A_j)$. Note that if there is a set of centers C' such that all the optimal clusters are covered, then $\phi_{C'}(P) \leq \frac{8}{(\alpha\beta)^2}\phi_{C_{OPT}}(P)$. We will show that, with probability $\Omega(2^{-2k})$, either C_k covers all the optimal clusters or gives a constant approximation. Recall that C_i denotes the set of centers after i centers are picked. Let R_i denote the set of indices of optimal clusters that are covered by C_i. Let $Y_i = \cup_{j \in R_i} A_j$ and $\bar{Y}_i = P \setminus Y_i$. The probability that $(i+1)^{th}$ chosen center covers a previously uncovered cluster is given by $\frac{\phi_{C_i}(\bar{Y}_i)}{\phi_{C_i}(P)}$. Suppose that $\frac{\phi_{C_i}(\bar{Y}_i)}{\phi_{C_i}(P)} < 1/2$. This implies that $\phi_{C_i}(\bar{Y}_i) < \phi_{C_i}(Y_i)$. This further implies that

$$\phi_{C_i}(P) = \phi_{C_i}(\bar{Y}_i) + \phi_{C_i}(Y_i) < 2\phi_{C_i}(Y_i) \leq \frac{16}{(\alpha\beta)^2}\phi_{C_{OPT}}(Y_i) \leq \frac{16}{(\alpha\beta)^2}\phi_{C_{OPT}}(P).$$

The above basically means that the current set of centers already gives a constant approximation with respect to the entire point set P. Choosing more centers will only lower the cost. On the other hand, if $\frac{\phi_{C_i}(\bar{Y}_i)}{\phi_{C_i}(P)} \geq 1/2$, then this implies that with probability at least $1/2$ the $(i+1)^{th}$ center is from one of the uncovered clusters. Conditioned on this, from Lemma 2 we know that with probability at least $1/2$, the newly chosen center covers a previously uncovered cluster. So, with probability at least $1/4$, a new cluster gets covered in step $(i+1)$.

So, either the set of chosen centers C_k gives an approximation factor of $\frac{16}{(\alpha\beta)^2}$ or with probability at least 2^{-2k} covers all optimal clusters. The latter implies that C_k gives $\frac{8}{(\alpha\beta)^2}$ approximation. So, in summary, **SampAlg** gives an $\frac{16}{(\alpha\beta)^2}$-approximation with probability at least 2^{-2k}. □

4 Conclusions and Open Problems

In this paper, we have shown that given that the data is separable in the spirit of Ostrovsky et al. [10], i.e., $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} \geq 1 + \gamma_1$ (for some fixed constant γ_1), then the k-means++ based sampling algorithm **SampAlg** gives an $O(1)$ approximation with

probability $\Omega(1/k)$. On the other hand, Brunsch and Röglin [8] gave an instance where **SampAlg** gives $(2/3-\epsilon)\log k$ approximation with probability exponentially small in k. However, their instance is not separable, i.e., $\frac{\Delta_{k-1}(P)}{\Delta_k(P)} = 1+\gamma_2$, where $\gamma_2 = o(1)$ and use high dimension. Some interesting open questions are:

- How does **SampAlg** behave when $1+\gamma_2 \leq \frac{\Delta_{k-1}(P)}{\Delta_k(P)} \leq 1+\gamma_1$?
- How does **SampAlg** behave for planar k-median instances (or in general low dimensional instances)?

The planar (dimension = 2) k-means problem was shown to be NP-hard by Mahajan et al. [9]. The lower-bound instances constructed by Arthur and Vassilvitskii [5], Aggarwal et al. [3], and Brunsch and Röglin [8] use high dimension. So, it may be possible that **SampAlg** gives $O(1)$ with high probability for any planar k-means instances. Another interesting direction is to explore the behavior of **SampAlg** when the data satisfies (c,ϵ)-closeness property of Balcan et al. [6]. This property was argued to be weaker than the separability condition of Ostrovsky et al. [10].

References

1. Ackermann, M.R.: Algorithms for the Bregman k-Median Problem. PhD thesis, University of Paderborn, Department of Computer Science (2009)
2. Ackermann, M.R., Blömer, J.: Bregman Clustering for Separable Instances. In: Kaplan, H. (ed.) SWAT 2010. LNCS, vol. 6139, pp. 212–223. Springer, Heidelberg (2010)
3. Aggarwal, A., Deshpande, A., Kannan, R.: Adaptive Sampling for k-Means Clustering. In: Dinur, I., Jansen, K., Naor, J., Rolim, J. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 15–28. Springer, Heidelberg (2009)
4. Ailon, N., Jaiswal, R., Monteleoni, C.: Streaming k-means approximation. In: Advances in Neural Information Processing Systems, vol. 22, pp. 10–18 (2009)
5. Arthur, D., Vassilvitskii, S.: k-means++: the advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), pp. 1027–1035 (2007)
6. Balcan, M.-F., Blum, A., Gupta, A.: Approximate clustering without the approximation. In: Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2009), pp. 1068–1077 (2009)
7. Banerjee, A., Merugu, S., Dhillon, I.S., Ghosh, J.: Clustering with Bregman divergences. Journal of Machine Learning Research 6, 1705–1749 (2005)
8. Brunsch, T., Röglin, H.: A Bad Instance for k-Means++. In: Ogihara, M., Tarui, J. (eds.) TAMC 2011. LNCS, vol. 6648, pp. 344–352. Springer, Heidelberg (2011)
9. Mahajan, M., Nimbhorkar, P., Varadarajan, K.: The Planar k-Means Problem is NP-Hard. In: Das, S., Uehara, R. (eds.) WALCOM 2009. LNCS, vol. 5431, pp. 274–285. Springer, Heidelberg (2009)
10. Ostrovsky, R., Rabani, Y., Schulman, L.J., Swamy, C.: The effectiveness of lloyd-type methods for the k-means problem. In: Proc. 47th IEEE FOCS, pp. 165–176 (2006)

A Sharper Local Lemma with Improved Applications

Kashyap Kolipaka, Mario Szegedy, and Yixin Xu

Department of Computer Science
Rutgers, The State University of New Jersey
{kolipaka,szegedy,yixinxu}@cs.rutgers.edu

Abstract. We give a new family of Lovász Local Lemmas (LLL), with applications. Shearer has given the most general condition under which the LLL holds, but the original condition of Lovász is simpler and more practical. Do we have to make a choice between practical and optimal? In this article we present a continuum of LLLs between the original and Shearer's conditions. One of these, which we call Clique LLL (CLLL), particularly stands out, and is natural in those settings, where the event space is defined with discrete independent random variables (á la Moser and Tardos). Using this version we get improved bounds in applications for Acyclic Edge Coloring and Non-repetitive Vertex Coloring.

Keywords: Lovász Local Lemma, Independent set polynomial, Hard-core lattice model, Graph coloring.

1 Introduction

The setting for the Lovász Local Lemma(LLL) has a collection of *bad* events $\{A_1, A_2, \ldots, A_n\}$ and a dependency graph G on the node set $[n] = \{1, 2, \ldots, n\}$. For $i \in [n]$, let $N(i)$ denote the set of neighbors of the node i. The dependency graph gives the information that the event A_i is independent of the system of events generated by $\{A_j \mid j \notin N(i)\}$. The motivation for the LLL is to find sufficient conditions such that $\text{Prob}(\bigcap_{i\in[n]} \overline{A}_i) > 0$. The idea of the Local Lemma was first circulated by Laci Lovász in the early 1970s, in an unpublished note. It was first published in [EL75] and was subsequently sharpened in a result of Spencer [Spe77] to the following popular form:

Theorem 1 (Original LLL [EL75, Spe77]). *If there exist* $x_i \in (0, 1)$ *(*$i \in [n]$*), such that*

$$\forall i \in [n] : \ \text{Prob}(A_i) \leq x_i \prod_{j\in N(i)} (1 - x_j),$$

then $\text{Prob}(\overline{A}_1 \wedge \ldots \wedge \overline{A}_n) > 0$.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 603–614, 2012.
© Springer-Verlag Berlin Heidelberg 2012

1.1 The Variable Framework

In most applications of the LLL the events are determined by a set of independent discrete random variables, $vbl = \{v_1, v_2, \ldots, v_m\}$. Each event A_i is completely determined by some subset of variables, $vbl(i) \subseteq vbl$. The dependency graph in such applications is constructed in a canonical way by adding the edge (i, j) if and only if $vbl(i) \cap vbl(j) \neq \emptyset$. We will call this setting of the LLL the *variable framework*. LLL has many applications in this framework besides the well-known k-SAT.

Acyclic Edge Coloring. Given an undirected graph G, an *acyclic edge coloring* is a proper edge coloring such that no cycle is 2-edge-colored. The *acyclic edge chromatic number* of G is the minimum number of colors in an acyclic edge coloring of G and is denoted by $a'(G)$. Let $a'(d)$ denote the maximum value $a'(G)$ over all graphs with max-degree at most d. Acyclic colorings were studied in a series of works [AMR91, Grü73, MR98, MNS07]. Alon, McDiarmid and Reed [AMR91] consider acyclic edge colorings and among other things, use the LLL to prove that $a'(d) < 64d$. This bound was later improved by Molloy and Reed [MR98] to $16d$, again using the LLL. It is known that $a'(d) \geq d + 2$ and Alon, Sudakov and Zaks [ASZ01] conjecture that in fact $a'(d) = d+2$. They also prove the conjecture for graphs with girth $\Omega(d \log d)$. In further efforts towards proving this conjecture, Muthu, Narayanan and Subramanian [MNS07] showed that $a'(d) < 4.52d$ for graphs with girth at least 220 and $a'(d) < 6d$ for graphs with girth at least 9. More recently, Haeupler, Saha and Srinivasan [HSS10] made all the above results constructive by extending the algorithm of Moser and Tardos [MT10] to the case of an exponential number of events.

Non-repetitive Vertex Coloring. Given an undirected graph G, a *nonrepetitive vertex coloring* is a proper coloring of the vertices of G such that there is no simple path with an even number of vertices, such that, the sequence of colors in the first half is same as the sequence of colors in the second half. The minimum number of colors in such a coloring for G is called the *Thue number* of G, denoted by $\pi(G)$. The original result regarding Thue numbers was due to Thue in [Thu06] who proved that $\pi(G) \leq 4$ if G is a tree. There have been several works related to the Thue numbers in [AG08, BGK+07, Cur05, Gry08]. A special case of interest for Thue numbers is when the degrees are bounded by d. Let $\pi(d) = \max\{\pi(G) \mid \text{max-degree of G } \leq d\}$. Alon, Grytczuk, Hauszczak and Riordan [AGHR02] proved that $\pi(d)$ is in $O(d^2)$ and $\Omega(\frac{d^2}{\log d})$. Their proof of the upper bound uses the LLL. Grytczuk in [Gry07] shows that this can be improved to $16d^2$, which is basically the same random coloring as [AGHR02] with a more optimized application of the LLL. In a recent development, Dujmovic, Joret, Kozik and Wood [DJKW12] have shown that $\pi(d) \leq (1 + o(1))d^2$, using the entropy-compression method of Moser and Tardos [MT10].

1.2 Shearer's Bound

Given the variety of applications of the LLL, it is natural to ask if we could get even better bounds for the listed problems by improving on the LLL. The conditions given in Theorem 1 are indeed not optimal for a fixed graph. Shearer found the exact characterization of those sequences $p = (p_1, \ldots, p_n)$ of probabilities for which $\text{Prob}(\overline{A}_1 \wedge \ldots \wedge \overline{A}_n) > 0$ whenever $\text{Prob}(A_i) = p_i$, and $\{A_i\}_{i=1}^n$ has dependency graph G [She85, SS06]. To describe this characterization, let $\text{Indep}(G)$ denote the set of all independent sets of G, including the empty set, and define the quantities

$$q_I = q_I(G,p) = \sum_{J \in \text{Indep}(G),\ I \subseteq J} (-1)^{|J|-|I|} \prod_{i \in J} p_i \tag{1}$$

for any $I \in \text{Indep}(G)$.

Theorem 2 (Shearer, [She85]). *Let G be a dependency graph on $[n]$. Then for a vector $p = (p_1, \ldots, p_n)$ of non-zero probabilities the following are equivalent:*

1. *For every system $\{A_i\}_{i=1}^n$ of events with $\text{Prob}(A_i) = p_i$ $(1 \le i \le n)$ with dependency graph G it holds that $\text{Prob}(\overline{A}_1 \wedge \ldots \wedge \overline{A}_n) > 0$;*
2. *$q_I(G,p) > 0$ for all $I \in \text{Indep}(G)$.*

Furthermore, when the above holds, there exists $\{B_i\}_{i=1}^n$ such that $\text{Prob}(B_i) = p_i$ $(1 \le i \le n)$ with dependency graph G such that for every independent set I of G:

$$\text{Prob}(\bigwedge_{i \in I} B_i \wedge \bigwedge_{i \notin I} \overline{B_i}) = q_I(G,p).$$

This is the unique instance that minimizes $\text{Prob}(\overline{A}_1 \wedge \ldots \wedge \overline{A}_n)$, and it also has the property that all neighboring events are disjoint. We call this the extreme instance.

Shearer's Bound and Statistical Mechanics. The *symmetric Shearer bound* $p_c(G)$ for a graph G is the smallest $p > 0$ such that $(G, (p, p \ldots))$ does not satisfy Shearer's condition. Scott and Sokal [SS06] show that p_c is a point of singularity in the hard-core lattice gas model, where some thermodynamic quantities are known to exhibit non-trivial behavior. The hard-core lattice gas model deals with infinite graphs, among which the integer lattices are of special interest. Several techniques for computing p_c have been developed in statistical mechanics. A couple of examples are transfer matrix analysis and phenomenological renormalization [Gut87, Woo85, Tod99]. In fact, using the transfer matrix analysis, Todo [Tod99] has computed the amazingly precise estimate of $p_c = 0.11933888188(1)$ for the square lattice, $\mathbb{Z}^2$. The above methods are quite complex and require knowledge of involved tools from statistical mechanics. In [SS06], Scott and Sokal use the LLL to estimate p_c, proving a lower bound of $\approx .105468$ on $p_c(\mathbb{Z}^2)$. We use our results to improve on this LLL-based lower bound on $p_c(\mathbb{Z}^2)$.

1.3 Our Results

Shearer's condition (Theorem 2) is optimal because it uses all the global information regarding the structure of the dependency graph. In contrast the original LLL (Theorem 1) only uses minimalistic local structural information. Our main result is that we give a hierarchy of LLLs, increasingly complex, that use an increasing amount of local information in a non-trivial way, and on limit give Shearer's bound. Here are our results:

Clique Lovász Local Lemma (CLLL). First we present a special and very important member of our hierarchy that we call CLLL. The CLLL is a generalization of the LLL and it is especially useful when the neighborhoods of the nodes can be decomposed into a small number of cliques. This is the case in the variable framework, where the cliques correspond to the variables. The CLLL is described in Section 2.

We are aware of only one improvement, by Bissacot, Fernandez, Procacci and Scoppola [BFPS11], that lies between the conditions of Theorem 1 and Theorem 2, but this is not as user-friendly as the CLLL. We also show better bounds, which at the same time are achieved in a much more straightforward manner than using the version from [BFPS11]. Their proof uses relatively heavy tools from multivariate complex analysis. In contrast, we give a direct and elementary proof by modifying the original proof of Lovász [Spe77]. Our more general decomposition theorem follows the same lines.

Algorithmic Aspects. In a recent exciting development Moser [Mos09], Moser and Tardos (MT) [MT10], improving on a long sequence of earlier results that started with the work of Jozsef Beck [Bec91, Alo91, MR98, CS00, Sri08], show that the original LLL can also be made fully efficient. Pegden [Peg11] later proved the convergence of the MT algorithm with an explicit bound on running time for the LLL of [BFPS11]. In [KS11], Kolipaka and Szegedy have shown that the MT algorithm works efficiently up-to Shearer's bound. They also give a formula for the running time that we will exploit in this paper. Relying on [KS11] we show that in the case of CLLL a nice formula can be obtained similar to that in MT.

Improved Bounds for Coloring Problems. Using the CLLL in a straightforward way we improve bounds for the following problems.

1. **Acyclic Edge Coloring.** We prove that $a'(d) \leq 8.6d$. This improves on a previous bound of $9.62d$ by Ndreca, Procacci and Scoppola [NPS10] who in turn improved on the bound of $16d$ as shown by Molloy and Reed [MR98].
2. **Non-repetitive Vertex Coloring.** We show that $\pi(d) \leq 10.4d^2$. This improves on the previous bound of $16d^2$ by Grytczuk [Gry07].

A Family of Lovász Local Lemmas. We show that there is a family of LLLs, where each lemma in the family corresponds to a decomposition of the dependency graph G into vertex-induced subgraphs that cover all the edges of G. In general, the complexity of the constraints in our LLL hierarchy depends on the

size and structure of the parts of the decomposition. When the induced subgraphs are simply the edges of G, we get the original LLL (in fact a strictly better result) and if there is only one induced subgraph, then we get Shearer's condition, which is the best possible. This family is the result of the decomposition theorem described in Section 4.

Lower Bounds for $p_c(\mathbb{Z}^2)$. The standard statistical mechanics tools used to estimate $p_c(\mathbb{Z}^2)$ are complicated and require heavy computations. Our approach to lower bound $p_c(\mathbb{Z}^2)$ is to use better decompositions to get close to the symmetric Shearer bound. We apply the decomposition theorem for a very simple decomposition and prove that $p_c(\mathbb{Z}^2) \geq 0.1101$. This improves on the LLL-based lower bound of 0.1054 due to [SS06]. This is described in Section 4.1.

Note on Lopsidependency. In [ES91], Erdős and Spencer introduced the notion of *lopsidependency graph*. This a generalization of the notion of dependency graph, wherein the mutual independence between non-neighbors is replaced by the more general (and weaker) condition: for every $i \in [n]$ and a set S of its non-neighbors, we have $\text{Prob}(A_i \mid \bigcap_{j \in S} \overline{A_j}) \leq \text{Prob}(A_i)$. It can easily be shown that the statements of Theorem 1 and Theorem 2 are true for this case as well. We would like to note that this is also the case with our proofs of the CLLL and the decomposition theorem.

2 The Clique Lovász Local Lemma (CLLL)

Theorem 3 (Clique LLL). *Let $\{A_1, A_2, \ldots, A_n\}$ be a set of events with dependency graph G and let $\{K_1, K_2, \ldots, K_m\}$ be a set of cliques in G covering all the edges (not necessarily disjointly). If there exist a set of vectors $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ from $(0,1)^n$ such that following conditions are satisfied,*

- $\forall v \in [m] : \sum_{i \in K_v} x_{i,v} < 1$,
- $\forall i \in [n], \forall v$ *such that* $i \in K_v$:

$$\text{Prob}(A_i) \leq x_{i,v} \prod_{u \neq v : K_u \ni i} (1 - \sum_{j \in K_u \setminus \{i\}} x_{j,u}),$$

then:

1. $\text{Prob}(\bigcap_{i \in [n]} \overline{A}_i) \geq \prod_{v \in [m]} (1 - \sum_{i \in K_v} x_{i,v}) > 0$,
2. *In the variable framework, the running time of the algorithm of Moser and Tardos [MT10] is at most,*

$$\sum_{i \in [n]} \min_{v : K_v \ni i} \frac{x_{i,v}}{1 - \sum_{j \in K_v} x_{j,v}}.$$

Proof (of Statement 1). For ease of exposition, we will denote $\bigcap_{i \in S} \overline{A}_i$ with $\overline{A}_S$. We will use induction on $|S|$ to prove that,

$$\forall i \in [n], \forall K_v \ni i, \forall S \subseteq G \setminus K_v : \text{Prob}(A_i \mid \overline{A}_S) \leq x_{i,v}. \tag{2}$$

It is obviously true for $S = \emptyset$. Suppose it is true for all sets of size at most s. Let $|S| = s+1, X = S \cap N(i), Y = S \setminus N(i)$. Then

$$\text{Prob}(A_i \mid \overline{A}_S) = \frac{\text{Prob}(A_i \cap \overline{A}_X \cap \overline{A}_Y)}{\text{Prob}(\overline{A}_X \cap \overline{A}_Y)} \leq \frac{\text{Prob}(A_i)}{\text{Prob}(\overline{A}_X \mid \overline{A}_Y)}.$$

The inequality uses the fact, $\text{Prob}(A_i \mid \overline{A}_Y) \leq \text{Prob}(A_i)$, which is true in both the dependency and lopsidependency graphs (Section 1.3). For each $K_u \ni i$, let $X_u = K_u \cap X$. Since X contains only neighbors of i that are not in K_v and all edges incident on i are covered by some clique $K_u \ni i$, we have, that $X \subseteq \bigcup_{u \neq v : K_u \ni i} K_u$. Then using the chain rule

$$\text{Prob}(\overline{A}_X \mid \overline{A}_Y) \geq \prod_{u \neq v : K_u \ni i} \text{Prob}(\overline{A}_{X_u} \mid \overline{A}_{Y_u}) \text{ where } Y_u \subseteq S \setminus K_u.$$

Since X_u, Y_u are disjoint subsets of S, if $X_u \neq \emptyset$ then $|Y_u| < |S| = s+1$. Also, by definition $Y_u \subseteq G \setminus K_u$. Therefore using the induction hypothesis we can conclude that for every $j \in X_u$, we have $\text{Prob}(A_j \mid \overline{A}_{Y_u}) \leq x_{j,u}$. By applying the union bound for each X_u we have

$$\text{Prob}(\overline{A}_X \mid \overline{A}_Y) \geq \prod_{u \neq v : K_u \ni i} (1 - \sum_{j \in X_u} x_{i,u}) \geq \prod_{u \neq v : K_u \ni i} (1 - \sum_{j \in K_u \setminus \{i\}} x_{j,u})$$

which gives (2).

The result now follows from an easy of application of the chain rule and then the union bound. Let $S_1, S_2, \ldots, S_m$ be the sequence of subsets of vertices given by: $S_1 = K_1$ and $\forall 1 \leq v < m, S_{v+1} = K_{v+1} \setminus \cup_{\ell=1}^{v} S_\ell$. Also, let $S_{>v} = \cup_{\ell=v+1}^{m} S_\ell$. It is easy to see that $S_1 \cup S_2 \cup \ldots S_m = [n]$, therefore,

$$\text{Prob}(\overline{A}_{[n]}) = \prod_{v=1}^{m} \text{Prob}(\overline{A}_{S_v} \mid \overline{A}_{S_{>v}})$$

Clearly, $S_{>v} \subseteq G \setminus K_v$. Therefore using (2), we have $\text{Prob}(\overline{A}_i \mid \overline{A}_{S_{>v}}) \leq x_{i,v}$ for every node $i \in S_v$. Using the union bound here for the events, $(\overline{A}_i \mid \overline{A}_{S_{>v}})$, for all $i \in S_v$ gives the result.

Proof (of Statement 2). It can be easily shown that if the probabilities satisfy the conditions of the theorem, they also satisfy the Shearer condition for the graph. Now, [KS11] prove that when the probabilities satisfy Shearer's condition, the running time of the Moser and Tardos algorithm is at most $\sum_{i \in [n]} \frac{q_{\{i\}}(G,p)}{q_\emptyset(G,p)}$. Using Theorem 2 this can be written as,

$$\sum_{i \in [n]} \frac{q_{\{i\}}}{q_\emptyset} = \sum_{i \in [n]} \frac{\text{Prob}(B_i \mid \bigcap_{j \neq i} \overline{B}_j)}{1 - \text{Prob}(B_i \mid \bigcap_{j \neq i} \overline{B}_j)}$$

where $\{B_1, B_2, \ldots, B_n\}$ are the events of the extreme instance. Now let K_v be any clique that contains i. Then,

$$\text{Prob}(B_i \mid \bigcap_{j \neq i} \overline{B}_j) \leq \frac{\text{Prob}(B_i \mid \overline{B}_{[n] \setminus K_v})}{\text{Prob}(\overline{B}_{K_v \setminus \{i\}} \mid \overline{B}_{[n] \setminus K_v})}$$

Using (2) then gives $\text{Prob}(B_i \mid \bigcap_{j \neq i} \overline{B}_j) \leq \frac{x_{i,v}}{1-\sum_{j \in K_v \setminus \{i\}} x_{j,v}}$, giving the required result.

2.1 CLLL vs. Previous LLLs

First we should mention that CLLL always yields better bounds than Theorem 1, the original LLL. This can be immediately seen if we let the cliques in the decomposition be the individual edges. Let $(x_1, x_2, \ldots, x_n)$ be numbers that satisfy Theorem 1. For every edge $e = (i, j)$, set $x_{i,e} = x_i$ and $x_{j,e} = x_j$ when $x_i + x_j < 1$ and $x_{i,e} = 1 - x_j$ and $x_{j,e} = 1 - x_i$ when $x_i + x_j > 1$. If $x_i + x_j = 1$ setting $x_{i,e} = x_i - \epsilon, x_{j,e} = x_j$ for an arbitrarily small $\epsilon > 0$ (and then consider the bounds as ϵ goes to zero). In all the above cases the upper bound on $\text{Prob}(A_i)$ given by the CLLL is at least $\frac{x_i \prod_{j \in N(i)}(1-x_j)}{\max_{j \in N(i)}(1-x_j)}$, which is greater than the bound from the Theorem 1.

CLLL has the nice property that for graphs with maximum degree d we obtain the optimal $(d-1)^{d-1}/d^d$ bound, while Theorem 1 gives only $d^d/(d+1)^{d+1}$. It also gives the correct (union) bound when G is a clique, a natural extreme case, where Theorem 1 fails to work. CLLL gives better bounds than the LLL version of Bissacot, Fernandez, Procacci and Scoppola [BFPS11] for all examples we have studied, it has a simpler proof, and it is more natural to apply.

3 Applications of the Clique Lovász Local Lemma

The applications in this section will correspond to the variable framework described in Section 1.1. The events are determined by a set of independent discrete random variables $vbl = \{v_1, v_2, \ldots, v_m\}$. To apply the Clique LLL in this framework, we observe that each variable $v \in vbl$ corresponds to a clique K_v in the dependency graph formed by the nodes $\{i \mid vbl(i) \ni v\}$. This describes a canonical way to decompose the dependency graph in the variable framework into cliques $\{K_1, K_2, \ldots, K_m\}$.

3.1 Acyclic Edge Coloring

The main tool in the previous results ([AMR91], [MR98], [NPS10]) results is the asymmetric version of the LLL and hence we are able to improve upon the results in a straightforward way. Our proof is essentially the same as [AMR91] and [MR98] except we now use the Clique LLL.

Theorem 4. $a'(d) \leq 8.6d$.

Proof. Each edge of G is independently assigned a color from $\{1, 2, \ldots, k\}$ uniformly at random. For the application of the LLL, we identify the following types of *bad* events:

1. The edges in a path of length 2 are assigned the same color. We denote the set of all such events by $\mathcal{B}_1$. Also, if $A_1 \in \mathcal{B}_1, \text{Prob}(A_1) = \frac{1}{k}$.

2. An even length cycle C is properly 2-edge-colored. The set of all such events corresponding to a cycle of length 2ℓ ($\ell > 1$) is denoted by $\mathcal{B}_\ell$ and if $A_\ell \in \mathcal{B}_\ell, \text{Prob}(A_\ell) \leq \frac{1}{k^{2\ell-2}}$.

Clearly, an edge coloring is acyclic if and only if none of the above events occur. We will prove that if $k \geq 9d$, the conditions of theorem 3 can be satisfied and hence there is a positive probability that the edge coloring is acyclic. The random variables that determine an event are the colors of the corresponding edges. For each event $A_i \in \mathcal{B}_i$ and edge e that effects it we set $x_{A_i,e} = x_i = \frac{c}{(1+\epsilon)}\frac{1}{(2d-2)}$ if $i = 1$ otherwise if $i > 1$ we set $x_{A_i,e} = x_i = \frac{c}{(1+\epsilon)^i}\frac{1}{d^{2i-2}}$, while for some ϵ it will be determined later.

Now the number of cycles of length $2i$ that contain any given edge is at most d^{2i-2} and the number of length 2 paths that contain any given edge is at most $(2d-2)$. Therefore edge e is contained in at most d^{2i-2} events from $\mathcal{B}_i$ for $i > 1$ and $(2d-2)$ events from $\mathcal{B}_1$. To prove the theorem, it is enough to have

$$\forall e \in A_i : x_{A_i,e} \prod_{e' \in A_i \setminus \{e\}} (1 - \sum_{e' \in A} x_{A,e'}) \geq \text{Prob}(A_i),$$

that is,

$$x_i(1 - c\sum_{j=1}^{\infty}(1+\epsilon)^{-j})^{2i-1} \geq \text{Prob}(A_i).$$

From the above, it is enough to have $\epsilon > c$ and $\frac{k}{d} \geq \min\{(1+\epsilon)(1+\frac{1}{\epsilon-c})^{\frac{3}{2}}, \frac{2}{c}(1+\epsilon)(1+\frac{1}{\epsilon-c})\}$. Minimizing over ϵ, c, it is easy to verify that $k \geq 8.6d$ is enough to satisfy the condition.

3.2 Non-repetitive Vertex Coloring

We will be concerned with the upper bound on $\pi(d)$ (Section 1.1). Alon, Grytczuk, Hauszczak and Riordan [AGHR02] prove that $\pi(d)$ is $O(d^2)$ using the LLL. Grytczuk in [Gry07] shows that this can be improved to $16d^2$, which is basically the same random coloring as [AGHR02] with a more optimized application of the LLL. We use the same random coloring and replace the application of LLL with the Clique LLL.

Theorem 5. $\pi(d) < 10.4d^2$.

Proof. Suppose we assign colors from $\{1, 2, \ldots k\}$ to the vertices of G independently and uniformly at random. For every simple path P of length $2\ell - 1$ (with 2ℓ vertices) we say that a bad event occurs if the sequence of colors in the first half of P is the same as the second half. We denote the set of bad events corresponding to paths of length $2\ell - 1$ by $\mathcal{B}_\ell$. It is easy to see that probability of any event in $\mathcal{B}_\ell$ is $\frac{1}{k^\ell}$. Let $\epsilon > 0$ be a constant, which we will set later. For each $A_\ell \in \mathcal{B}_\ell$, a vertex v in the path corresponding to A_ℓ we set,

$$x_{\ell,v} = x_\ell = \frac{1}{(6d^2)^\ell}.$$

For every vertex v, the number of paths of length $2\ell - 1$ containing v is at most $\ell d^{2\ell}$. Therefore to prove the theorem it is enough to have,

$$x_{\ell,v} \prod_{u \in vbl_\ell \setminus \{v\}} (1 - \sum_{j : vbl(j) \ni u} x_{j,u}) \geq \text{Prob}(A_\ell),$$

that is,

$$x_\ell (1 - \sum_{j=1}^{\infty} j 6^{-j})^{2\ell-1} = \frac{1}{(6d^2)^\ell} (1 - \frac{6}{25})^{2\ell-1} \geq \text{Prob}(A_\ell) = \frac{1}{k^\ell}.$$

It is easy to verify that it is enough to have $k \geq 10.4d$.

4 Decomposition Theorem

In this section we will prove the decomposition theorem, a generalization of the Clique LLL. We have shown via the Clique LLL (Theorem 3) how to achieve sharper bounds by using more local information, namely that the neighborhood of a node can be covered by a small number of cliques. The decomposition theorem bridges the gap between Shearer's bound and the LLL by enabling the use of even more local information about a dependency graph. An example where this is useful is to estimate the Shearer bounds for multidimensional infinite grids. We will describe the application to 2-dimensional grid later in Section 4.1.

Definition 1 (Graph Decomposition). *Given an undirected graph G, a set of induced subgraphs $\{G_1, G_2, \ldots, G_T\}$ is called a* decomposition *if they cover all its edges. The G_is will be called the* parts *of the decomposition.*

Theorem 6 (Decomposition Theorem). *Let $\{A_1, A_2, \ldots, A_n\}$ be a set of events with dependency graph G. If $\{G_1, G_2, \ldots G_T\}$ is a decomposition of G and $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T\}$ are vectors in $(0,1)^n$ with the following properties:*

- $\forall j \in [T], \forall I \in \text{Indep}(G_j) : q_I(G_j, \boldsymbol{x}_j) > 0$
- $\forall i \in [n]$, $G_j \ni i$,

$$\text{Prob}(A_i) \leq x_{i,j} \prod_{\ell \neq j : G_\ell \ni i} \frac{q_\emptyset(G_\ell \setminus \{i\}, \boldsymbol{x}_\ell)}{q_\emptyset(G_\ell \setminus N^+(i), \boldsymbol{x}_\ell)}$$

then $\text{Prob}(\bigcap_{i=1}^n \overline{A_i}) \geq \prod_{j \in [T]} q_\emptyset(G_j, \boldsymbol{x}_j) > 0$.

Proof. For ease of exposition, we will denote $\bigcap_{i \in S} \overline{A_i}$ with $\overline{A}_S$. We will use induction on $|S|$ to prove that,

$$\forall i \in [n], G_j \ni i, S \subseteq G \setminus G_j : \text{Prob}(A_i \mid \overline{A}_S) \leq x_{i,j}. \tag{3}$$

Using (3) along with the chain rule (similar to Theorem 2) will then give the required result. Therefore we will now prove (3). It is obviously true for $S = \emptyset$,

now suppose it is true for all sets of size at most s. Let $|S| = s+1$. The result clearly holds if $S \cap N(i) = \emptyset$. Therefore we can assume without loss in generality that $S \cap N(i) \neq \emptyset$. Let $X = S \cap N(i), Y = S \setminus N(i)$. Then,

$$\text{Prob}(A_i \mid \overline{A}_S) = \frac{\text{Prob}(A_i \cap \overline{A}_X \cap \overline{A}_Y)}{\text{Prob}(\overline{A}_X \cap \overline{A}_Y)} \leq \frac{\text{Prob}(A_i)}{\text{Prob}(\overline{A}_X \mid \overline{A}_Y)}.$$

For each $\ell \neq j$ set $X_\ell = G_\ell \cap X$. Since X contains only neighbors of i that are not in G_j and all edges incident on i are covered by some graph in the decomposition, $X \subseteq \bigcup_{\ell \neq j : G_\ell \ni i} G_\ell$. Using the chain rule, $\text{Prob}(\overline{A}_X \mid \overline{A}_Y)$ is at least,

$$\prod_{\ell \neq j : G_\ell \ni i} \text{Prob}(\overline{A}_{X_\ell} \mid \overline{A}_{Y_\ell \cup Z_\ell}) = \prod_{\ell \neq j : G_\ell \ni i} \frac{\text{Prob}(\overline{A}_{X_\ell \cup Y_\ell} \mid \overline{A}_{Z_\ell})}{\text{Prob}(\overline{A}_{Y_\ell} \mid \overline{A}_{Z_\ell})}$$

where $Y_\ell \subseteq G_\ell \setminus N^+(i), Z_\ell \subseteq G \setminus G_\ell$. Since X_ℓ, Z_ℓ are disjoint subsets of S, if $X_\ell \neq \emptyset$, then $|Z_\ell| < |S| = s+1$. Also by definition, $Z_\ell \subseteq G \setminus G_\ell$. Therefore by the inductive hypothesis,

$$\forall i' \in G_\ell : \text{Prob}(A_{i'} \mid \overline{A}_{Z_\ell}) \leq x_{i',\ell}.$$

But we also know that $\mathbf{x}_\ell$ satisfies Shearer's condition for $G_\ell \setminus \{i\}$. Therefore, the events $(A_{i'} \mid \overline{A}_{Z_\ell}), i' \in G_\ell \setminus \{i\}$ satisfy the Shearer condition. We will then be done by the following lemma (the proof of which will appear in the full version).

Lemma 1. *Suppose $\{A_1, A_2 \ldots A_n\}$ is a set of events with dependency graph G and $\boldsymbol{x} \in (0,1)^n$ such that $\forall i \in [n] : \text{Prob}(A_i) \leq x_i$ and $(G, \boldsymbol{x})$ satisfy Shearer's condition. If H is an induced subgraph of G and $Y \subseteq V(H)$ then,*

$$\text{Prob}(\overline{A}_{[n] \setminus V(H)} \mid \overline{A}_Y) \geq \frac{q_\emptyset(G, \boldsymbol{x})}{q_\emptyset(H, \boldsymbol{x})}.$$

4.1 Lower Bounds for $p_c(\mathbb{Z}^2)$

Imagine the unit squares in $\mathbb{Z}^2$ to be colored in a black and white chess board pattern. We decompose this graph using the white unit squares as the parts of the decomposition. Clearly all the edges are covered by this decomposition and each node is present in exactly two parts.

Suppose g is a node in $\mathbb{Z}^2$ that is contained in the unit squares W_1, W_2. Let $x_{g,W_1} = x_{g,W_2} = x$. The Shearer polynomials are, $q_\emptyset(W_1, x) = q_\emptyset(W_2, x) = 1 - 4x + 2x^2$. Also, $q_\emptyset(W_1 \setminus \{g\}, x) = 1 - 3x + x^2$ and $q_\emptyset(W_1 \setminus N^+(g), x) = 1 - x$. Therefore to get a lower bound, it is enough to maximize,

$$x\frac{(1 - 3x + x^2)}{(1 - x)}, \text{ subject to } 1 - 4x + 2x^2 > 0, x \in (0,1).$$

Using Mathematica we get $p_c(\mathbb{Z}^2) > 2 - \frac{2}{2^{2/3}} > 0.1101$. With a slightly more complex decomposition we got an even better bound of 0.113. This can be improved by a series of increasingly more complex decompositions.

Acknowledgement. We thank Donald Knuth for the references to Theorem 1, for catching an important typo in the first write-up and for pointing out the importance of treating the lopsided case as well. We would also like to thank the anonymous referees for numerous detailed comments and suggestions.

References

[AG08] Alon, N., Grytczuk, J.: Breaking the rhythm on graphs. Discrete Mathematics 308(8), 1375–1380 (2008)

[AGHR02] Alon, N., Grytczuk, J., Haluszczak, M., Riordan, O.: Nonrepetitive colorings of graphs. Random Struct. Algorithms 21(3-4), 336–346 (2002)

[Alo91] Alon, N.: A parallel algorithmic version of the Local Lemma. In: FOCS, pp. 586–593 (1991)

[AMR91] Alon, N., McDiarmid, C., Reed, B.A.: Acyclic coloring of graphs. Random Struct. Algorithms 2(3), 277–288 (1991)

[ASZ01] Alon, N., Sudakov, B., Zaks, A.: Acyclic edge colorings of graphs. Journal of Graph Theory 37(3), 157–167 (2001)

[Bec91] Beck, J.: An algorithmic approach to the Lovász Local Lemma. i. Random Struct. Algorithms 2(4), 343–366 (1991)

[BFPS11] Bissacot, R., Fernndez, R., Procacci, A., Scoppola, B.: An improvement of the Lovász Local Lemma via cluster expansion. Combinatorics, Probability and Computing, FirstView, 1–11 (2011)

[BGK+07] Bresar, B., Grytczuk, J., Klavzar, S., Niwczyk, S., Peterin, I.: Nonrepetitive colorings of trees. Discrete Mathematics 307(2), 163–172 (2007)

[CS00] Czumaj, A., Scheideler, C.: Coloring non-uniform hypergraphs: a new algorithmic approach to the general Lovász Local Lemma. In: SODA, pp. 30–39 (2000)

[Cur05] Currie, J.D.: Pattern avoidance: themes and variations. Theor. Comput. Sci. 339, 7–18 (2005)

[DJKW12] Dujmovic, V., Joret, G., Kozik, J., Wood, D.R.: Nonrepetitive colouring via entropy compression. CoRR, abs/1112.5524 (2012)

[EL75] Erdös, P., Lovász, L.: Problems and results on 3-chromatic hypergraphs and some related questions. In: Hajnal, A., Rado, R., Sos, V.T. (eds.) Infinite and Finite Sets (to Paul Erdos on his 60th birthday), pp. 609–627 (1975)

[ES91] Erdös, P., Spencer, J.: Lopsided Lovász Local Lemma and latin transversals. Discrete Applied Mathematics 30(2-3), 151–154 (1991)

[Grü73] Grünbaum, B.: Acyclic colorings of planar graphs. Israel Journal of Mathematics 14, 390–408 (1973)

[Gry07] Grytczuk, J.: Nonrepetitive colorings of graphs: A survey. Int. J. Math. Mathematical Sciences 2007 (2007)

[Gry08] Grytczuk, J.: Thue type problems for graphs, points, and numbers. Discrete Mathematics 308(19), 4419–4429 (2008)

[Gut87] Guttmann, A.J.: Comment: Comment on 'the exact location of partition function zeros, a new method for statistical mechanics'. Journal of Physics A Mathematical General 20, 511–512 (1987)

[HSS10] Haeupler, B., Saha, B., Srinivasan, A.: New constructive aspects of the Lovasz Local Lemma. In: FOCS, pp. 397–406 (2010)

[KS11] Kolipaka, K.B.R., Szegedy, M.: Moser and Tardos meet Lovász. In: STOC, pp. 235–244 (2011)

[MNS07] Muthu, R., Narayanan, N., Subramanian, C.R.: Improved bounds on acyclic edge colouring. Discrete Mathematics 307(23), 3063–3069 (2007)
[Mos09] Moser, R.A.: A constructive proof of the Lovász Local Lemma. In: STOC, pp. 343–350 (2009)
[MR98] Molloy, M., Reed, B.A.: Further algorithmic aspects of the Local Lemma. In: STOC, pp. 524–529 (1998)
[MT10] Moser, R.A., Tardos, G.: A constructive proof of the general Lovász Local Lemma. J. ACM 57(2) (2010)
[NPS10] Ndreca, S., Procacci, A., Scoppola, B.: Improved bounds on coloring of graphs (2010)
[Peg11] Pegden, W.: An improvement of the Moser-Tardos algorithmic local lemma. CoRR, abs/1102.2853 (2011)
[She85] Shearer, J.B.: On a problem of Spencer. Combinatorica 5(3), 241–245 (1985)
[Spe77] Spencer, J.: Asymptotic lower bounds for Ramsey functions. Discrete Mathematics 20, 69–76 (1977)
[Sri08] Srinivasan, A.: Improved algorithmic versions of the Lovász Local Lemma. In: SODA, pp. 611–620 (2008)
[SS06] Scott, A.D., Sokal, A.D.: On dependency graphs and the lattice gas. Combinatorics, Probability & Computing 15(1-2), 253–279 (2006)
[Thu06] Thue, A.: Über unendliche Zeichenreihen. Norske Vid Selsk. Skr. I. Mat. Nat. Kl. Christian 7, 1–22 (1906)
[Tod99] Todo, S.: Transfer-matrix study of negative-fugacity singularity of hard-core lattice gas. International Journal of Modern Physics C 10, 517–529 (1999)
[Woo85] Wood, D.W.: The exact location of partition function zeros, a new method for statistical mechanics. Journal of Physics A: Mathematical and General 18(15), L917 (1985)

Finding Small Sparse Cuts by Random Walk

Tsz Chiu Kwok and Lap Chi Lau

The Chinese University of Hong Kong

Abstract. We study the problem of finding a small sparse cut in an undirected graph. Given an undirected graph $G = (V, E)$ and a parameter $k \leq |E|$, the small sparsest cut problem is to find a set $S \subseteq V$ with minimum conductance among all sets with volume at most k. Using ideas developed in local graph partitioning algorithms, we obtain the following bicriteria approximation algorithms for the small sparsest cut problem:

- If there is a set $U \subseteq V$ with conductance ϕ and $\text{vol}(U) \leq k$, then there is a polynomial time algorithm to find a set S with conductance $O(\sqrt{\phi/\epsilon})$ and $\text{vol}(S) \leq k^{1+\epsilon}$ for any $\epsilon > 1/k$.
- If there is a set $U \subseteq V$ with conductance ϕ and $\text{vol}(U) \leq k$, then there is a polynomial time algorithm to find a set S with conductance $O(\sqrt{\phi \log k/\epsilon})$ and $\text{vol}(S) \leq (1+\epsilon)k$ for any $\epsilon > 2\log k/k$.

These algorithms can be implemented locally using truncated random walk, with running time almost linear to k.

1 Introduction

For an undirected graph $G = (V, E)$, the conductance of a set $S \subseteq V$ is defined as $\phi(S) = |\delta(S)|/\text{vol}(S)$, where $\delta(S)$ is the set of edges with one endpoint in S and another endpoint in $V - S$, and $\text{vol}(S) = \sum_{v \in S} d(v)$ where $d(v)$ is the degree of v in G. Let $n = |V|$ and $m = |E|$. The conductance of G is defined as $\phi(G) = \min_{S:\text{vol}(S) \leq m} \phi(S)$. The conductance of a graph is an important parameter that is closely related to the expansion of a graph and the mixing time of a random walk [9]. Finding a set of small conductance, called a sparse cut, is a well-studied algorithmic problem that has applications in different areas. Several approximation algorithms are known for the sparsest cut problem. The spectral partitioning algorithm by Cheeger's inequality [7,1] finds a set of of conductance $\sqrt{\phi(G)}$ with volume at most m. The linear programming rounding algorithm by Leighton and Rao [11] finds a set of conductance $O(\phi(G)\log(n))$ with volume at most m. The semidefinite programming rounding algorithm by Arora, Rao and Vazirani [5] finds a set of conductance $O(\phi(G)\sqrt{\log(n)})$ with volume at most m.

Recently there has been much interest in studying the small sparsest cut problem, to determine $\phi_k(G) = \min_{S:\text{vol}(S) \leq k} \phi(S)$ for a given k, and to find a set of smallest conductance among all sets of volume at most k. This is also known as the expansion profile of the graph [13,16]. There are two main motivations for this problem. One is the small set expansion conjecture [15], which states that for

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 615–626, 2012.
© Springer-Verlag Berlin Heidelberg 2012

every constant $\epsilon > 0$ there exists a constant $\delta > 0$ such that it is NP-hard to distinguish whether $\phi_{\delta m}(G) \leq \epsilon$ or $\phi_{\delta m}(G) \geq 1 - \epsilon$. This conjecture is shown to be closely related to the unique games conjecture [15], and so it is of interest to understand what algorithmic techniques can be used to estimate $\phi_k(G)$. There are bicriteria approximation algorithms for this problem using semidefinite programming relaxations: Raghavendra, Steurer and Tetali [16] obtained an algorithm that finds a set S with $\mathrm{vol}(S) \leq O(k)$ and $\phi(S) \leq O(\sqrt{\phi_k(G)\log(m/k)})$, and Bansal et.al. [6] obtained an algorithm that finds a set S with $\mathrm{vol}(S) \leq (1+\epsilon)k$ and $\phi(S) \leq O(f(\epsilon)\ \phi_k(G)\sqrt{\log n \log(m/k)})$ for any $\epsilon > 0$ where $f(\epsilon)$ is a function depends only on ϵ.

There are also algorithms for finding a small sparse cut using the eigenvalues and eigenvectors of the Laplacian matrix: Arora, Barak and Steurer [4] gave a random walk based algorithm that returns a set of conductance $O(\sqrt{\lambda_{n^{100\epsilon}}/\epsilon})$ with size $O(n^{1-\epsilon})$ for $0 < \epsilon < 1$, where λ_i is the i-th smallest eigenvalue of the Laplacian matrix of the graph. Lee, Oveis Gharan and Trevisan [10] and Louis, Raghavendra, Tetali and Vempala [12] gave spectral algorithms that return a set of conductance $O(\sqrt{\lambda_{n/k}\log(n/k)})$ with size $O(k)$. We note that these results give sufficient conditions to find a small sparse cut efficiently, but they do not imply a bicriteria approximation algorithm for the small sparsest cut problem. Recently there is also a random walk based algorithm for finding a balanced separator with conductance $O(\sqrt{\phi})$ in nearly linear time [19] where ϕ is the conductance of the optimal balanced separator, but this result does not apply directly for the small sparsest cut problem.

Another motivation is the design of local graph partitioning algorithms in massive graphs. In some situations, we have a massive graph $G = (V, E)$ and a vertex $v \in V$, and we would like to identify a small set S with small conductance that contains v (if it exists). The graph may be too big that it is not feasible to read the whole graph and run some nontrivial approximation algorithms. So it would be desirable to have a local algorithm that only explores a small part of the graph, and outputs a set S with small conductance that contains v, and the running time of the algorithm depends only on $\mathrm{vol}(S)$ and $\mathrm{polylog}(n)$. All local graph partitioning algorithms are based on some random walk type processes. The efficiency of the algorithm is measured by the work/volume ratio, which is defined as the ratio of the running time and the volume of the output set. Spielman and Teng [18] proposed the first local graph partitioning algorithm using truncated random walk, that returns a set S' with $\phi(S') = O(\phi^{1/2}(S)\log^{3/2} n)$ if the initial vertex is a random vertex in S, and the work/volume ratio of the algorithm is $O(\phi^{-2}(S)\mathrm{polylog}(n))$. Anderson, Chung, Lang [2] used local pagerank vectors to find a set S' with $\phi(S') = O(\sqrt{\phi(S)\log k})$ and work/volume ratio $O(\phi^{-1}(S)\mathrm{polylog}(n))$, if the initial vertex is a random vertex in a set S with $\mathrm{vol}(S) = k$. Anderson and Peres [3] used the volume-biased evolving set process to obtain a local graph partitioning algorithm with work/volume ratio $O(\phi^{-1/2}\mathrm{polylog}(n))$ and a similar conductance guarantee as in [2]. Note that the running time of these algorithms would be sublinear if the volume of the output set is small, which is the case of interest in massive graphs.

1.1 Main Results

We show that the techniques developed in local graph partitioning algorithms [18,8] can be used to obtain bicriteria approximation algorithms for the small sparsest cut problem. We note that the algorithm in Theorem 1 is the same as the algorithm of Arora, Barak and Steurer [4], but we adapt the analysis in local graph partitioning algorithms to prove a tradeoff between the conductance guarantee and the volume of the output set.

Theorem 1. *Given an undirected graph $G = (V, E)$ and a parameter k, there is a polynomial time algorithm to do the following:*

1. *Find a set S with $\phi(S) = O(\sqrt{\phi_k(G)/\epsilon})$ and $vol(S) \leq k^{1+\epsilon}$ for any $\epsilon > 1/k$.*
2. *Find a set S with $\phi(S) = O(\sqrt{\phi_k(G) \log k/\epsilon})$ and $vol(S) \leq (1+\epsilon)k$ for any $\epsilon > 2 \log k/k$.*

For the small sparsest cut problem, when k is sublinear ($k = O(m^c)$ for $c < 1$), the performance guarantee of the bicriteria approximation algorithm in Theorem 1(2) is similar to that of Raghavendra, Steurer and Tetali [16]. Also, when k is sublinear, the conductance guarantee of Theorem 1(1) is independent of n, which matches the performance of spectral partitioning while having a bound on the volume of the output set. These show that random walk algorithms can also be used to give nontrivial bicriteria approximations for the small sparsest cut problem. We note that the result of Anderson and Peres [3] implies a similar statement to Theorem 1(2), with the same conductance guarantee and $\text{vol}(S) = O(k)$. The algorithms in Theorem 1 can also be implemented locally by using the truncated random walk algorithm.

Theorem 2. *For an undirected graph $G = (V, E)$ and a set $U \subseteq V$, given $\phi \geq \phi(U)$ and $k \geq vol(U)$, there exists an initial vertex such that the truncated random walk algorithm can find a set S with $\phi(S) \leq O(\sqrt{\phi/\epsilon})$ and $vol(S) \leq O(k^{1+\epsilon})$ for any $\epsilon > 2/k$. The runtime of the algorithm is $\tilde{O}(k^{1+2\epsilon}\phi^{-2})$.*

When k is sublinear, the conductance guarantee of Theorem 2 matches that of spectral partitioning, improving on the conductance guarantees in previous local graph partitioning algorithms. However, we note that our notion of a local graph partitioning algorithm is much weaker than previous work [18,2,3], as they proved that a random initial vertex u will work with a constant probability, while we only prove that there exists an initial vertex that will work and unable to prove the high probability statement.

In Section 4 we discuss a connection to the small set expansion conjecture.

Independent Work. Oveis Gharan and Trevisan [20] proved Theorem 1 independently. They also proved a stronger version of Theorem 2, with a faster running time ($\tilde{O}(k^{1+2\epsilon}\phi^{-1/2})$) and also the algorithm works for a random initial vertex in S with constant probability. We note that their result implies that our truncated random walk algorithm will also succeed with constant probability if we start from a random initial vertex in S.

1.2 Techniques

The techniques are from the work of Spielman and Teng [18] and Chung [8]. Our goal in Theorem 1(1) is equivalent to distinguish the following two cases: (a) there is a set S with $\mathrm{vol}(S) \leq k$ and $\phi(S) \leq \varphi$, or (b) the conductance of every set of volume at most ck is at least $\Omega(\sqrt{\varphi})$ for some $c > 1$. As in [18], we use the method of Lovász and Simonovits [14] that considers the total probability of the k edges with largest probability after t steps of random walk, call this number $C_t(k)$. In case (a), we use the idea of Chung [8] that uses the local eigenvector of S of the Laplacian matrix to show that there exists an initial vertex such that $C_t(k) \geq (1 - \frac{\varphi}{2})^t$. In case (b), we use a result of Lovász and Simonovits [14] to show that $C_t(k) \leq \frac{1}{c} + \sqrt{k}(1 - M\varphi)^t$ for a large enough constant M, no matter what is the initial vertex of the random walk. Hence, say when $c \geq k^{0.01}$, by setting $t = \Theta(\log k/\varphi)$, we expect that $C_t(k)$ is significantly greater than $1/c$ in case (a) but at most $1/c$ plus a negligible term in case (b), and so we can distinguish the two cases. To prove Theorem 2(1), we use the truncated random walk algorithm as in [18] to give a bound on the runtime. Theorem 1(2) is a corollary of Theorem 1(1).

2 Finding Small Sparse Cuts

The organization of this section is as follows. First we review some basics about random walk in undirected graphs. Then we present our algorithm in Theorem 1 and the proof outline, and then we present the analysis and complete the proof of Theorem 1.

2.1 Random Walk

In the following we assume $G = (V, E)$ is a simple unweighted undirected connected graph with $n = |V|$ vertices and $m = |E|$ edges. Our algorithms are based on random walk. Let p_0 be an initial probability distribution on vertices. Let A be the adjacency matrix of G, D be the diagonal degree matrix of G, and $W = \frac{1}{2}(I + D^{-1}A)$ be the lazy random walk matrix. The probability distribution after t steps of lazy random walk is defined as $p_t = p_0 W^t$. (For convenience, we use p_t to denote a row vector, while all other vectors by default are column vectors.) For a subset $S \subseteq V$, we use $p_t(S)$ to denote $\sum_{u \in S} p_t(u)$.

To analyze the probability distribution after t steps of lazy random walk, we use the method developed by Lovász and Simonovits [14] as in other local graph partitioning algorithms [18,2]. We view the graph as directed by replacing each undirected edge with two directed edges with opposite directions. Given a probability distribution p on vertices, each directed edge $e = uv$ is assigned probability $q(e) = p(u)/d_u$. Let $e_1, e_2, \ldots, e_{2m}$ be an ordering of the directed edges such that $q(e_1) \geq q(e_2) \geq \ldots \geq q(e_{2m})$. The curve introduced by Lovász and Simonovits $C : [0, 2m] \to [0, 1]$ is defined as follows: for integral x, $C(x) = \sum_{i=1}^{x} q(e_i)$; for fractional $x = \lfloor x \rfloor + r$, $C(x) = (1-r)C(\lfloor x \rfloor) + rC(\lceil x \rceil)$. Let C_t be

the curve when the underlying distribution is p_t. Let $v_1, v_2, \ldots, v_n$ be an ordering of the vertices such that $p_t(v_1)/d(v_1) \geq p_t(v_2)/d(v_2) \geq \ldots \geq p_t(v_n)/d(v_n)$. Then $C_t(\sum_{i=1}^{j} d(v_i)) = \sum_{i=1}^{j} p_t(v_i)$ for all $j \in [n]$. We call the points $x_j = \sum_{i=1}^{j} d(v_i)$ extreme points, and note that the curve is linear between two extreme points. We also call the sets $S_{t,j} = \{v_1, \ldots, v_j\}$ for $1 \leq j \leq n$ the level sets at time t.

The curve C_t is concave, and it approaches the straight line $x/(2m)$ when p_t approaches the stationary distribution. Lovász and Simonovits [14] analyzed the convergence rate of this curve to the straight line based on the conductances of the level sets.

Lemma 1 (Lovász-Simonovits [14]). *Let $x = x_j \leq m$ be an extreme point at time t and $S = S_{t,j}$ be the corresponding level set. If $\phi(S) \geq \varphi$, then $C_t(x) \leq \frac{1}{2}(C_{t-1}(x - \varphi x) + C_{t-1}(x + \varphi x))$.*

2.2 Algorithm

Our algorithm is simple and is the same as in Arora, Barak and Steurer [4]. For each vertex v, we use it as the initial vertex of the random walk, and compute the probability distributions p_t for $1 \leq t \leq O(n^2 \log n)$. Then we output the set of smallest conductance among all level sets $S_{t,j}$ (of all initial vertices) of volume at most ck, where in Theorem 1(1) we set $c = k^\epsilon$ and in Theorem 1(2) we set $c = 1 + \epsilon$. Clearly this is a polynomial time algorithm.

To analyze the performance of the algorithm, we give upper and lower bound on the curve based on the conductances. On one hand, we use Lemma 1 to prove that if all level sets of volume at most ck are of conductance at least ϕ_1, then the curve satisfies $C_t(x) \leq f_t(x) := \frac{x}{ck} + \sqrt{x}(1 - \frac{\phi_1^2}{8})^t$ for all $x \leq k$. Informally, this says that if ϕ_1 is large, then $C_t(k)$ is at most $1/c$ plus a negligible term when t is large enough. This statement holds regardless of the initial vertex of the random walk. On the other hand, if there exists a set S of volume at most k with conductance ϕ_2, then we use the idea of Chung [8] that uses the local eigenvector of S of the Laplacian matrix to show that there exists an initial vertex for which $C_t(k) \geq (1 - \frac{\phi_2}{2})^t$. Informally, this says that if ϕ_2 is small, then $C_t(k)$ is significantly larger than $1/c$ if c is large. Finally, by combining the upper and lower bound for $C_t(k)$ and choosing an appropriate t, we show that $\phi_1 \leq O(\sqrt{\phi_2})$ when $c = k^\epsilon$ and $\phi_1 \leq O(\sqrt{\phi_2 \log k})$ when $c = 1 + \epsilon$. Hence the algorithm can find a level set with the required conductance.

2.3 Upper Bound

We prove the upper bound using Lemma 1. We note that the following statement is true for any initial probability distribution, in particular when $p_0 = \chi_v$ for any v.

Theorem 3. *Suppose for all $t' \leq t$ and $i \in [n]$, we have $\phi(S_{t',i}) \geq \phi_1$ whenever $vol(S_{t',i}) \leq l \leq m$. Then the curve satisfies $C_t(x) \leq f_t(x) := \frac{x}{l} + \sqrt{x}(1 - \frac{\phi_1^2}{8})^t$ for all $x \leq 2m$.*

Proof. Let the extreme points x_i satisfy $0 = x_0 \le x_1 \le x_2 \le \ldots \le x_i \le l < x_{i+1}$. Note that C_t is linear between extreme points and between x_i and l, and f_t is concave. So we only need to show the inequality for extreme points and for $x \ge l$. When $x \ge l$, the inequality always hold as $f_t(x) \ge 1 \ge C_t(x)$ for any t. Now we would prove by induction. When $t = 0$ the inequality is trivial as $f_0(x) \ge 1 \ge C_0(x)$ for all $x \ge 1$. When $t > 0$ and x is an extreme point,

$$\begin{aligned} C_t(x) &\le \frac{1}{2}(C_{t-1}(x-\phi_1 x) + C_{t-1}(x+\phi_1 x)) \quad \text{(by Lemma 1)} \\ &\le \frac{1}{2}(f_{t-1}(x-\phi_1 x) + f_{t-1}(x+\phi_1 x)) \quad \text{(by induction)} \\ &= \frac{x}{l} + \frac{1}{2}\sqrt{x}(1-\frac{\phi_1^2}{8})^{t-1}(\sqrt{1-\phi_1} + \sqrt{1+\phi_1}) \\ &\le \frac{x}{l} + \sqrt{x}(1-\frac{\phi_1^2}{8})^t, \end{aligned}$$

where the last inequality follows from Taylor expansions of $\sqrt{1-\phi_1}$ and $\sqrt{1+\phi_1}$.

2.4 Lower Bound

The idea is to use the local eigenvector of S of the normalized Laplacian matrix to show that there is an initial distribution such that $p_t(S) \ge (1-\frac{\phi_2}{2})^t$.

Theorem 4. *Assume $S \subseteq V$ where $vol(S) \le m$ and $\phi(S) \le \phi_2$. Then there exists a vertex v such that if $p_0 = \chi_v$, then $p_t(S) \ge (1-\frac{\phi_2}{2})^t$.*

Proof. Let $\mathcal{L} = D^{-\frac{1}{2}}LD^{-\frac{1}{2}}$ be the normalized Laplacian matrix, where $L = D - A$ is the Laplacian matrix of the graph. For any matrix M with rows and columns indexed by V, let M_S be the $|S| \times |S|$ submatrix of M with rows and columns indexed by the vertices in S. Consider the smallest eigenvalue λ_S of $\mathcal{L}_S$ and its corresponding eigenvector v_S. Let χ_S be the characteristic vector of S. We have

$$(D_S^{\frac{1}{2}}\mathbf{1})^T\mathcal{L}_S(D_S^{\frac{1}{2}}\mathbf{1}) = \mathbf{1}^T L_S \mathbf{1} = \sum_{e=uv\in E}(\chi_S(u)-\chi_S(v))^2 = |\delta(S)|.$$

So, by the Courant-Fischer theorem,

$$\lambda_S \le \frac{(D_S^{\frac{1}{2}}\mathbf{1})^T\mathcal{L}_S(D_S^{\frac{1}{2}}\mathbf{1})}{\|D_S^{\frac{1}{2}}\mathbf{1}\|_2^2} = \frac{|\delta(S)|}{\mathrm{vol}(S)} \le \phi_2.$$

We assume without loss of generality that S is a connected subgraph. Then, by the Perron-Frobenius theorem, the eigenvector v_S can be assumed to be positive, and we can rescale v_S such that $D_S^{\frac{1}{2}}v_S$ is a probability distribution. Let $p_{t,S}$ denote the restriction of p_t on S. We set the initial distribution p_0 such that

$p_{0,S} = (D_S^{\frac{1}{2}} v_S)^T$, and $p_{0,V-S} = 0$. We would show that $p_{t,S} \geq (1 - \frac{\lambda_S}{2})^t p_{0,S}$ by induction. Clearly the statement is true when $t = 0$. For $t > 0$, we have

$$\begin{aligned}
p_{t,S} &\geq p_{t-1,S} W_S \\
&= p_{t-1,S} \cdot \frac{(I_S + D_S^{-1} A_S)}{2} \\
&\geq (1 - \frac{\lambda_S}{2})^{t-1} v_S^T D_S^{\frac{1}{2}} \frac{(I_S + D_S^{-1} A_S)}{2} \quad \text{(by induction)} \\
&= (1 - \frac{\lambda_S}{2})^{t-1} v_S^T (I - \frac{\mathcal{L}_S}{2}) D_S^{\frac{1}{2}} \\
&= (1 - \frac{\lambda_S}{2})^t v_S^T D_S^{\frac{1}{2}} \\
&= (1 - \frac{\lambda_S}{2})^t p_{0,S}.
\end{aligned}$$

Therefore,

$$p_t(S) = p_{t,S}(S) \geq (1 - \frac{\lambda_S}{2})^t p_{0,S}(S) \geq (1 - \frac{\phi_2}{2})^t.$$

Since random walk is linear and v_S is a convex combination of χ_v where $v \in S$, there exists a vertex $v \in S$ such that if $p_0 = \chi_v$, then $p_t(S) \geq (1 - \frac{\phi_2}{2})^t$.

2.5 Proof of Theorem 1

We combine the upper bound and the lower bound to prove Theorem 1. We note that Theorem 1 is trivial if $\phi_k(G) \geq \epsilon$, and so we assume $\phi_k(G) < \epsilon$. We also assume $\epsilon \leq 0.01$, as otherwise we reset $\epsilon = 0.01$ and lose only a constant factor.

The algorithm is simple. Set $T = \epsilon k^2 \log k/4$. For each vertex u, set $p_0 = \chi_u$ and compute $S_{t,i}$ for all $t \leq T$ and $i \in [n]$. Denote these sets by $S_{t,i,u}$ to specify the starting vertex u. Output a set $S = S_{t,i,u}$ that achieves the minimum in $\min_{\text{vol}(S_{t,i,u}) \leq k^{1+\epsilon}} \phi(S_{t,i,u})$. Clearly the algorithm runs in polynomial time.

We claim that $\phi(S) \leq 4\sqrt{\phi_k(G)/\epsilon}$. Suppose to the contrary that the algorithm does not return such a set. Consider $t = \frac{\epsilon \log k}{2\phi_k(G)}$; note that $t \leq T$ as $\phi_k(G) \geq 1/k^2$ for a simple unweighted graph. Applying Theorem 3 with $l = k^{1+\epsilon}$, for any starting vertex u, we have

$$\begin{aligned}
C_t(k) &\leq \frac{k}{k^{1+\epsilon}} + \sqrt{k}(1 - 2\frac{\phi_k(G)}{\epsilon})^t \\
&\leq k^{-\epsilon} + \sqrt{k} \exp(-2\frac{\phi_k(G)}{\epsilon} \frac{\epsilon \log k}{2\phi_k(G)}) \\
&= k^{-\epsilon} + \sqrt{k} \exp(-\log k) \\
&= k^{-\epsilon} + k^{-\frac{1}{2}}.
\end{aligned}$$

On the other hand, suppose S^* is a set with $\text{vol}(S^*) \leq k$ and $\phi(S^*) = \phi_k(G)$. Then Theorem 4 says that there exists a starting vertex $u^* \in S^*$ such that

$$p_t(S^*) \geq (1 - \frac{\phi_k(G)}{2})^t$$

$$\begin{aligned}
&\geq \exp(-\phi_k(G)t) \quad (\text{for } \phi_k(G) < 0.01)\\
&= \exp(-\frac{1}{2}\epsilon \log k)\\
&= k^{-\frac{\epsilon}{2}}\\
&> k^{-\epsilon} + k^{-\frac{1}{2}} \quad (\text{for } k \geq \frac{1}{\epsilon} \text{ and } \epsilon \leq 0.01)
\end{aligned}$$

This is contradicting since $C_t(k) \geq p_t(S^*)$ for that starting vertex, completing the proof of Theorem 1(1).

Now we obtain Theorem 1(2) as a corollary of Theorem 1(1). Set $\epsilon' = \frac{\epsilon}{2 \log k}$. Then $k^{1+\epsilon'} \leq (1+\epsilon)k$. By using Theorem 1(1) with ϵ', we have Theorem 1(2).

3 Local Graph Partitioning

To implement the algorithm locally, we use truncated random walk as in [18]. Let $q_0 = \chi_v$. For each $t \geq 0$, we define $\tilde{p}_t$ by setting $\tilde{p}_t(v) = 0$ if $q_t(v) < \epsilon d(v)$ and setting $\tilde{p}_t(v) = q_t(v)$ if $q_t(v) \geq \epsilon d(v)$, and we define $q_{t+1} = \tilde{p}_t W$. Then, we just use $\tilde{p}_t$ to replace p_t in the algorithm in Section 2. To prove that the truncated random walk algorithm works, we first show that $\tilde{p}_t$ is a good approximation of p_t and can be computed locally. Then we show that the curve defined by $\tilde{p}_t$ satisfies the upper bound in Theorem 3, and it almost satisfies the lower bound in Theorem 4. Finally we combine the upper bound and the lower bound to prove Theorem 2.

3.1 Computing Truncated Distributions

Lemma 2. *There is an algorithm that compute $\tilde{p}_t$ such that $\tilde{p}_t \leq p_t \leq \tilde{p}_t(v) + \epsilon t d$ for every $0 \leq t \leq T$, with time complexity $O(T/\epsilon)$, where d is the degree vector.*

Proof. First we prove the approximation guarantee. By induction, we have the upper bound

$$\tilde{p}_t \leq q_t = \tilde{p}_{t-1} W \leq p_{t-1} W = p_t.$$

Also, by induction, we have the lower bound

$$p_t = p_{t-1} W \leq (\tilde{p}_{t-1} + \epsilon(t-1)d)W = q_t + \epsilon(t-1)d \leq \tilde{p}_t + \epsilon t d.$$

Next we bound the computation time. Let S_t be the support of $\tilde{p}_t$. In order to compute q_{t+1} from $\tilde{p}_t$, we need to update each vertex $v \in S_t$ and its neighbors. Using a perfect hash function, the neighbors of a vertex v can be updated in $O(d(v))$ steps, and thus q_{t+1} and $\tilde{p}_{t+1}$ can be computed in $O(\text{vol}(S_t))$ steps. Since each vertex $v \in S_t$ satisfies $\tilde{p}_t \geq \epsilon d(v)$, we have $\text{vol}(S_t) = \sum_{v \in S_t} d(v) \leq p_t(S_t)/\epsilon \leq 1/\epsilon$, and this completes the proof.

3.2 Approximate Upper Bound

We use the truncated probability distributions to define the curve $\tilde{C}_t$. Note that $\tilde{p}_t$ may not be a probability distribution and $\tilde{C}_t(2m)$ may be less than one. And we define the level sets $\tilde{S}_{t,i} = \{v_1, v_2, \ldots, v_i\}$ when we order the vertices such that $\tilde{p}_t(v_1)/d(v_1) \geq \tilde{p}_t(v_2)/d(v_2) \geq \ldots \geq \tilde{p}_t(v_n)/d(v_n)$. We show that $\tilde{C}_t$ would satisfy the same upper bound as in Theorem 3.

Lemma 3. *Suppose for all $t \leq T$ and $i \in [n]$, we have $\phi(\tilde{S}_{t,i}) \geq \phi_1$ whenever $vol(\tilde{S}_{t,i}) \leq l \leq m$. Then $\tilde{C}_t(x) \leq f_t(x) := \frac{x}{l} + \sqrt{x}(1 - \frac{\phi_1^2}{8})^t$ for all $x \leq 2m$.*

Proof. Let $\tilde{x}_i = \sum_{v \in \tilde{S}_{t,i}} d(v)$ be the extreme points defined by $\tilde{p}_{t'}$. By the same proof as in Theorem 3, it suffices to prove that Lemma 1 still holds after replacing p_t by $\tilde{p}_t$. It means that we need to show if $x = \tilde{x}_i \leq m$ is an extreme point (at time t), $S = \tilde{S}_{t,j}$ is the corresponding set of vertices and $\mathrm{vol}(S) \geq \phi$, then $\tilde{C}_t(x) \leq \frac{1}{2}(\tilde{C}_{t-1}(x - \phi x) + \tilde{C}_{t-1}(x + \phi x))$. This is true since the curve defined by $q_t = \tilde{p}_{t-1}W$ is less than $\frac{1}{2}(\tilde{C}_{t-1}(x - \phi x) + \tilde{C}_{t-1}(x + \phi x))$ by Lemma 1, and $\tilde{p}_t \leq q_t$.

3.3 Proof of Theorem 2

Suppose U is a subset of vertices with $\mathrm{vol}(U) \leq k$ and $\phi(U) \leq \varphi$, where $\frac{1}{\epsilon} \leq k \leq m$. We would prove that given k and φ and an initial vertex u in U with $p_t(U) \geq \frac{1}{c}(1 - \frac{\phi}{2})^t$ for a constant $c > 1$, the truncated random walk algorithm will output a set S with $\mathrm{vol}(S) \leq O(k^{1+\epsilon})$ and $\phi(S) \leq 8\sqrt{\varphi/\epsilon}$. The running time of the algorithm is $O(\epsilon^2 k^{1+2\epsilon} \log^3 k/\varphi^2)$.

For concreteness we set $c = 4$ in the following calculations. Set $T = \frac{\epsilon \log k}{2\varphi}$ and $\epsilon' = \frac{k^{-1-\epsilon}}{20T}$. Applying Lemma 2 with T and ϵ', we can compute all $\tilde{p}_t$ and thus $\tilde{S}_{t,i}$ for all $t \leq T$ and $i \in [5k^{1+\epsilon}]$ in $O(T \log k/\epsilon') = O(\epsilon^2 k^{1+\epsilon} \log^3 k/\varphi^2)$ steps (with an additional $\log k$ factor for sorting). By Lemma 2, the starting vertex u will give $\tilde{p}_T(U) \geq \frac{1}{4}(1 - \frac{\varphi}{2})^T - \epsilon' T \mathrm{vol}(U)$. We claim that one of the set $S = S_{t,i}$ must satisfy $\mathrm{vol}(S) \leq 5k^{1+\epsilon}$ and $\phi(S) \leq 8\sqrt{\varphi/\epsilon}$. Otherwise, setting $\phi_1 \geq 8\sqrt{\varphi/\epsilon}$, we have

$$\begin{aligned}
\tilde{p}_T(U) &\geq \frac{1}{4}(1 - \frac{\varphi}{2})^T - \epsilon' T \mathrm{vol}(U) \\
&\geq \frac{1}{4}\exp(-\varphi T) - \frac{k^{-\epsilon}}{20} \quad (\text{for } \phi < 0.01) \\
&= \frac{k^{-\frac{\epsilon}{2}}}{4} - \frac{k^{-\epsilon}}{20} \\
&> \frac{k^{-\epsilon}}{5} + k^{-\frac{1}{2}} \quad (\text{using } k^{-\frac{\epsilon}{2}} > k^{-\epsilon} + 4k^{-\frac{1}{2}} \text{ for } k \geq \frac{1}{\epsilon} \text{ and } \epsilon \leq 0.01) \\
&\geq \frac{k}{5k^{1+\epsilon}} + \sqrt{k}(1 - \frac{\phi_1^2}{8})^T \\
&\geq \tilde{C}_T(k),
\end{aligned}$$

which is a contradiction, completing the proof of Theorem 2.

4 Concluding Remarks

We presented a bicriteria approximation algorithm for the small sparsest cut problem with conductance guarantee independent of n, but the volume of the output set is $k^{1+\epsilon}$. We note that if one can also guarantee that the volume of the output set is at most Mk for an absolute constant M, then one can disprove the small set expansion conjecture, which states that for any constant ϵ there exists a constant δ such that distinguishing $\phi_{\delta m}(G) < \epsilon$ and $\phi_{\delta m}(G) > 1 - \epsilon$ is NP-hard. This can be viewed as an evidence that our analysis is almost tight, or an evidence that the small set expansion problem is not NP-hard. We note that this is also observed by Raghavendra, Stuerer and Tulsiani [17].

More formally, suppose there is a polynomial time algorithm with the following guarantee: given G with $\phi_k(G)$, always output a set S with $\phi(S) = f(\phi_k(G))$ and $\mathrm{vol}(S) = Mk$ where $f(x)$ is a function that tends to zero when x tends to zero (e.g. $f(x) = x^{1/100}$) and M is an absolute constant. Then we claim that there is a (small) constant ϵ such that whenever $\phi_k(G) < \epsilon$ there is a polynomial time algorithm to return a set S with $\phi(S) < 1 - \epsilon$ and $\mathrm{vol}(S) \leq k$.

We assume that G is a d-regular graph, as in [15] where the small set expansion conjecture was formulated. Suppose there is a subset U with $|U| = k$ and $\phi(U) < \epsilon$. First we use the algorithm to obtain a set S with $\phi(S) \leq f(\epsilon)$ and assume $|S| = Mk$ (instead of $|S| \leq Mk$). Next we show that a random subset $S' \subseteq S$ of size exactly k will have $\phi(S') < 1 - \epsilon$ with a constant probability for a small enough ϵ. Let $E(S)$ be the set of edges with both endpoints in S. Each edge in $E(S)$ has probability $2(\frac{1}{M})(1 - \frac{1}{M})$ to be in $\delta(S')$. So, the expected value of

$$|\delta(S')| \leq |\delta(S)| + 2(\frac{1}{M})(1 - \frac{1}{M})|E(S)|.$$

By construction $\mathrm{vol}(S') = kd$, and so the expected value of

$$\phi(S') \leq \frac{|\delta(S)|}{kd} + \frac{2(\frac{1}{M})(1 - \frac{1}{M})|E(S)|}{kd}.$$

Note that $|E(S)| \leq Mkd/2$ and $|\delta(S)|/kd = M\phi(S) \leq Mf(\epsilon)$, so the expected value of

$$\phi(S') \leq Mf(\epsilon) + 1 - \frac{1}{M}.$$

For a small enough ϵ depending only on M, the expected value of $\phi(S') \leq 1 - 10\epsilon$. Therefore, with a constant probability, we have $\phi(S') < 1 - \epsilon$. This argument can be derandomized using standard techniques.

We show that random walk can be used to obtain nontrivial bicriteria approximation algorithms for the small sparsest cut problem. We do not know of an example showing that our analysis is tight. It would be interesting to find examples showing the limitations of random walk algorithms (e.g. showing that they fail to disprove the small set expansion conjecture).

References

1. Alon, N., Milman, V.: Isoperimetric inequalities for graphs, and superconcentrators. Journal of Combinatorial Theory, Series B 38(1), 73–88 (1985)
2. Anderson, R., Chung, F.R.K., Lang, K.J.: Local graph partitioning using PageRank vectors. In: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 475–486 (2006)
3. Anderson, R., Peres, Y.: Finding sparse cuts locally using evolving sets. In: Proceedings of the 41st Annual ACM Symposium on Theory of Computing (STOC), pp. 235–244 (2009)
4. Arora, S., Barak, B., Steurer, D.: Subexponential algorithms for unique games and related problems. In: Proceedings of the 51st Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 563–572 (2010)
5. Arora, S., Rao, S., Vazirani, U.: Expander flows, geometric embeddings and graph partitioning. In: Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC), pp. 222–231 (2004)
6. Bansal, N., Feige, U., Krauthgamer, R., Makarychev, K., Nagarajan, V., Naor, J., Schwartz, R.: Min-max graph partitioning and small set expansion. In: Proceedings of the 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 17–26 (2011)
7. Cheeger, J.: A lower bound for the smallest eigenvalue of the Laplacian. In: Problems in Analysis, pp. 195–199. Princeton University Press (1970)
8. Chung, F.: A Local Graph Partitioning Algorithm Using Heat Kernel Pagerank. In: Avrachenkov, K., Donato, D., Litvak, N. (eds.) WAW 2009. LNCS, vol. 5427, pp. 62–75. Springer, Heidelberg (2009)
9. Horry, S., Linial, N., Wigderson, A.: Expander graphs and their applications. Bulletin of the American Mathematical Society 43(4), 439–561 (2006)
10. Lee, J.R., Oveis Gharan, S., Trevisan, L.: Multi-way spectral partitioning and higher-order Cheeger inequalities. In: Proceedings of the 44th Annual Symposium on Theory of Computing (STOC), pp. 1117–1130 (2012)
11. Leighton, F.T., Rao, S.: Multicommodity max-flow min-cut theorem and their use in designing approximation algorithms. Journal of the ACM 46(6), 787–832 (1999)
12. Louis, A., Raghavendra, P., Tetali, P., Vempala, S.: Many sparse cuts via higher eigenvalues. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC), pp. 1131–1140 (2012)
13. Lovász, L., Kannan, R.: Faster mixing via average conductance. In: Proceedings of the 31st Annual ACM Symposium on Theory of Computing (STOC), pp. 282–287 (1999)
14. Lovász, L., Simonovits, M.: The mixing time of Markov chains, an isoperimetric inequality, and computing the volume. In: Proceedings of the 31st Annual IEEE Symposium on Foundations of Computer Science (FOCS), pp. 346–354 (1990)
15. Raghavendra, P., Steurer, D.: Graph expansion and the unique games conjecture. In: Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC), pp. 755–764 (2010)
16. Raghavendra, P., Steurer, D., Tetali, P.: Approximations for the isoperimetric and spectral profile of graphs and related parameters. In: Proceedings of the 42nd Annual ACM Symposium on Theory of Computing (STOC), pp. 631–640 (2010)
17. Raghavendra, P., Steurer, D., Tulsiani, M.: Reductions between expansion problems. In: Proceedings of the 27th Annual IEEE Conference on Computational Complexity, CCC (2012)

18. Spielman, D.A., Teng, S.-H.: A local clustering algorithm for massive graphs and its applications to nearly-linear time graph partitioning. CoRR, abs/0809.3232 (2008)
19. Orecchia, L., Sachdeva, S., Vishnoi, N.K.: Approximating the exponential, the Lanczos method, and an $\tilde{O}(m)$-time spectral algorithm for balanced separator. In: Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC), pp. 1141–1160 (2012)
20. Oveis Gharan, S., Trevisan, L.: Approximating the expansion profile and almost optimal local graph clustering. CoRR, abs/1204.2021 (2012)

On Deterministic Sketching and Streaming for Sparse Recovery and Norm Estimation

Jelani Nelson[1,⋆], Huy L. Nguyễn[1,⋆⋆], and David P. Woodruff[2]

[1] Princeton University, USA
{minilek,hlnguyen}@princeton.edu
[2] IBM Almaden Research Center, San Jose, USA
dpwoodru@us.ibm.com

Abstract. We study classic streaming and sparse recovery problems using *deterministic* linear sketches, including ℓ_1/ℓ_1 and ℓ_∞/ℓ_1 sparse recovery problems, norm estimation, and approximate inner product. We focus on devising a fixed matrix $A \in \mathbb{R}^{m\times n}$ and a deterministic recovery/estimation procedure which work for all possible input vectors simultaneously. We contribute several improved bounds for these problems.

- A proof that ℓ_∞/ℓ_1 sparse recovery and inner product estimation are equivalent, and that incoherent matrices can be used to solve both problems. Our upper bound for the number of measurements is $m = O(\varepsilon^{-2}\min\{\log n, (\log n/\log(1/\varepsilon))^2\})$. We can also obtain fast sketching and recovery algorithms by making use of the Fast Johnson-Lindenstrauss transform. Both our running times and number of measurements improve upon previous work. We can also obtain better error guarantees than previous work in terms of a smaller tail of the input vector.
- A new lower bound for the number of linear measurements required to solve ℓ_1/ℓ_1 sparse recovery. We show $\Omega(k/\varepsilon^2 + k\log(n/k)/\varepsilon)$ measurements are required to recover an x' with $\|x - x'\|_1 \leq (1+\varepsilon)\|x_{tail(k)}\|_1$, where $x_{tail(k)}$ is x projected onto all but its largest k coordinates in magnitude.
- A tight bound of $m = \Theta(\varepsilon^{-2}\log(\varepsilon^2 n))$ on the number of measurements required to solve deterministic norm estimation, i.e., to recover $\|x\|_2 \pm \varepsilon\|x\|_1$.

For all the problems we study, tight bounds are already known for the randomized complexity from previous work, except in the case of ℓ_1/ℓ_1 sparse recovery, where a nearly tight bound is known. Our work thus aims to study the deterministic complexities of these problems.

1 Introduction

In this work we provide new results for the point query problem as well as several other related problems: approximate inner product, ℓ_1/ℓ_1 sparse recovery,

⋆ Supported by NSF grant CCF-0832797.
⋆⋆ Supported in part by NSF grant CCF-0832797 and a Gordon Wu fellowship.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 627–638, 2012.
© Springer-Verlag Berlin Heidelberg 2012

and deterministic norm estimation. For many of these problems efficient randomized sketching and streaming algorithms exist, and thus we are interested in understanding the *deterministic* complexities of these problems.

1.1 Applications

Here we give a motivating application of the point query problem; for a formal definition of the problem, see below. Consider k servers $S^1, \ldots, S^k$, each holding a database $D^1, \ldots, D^k$, respectively. The servers want to compute statistics of the union D of the k databases. For instance, the servers may want to know the frequency of a record or attribute-pair in D. It may be too expensive for the servers to communicate their individual databases to a centralized server, or to compute the frequency exactly. Hence, the servers wish to communicate a short summary or "sketch" of their databases to a centralized server, who can then combine the sketches to answer frequency queries about D.

We model the databases as vectors $x^i \in \mathbb{R}^n$. To compute a sketch of x^i, we compute Ax^i for some $A \in \mathbb{R}^{m \times n}$. Importantly, $m \ll n$, and so Ax^i is much easier to communicate than x^i. The servers compute $Ax^1, \ldots, Ax^k$, respectively, and transmit these to a centralized server. Since A is a linear map, the centralized server can compute Ax for $x = c_1x^1 + \ldots c_kx^k$ for any real numbers $c_1, \ldots, c_k$. Notice that the c_i are allowed to be both positive and negative, which is crucial for estimating the frequency of record or attribute-pairs in the difference of two datasets, which allows for tracking which items have experienced a sudden growth or decline in frequency. This is useful in network anomaly detection [11,17,24,32,37], and also for transactional data [16]. It is also useful for maintaining the set of frequent items over a changing database relation [16].

Associated with A is an output algorithm Out which given Ax, outputs a vector x' for which $\|x' - x\|_\infty \leq \varepsilon \|x_{tail(k)}\|_1$ for some number k, where $x_{tail(k)}$ denotes the vector x with the top k entries in magnitude replaced with 0. Thus x' approximates x well on every coordinate. We call the pair (A, Out) a solution to the point query problem. Given such a matrix A and an output algorithm Out, the centralized server can obtain an approximation to the value of every entry in x, which depending on the application, could be the frequency of an attribute-pair. It can also, e.g., extract the maximum frequencies of x, which are useful for obtaining the most frequent items. The centralized server obtains an entire histogram of values of coordinates in x, which is a useful low-memory representation of x. Notice that the communication is mk words, as opposed to nk if the servers were to transmit $x^1, \ldots, x^n$.

Our correctness guarantees hold for all input vectors simultaneously using one fixed (A, Out) pair, and thus it is stronger and should be contrasted with the guarantee that the algorithm succeeds given Ax with high probability for some fixed input x. For example, for the point query problem, the latter guarantee is achieved by the CountMin sketch [15] or CountSketch [13]. One of the reasons the randomized guarantee is less useful is because of *adaptive* queries. That is, suppose the centralized server computes x' and transmits information about x' to $S^1, \ldots, S^k$. Since x' could depend on A, if the servers were to then use the

same matrix A to compute sketches $Ay^1, \ldots, Ay^k$ for databases $y^1, \ldots, y^k$ which depend on x', then A need not succeed, since it is not guaranteed to be correct with high probability for inputs y^i which depend on A.

1.2 Notation and Problem Definitions

Throughout this work $[n]$ denotes $\{1, \ldots, n\}$. For q a prime power, $\mathbb{F}_q$ denotes the finite field of size q. For $x \in \mathbb{R}^n$ and $S \subseteq [n]$, x_S denotes the vector with $(x_S)_i = x_i$ for $i \in S$, and $(x_S)_i = 0$ for $i \notin S$. The notation x_{-i} is shorthand for $x_{[n]\setminus\{i\}}$. For a matrix $A \in \mathbb{R}^{m \times n}$ and integer $i \in [n]$, A_i denotes the ith column of A. For matrices A and vectors x, A^T and x^T denote their transposes. For $x \in \mathbb{R}^n$ and integer $k \leq n$, we let $head(x, k) \subseteq [n]$ denote the set of k largest coordinates in x in absolute value, and $tail(x, k) = [n] \setminus head(x, k)$. We often use $x_{head(k)}$ to denote $x_{head(x,k)}$, and similarly for the tail. For real numbers $a, b, \varepsilon \geq 0$, we use the notation $a = (1 \pm \varepsilon)b$ to convey that $a \in [(1-\varepsilon)b, (1+\varepsilon)b]$. A collection of vectors $\{C_1, \ldots, C_n\} \in [q]^t$ is called a *code* with *alphabet size* q and *block length* t, and we define $\Delta(C_i, C_j) = |\{k : (C_i)_k \neq (C_j)_k\}|$. The *relative distance* of the code is $\max_{i \neq j} \Delta(C_i, C_j)/t$.

We now define the problems that we study in this work, which all involve some *error parameter* $0 < \varepsilon < 1/2$. We want to design a fixed $A \in \mathbb{R}^{m \times n}$ and deterministic algorithm Out for each problem satisfying the following.

Problem 1: In the ℓ_∞/ℓ_1 *recovery problem*, also called the *point query problem*, $\forall x \in \mathbb{R}^n$, $x' = Out(Ax)$ satisfies $\|x - x'\|_\infty \leq \varepsilon \|x\|_1$. The pair (A, Out) furthermore satisfies the *k-tail guarantee* if actually $\|x - x'\|_\infty \leq \varepsilon \|x_{tail(k)}\|_1$.

Problem 2: In the *inner product problem*, $\forall x, y \in \mathbb{R}^n$, $\alpha = Out(Ax, Ay)$ satisfies $|\alpha - \langle x, y \rangle| \leq \varepsilon \|x\|_1 \|y\|_1$.

Problem 3: In the ℓ_1/ℓ_1 *recovery problem with the k-tail guarantee*, $\forall x \in \mathbb{R}^n$, $x' = Out(Ax)$ satisfies $\|x - x'\|_1 \leq (1+\varepsilon)\|x_{tail(k)}\|_1$.

Problem 4: In the ℓ_2 *norm estimation problem*, $\forall x \in \mathbb{R}^n$, $\alpha = Out(Ax)$ satisfies $|\|x\|_2 - \alpha| \leq \varepsilon \|x\|_1$.

We note that for the first, second, and fourth problems above, our errors are additive and not relative. This is because relative error is impossible to achieve with a sublinear number of measurements. If A is a fixed matrix with $m < n$, then it has some non-trivial kernel. Since for all the problems above an Out procedure would have to output 0 when $Ax = 0$ to achieve bounded relative approximation, such a procedure would fail on any input vector in the kernel which is not the 0 vector.

For Problem 2 one could also ask to achieve additive error $\varepsilon \|x\|_p \|y\|_p$ for $p > 1$. For $y = e_i$ for a standard unit vector e_i, this would mean approximating x_i up to additive error $\varepsilon \|x\|_p$. This is not possible unless $m = \Omega(n^{2-2/p})$ for $1 < p \leq 2$ and $m = \Omega(n)$ for $p \geq 2$ [21]. For Problem 3, it is known that the analogous guarantee of returning x' for which $\|x - x'\|_2 \leq \varepsilon \|x_{tail(k)}\|_2$ is not possible unless $m = \Omega(n)$ [14].

1.3 Our Contributions and Related Work

We study the four problems stated above, where we have the deterministic guarantee that a single pair (A, Out) provides the desired guarantee for all input vectors simultaneously. We first show that point query and inner product are equivalent up to changing ε by a constant factor. We then show that any "incoherent matrix" A can be used for these two problems to perform the linear measurements; that is, a matrix A whose columns have unit ℓ_2 norm and such that each pair of columns has dot product at most ε in magnitude. Such matrices can be obtained from the Johnson-Lindenstrauss (JL) lemma [29], almost pairwise independent sample spaces [7,38], or error-correcting codes, and they play a prominent role in compressed sensing [18,36] and mathematical approximation theory [25]. The connection between point query and codes was implicit in [22], though a suboptimal code was used, and the observation that the more general class of incoherent matrices suffices is novel. This connection allows us to show that $m = O(\varepsilon^{-2} \min\{\log n, (\log n/\log(1/\varepsilon))^2\})$ measurements suffice, and where Out and the construction of A are completely deterministic. Alon has shown that any incoherent matrix must have $m = \Omega(\varepsilon^{-2} \log n/\log(1/\varepsilon))$ [6]. Meanwhile the best known lower bound for point query is $m = \Omega(\varepsilon^{-2} + \varepsilon^{-1} \log(\varepsilon n))$ [19,20,27], and the previous best known upper bound was $m = O(\varepsilon^{-2} \log^2 n/(\log 1/\varepsilon + \log\log n))$ [22]. If the construction of A is allowed to be Las Vegas polynomial time, then we can use the Fast Johnson-Lindenstrauss transforms [2,3,4,34] so that Ax can be computed quickly, e.g. in $O(n \log m)$ time as long as $m < n^{1/2-\gamma}$ [3], and with $m = O(\varepsilon^{-2} \log n)$. Our Out algorithm is equally fast. We also show that for point query, if we allow the measurement matrix A to be constructed by a polynomial Monte Carlo algorithm, then the $1/\varepsilon^2$-tail guarantee can be obtained essentially "for free", i.e. by keeping $m = O(\varepsilon^{-2} \log n)$. Previously the work [22] only showed how to obtain the $1/\varepsilon$-tail guarantee "for free" in this sense of not increasing m (though the m in [22] was worse). We note that for randomized algorithms which succeed with high probability for any given input, it suffices to take $m = O(\varepsilon^{-1} \log n)$ by using the CountMin data structure [15], and this is optimal [30] (the lower bound in [30] is stated for the so-called heavy hitters problem, but also applies to the ℓ_∞/ℓ_1 recovery problem).

For the ℓ_1/ℓ_1 sparse recovery problem with the k-tail guarantee, we show a lower bound of $m = \Omega(k \log(\varepsilon n/k)/\varepsilon + k/\varepsilon^2)$. The best upper bound is $O(k \log(n/k)/\varepsilon^2)$ [28]. Our lower bound implies a separation for the complexity of this problem in the case that one must simply pick a random (A, Out) pair which works for some given input x with high probability (i.e. not for all x simultaneously), since [39] showed an $m = O(k \log n \log^3(1/\varepsilon)/\sqrt{\varepsilon})$ upper bound in this case. The first summand of our lower bound uses techniques used in [9,39]. The second summand uses a generalization of an argument of Gluskin [27], which was later rediscovered by Ganguly [20], which showed the lower bound $m = \Omega(1/\varepsilon^2)$ for point query.

Finally, we show how to devise an appropriate (A, Out) for ℓ_2 norm estimation with $m = O(\varepsilon^{-2} \log(\varepsilon^2 n))$, which is optimal. The construction of A is randomized but then works for all x with high probability. The proof takes A according

to known upper bounds on Gelfand widths, and the recovery procedure Out requires solving a simple convex program. As far as we are aware, this is the first work to investigate this problem in the deterministic setting. In the case that (A, Out) can be chosen randomly to work for any fixed x with high probability, one can use the AMS sketch [8] with $m = O(\varepsilon^{-2}\log(1/\delta))$ to succeed with probability $1-\delta$ and to obtain the better guarantee $\varepsilon\|x\|_2$. The AMS sketch can also be used for the inner product problem to obtain error guarantee $\varepsilon\|x\|_2\|y\|_2$ with the same m.

Due to space constraints, many of our proofs are omitted or abbreviated. Full proofs can be found in the full version.

2 Point Query and Inner Product Estimation

We first show that the problems of point query and inner product estimation are equivalent up to changing the error parameter ε by a constant factor.

Theorem 1. *Any solution (A, Out') to inner product estimation with error parameter ε yields a solution (A, Out) to the point query problem with error parameter ε. Also, a solution (A, Out) for point query with error ε yields a solution (A, Out') to inner product with error 12ε. The time complexities of Out and Out' are equal up to* poly(n) *factors.*

Proof: Let (A, Out') be a solution to the inner product problem such that $Out'(Ax, Ay) = \langle x, y\rangle \pm \varepsilon\|x\|_1\|y\|_1$. Then given $x \in \mathbb{R}^n$, to solve the point query problem we return the vector with $Out(Ax)_i = Out'(Ax, Ae_i)$, and our guarantees are immediate.

Now let (A, Out) be a solution to the point query problem. Given $x, y \in \mathbb{R}^n$, let $x' = Out(Ax), y' = Out(Ay)$. Our estimate for $\langle x, y\rangle$ is $Out'(Ax, Ay) = \left\langle x'_{head(1/\varepsilon)}, y'_{head(1/\varepsilon)}\right\rangle$. Correctness is proven in the full version. ■

Since the two problems are equivalent up to changing ε by a constant factor, we focus on point query. We first have the following lemma, stating that any *incoherent matrix* A has a correct associated Out procedure (namely, multiplication by A^T). An incoherent matrix, is an $m \times n$ matrix A for which all columns A_i of A have unit ℓ_2 norm, and for all $i \neq j$ we have $|\langle A_i, A_j\rangle| \leq \varepsilon$.

Lemma 1. *Any incoherent matrix A with error parameter ε has an associated* poly(mn)*-time deterministic recovery procedure Out for which (A, Out) is a solution to the point query problem. In fact, for any $x \in \mathbb{R}^n$, given Ax and $i \in [n]$, the output x'_i satisfies $|x'_i - x_i| \leq \varepsilon\|x_{-i}\|_1$.*

It is known that any incoherent matrix has $m = \Omega((\log n)/(\varepsilon^2 \log 1/\varepsilon))$ [6], and the JL lemma implies such matrices with $m = O((\log n)/\varepsilon^2)$ [29]. For example, there exist matrices in $\{-1/\sqrt{m}, 1/\sqrt{m}\}^{m\times n}$ satisfying this property [1], which can also be found in poly(n) time [41] (we note that [41] gives running time exponential in precision, but the proof holds if the precision is taken to

be $O(\log(n/\varepsilon))$. It is also known that incoherent matrices can be obtained from almost pairwise independent sample spaces [7,38] or error-correcting codes, and thus these tools can also be used to solve the point query problem. The connection to codes was already implicit in [22], though the code used in that work is suboptimal, as we will show soon. Below we elaborate on what bounds these tools provide for incoherent matrices, and thus the point query problem.

Incoherent matrices from JL: The upside of the connection to the JL lemma is that we can obtain incoherent matrices A such that Ax can be computed quickly, via the Fast Johnson-Lindenstrauss Transform introduced by Ailon and Chazelle [2] or related subsequent works. The JL lemma states the following.

Theorem 2 (JL lemma). *For any $x_1, \ldots, x_N \in \mathbb{R}^n$ and any $0 < \varepsilon < 1/2$, there exists $A \in \mathbb{R}^{m \times n}$ with $m = O(\varepsilon^{-2} \log N)$ such that for all $i, j \in [N]$ we have $\|Ax_i - Ax_j\|_2 = (1 \pm \varepsilon)\|x_i - x_j\|_2$.*

Consider the matrix A obtained from the JL lemma when the set of vectors is $\{0, e_1, \ldots, e_n\} \in \mathbb{R}^n$. Then columns A_i of A have ℓ_2 norm $1 \pm \varepsilon$, and furthermore for $i \neq j$ we have $|\langle A_i, A_j \rangle| = (\|A_i - A_j\|_2^2 - \|A\|_i^2 - \|A\|_j^2)/2 = ((1 \pm \varepsilon)^2 2 - (1 \pm \varepsilon) - (1 \pm \varepsilon))/2 \leq 2\varepsilon + \varepsilon^2/2$. By scaling each column to have ℓ_2 norm exactly 1, we still preserve that dot products between pairs of columns are $O(\varepsilon)$ in magnitude.

Incoherent matrices from almost pairwise independence: An *ε-almost pairwise independent sample space* a set $S \subseteq \{-1, 1\}^n$ satisfying the following. For any $i \neq j \in [n]$, the ℓ_1 distance between the uniform distribution over $\{-1, 1\}^2$ and the distribution of x_i, x_j when x is drawn uniformly at random from S is at most ε. A matrix whose rows are the elements of S, divided by a scale factor of $\sqrt{S}$, is incoherent. Details are in the full version, but we do not delve deeper since this approach does not improve upon the bounds via JL matrices.

Incoherent matrices from codes: Finally we explain the connection between incoherent matrices and codes. A connection to balanced binary codes was made in [6], and to arbitrary codes over larger alphabets without detail in a remark in [5]. Though not novel, we elaborate on this latter connection for the sake of completeness. Let $\mathcal{C} = \{C_1, \ldots, C_n\}$ be a code with alphabet size q, block length t, and relative distance $1 - \varepsilon$. The fact that such a code gives rise to a matrix $A \in \mathbb{R}^{m \times n}$ for point query with error parameter ε was implicit in [22], but we make it explicit here. We let $m = qt$ and conceptually partition the rows of A arbitrarily into t sets each of size q. For the column A_i, let $(A_i)_{j,k}$ denote the entry of A_i in the kth coordinate of the jth block. We set $(A_i)_{j,k} = 1/\sqrt{t}$ if $(C_i)_j = k$, and $(A_i)_{j,k} = 0$ otherwise. Each column has exactly t non-zero entries of value $1/\sqrt{t}$, and thus has ℓ_2 norm 1. Furthermore, for $i \neq j$, $\langle A_i, A_j \rangle = (t - \Delta(C_i, C_j))/t \leq \varepsilon$.

The work [22] instantiated the above with the following *Chinese remainder code* [35,42,44], which yielded $m = O(\varepsilon^{-2} \log^2 n / (\log 1/\varepsilon + \log\log n))$. We observe here that this bound is never optimal. A random code with $q = 2/\varepsilon$ and $t = O(\varepsilon^{-1} \log n)$ has the desired properties by applying the Chernoff bound on a pair

of codewords, then a union bound over codewords (alternatively, such a code is promised by the Gilbert-Varshamov (GV) bound). If ε is sufficiently small, a Reed-Solomon code performs even better. That is, we take a finite field $\mathbb{F}_q$ for $q = \Theta(\varepsilon^{-1}\log n/(\log\log n + \log(1/\varepsilon)))$ and $q = t$, and each C_i corresponds to a distinct degree-d polynomial p_i over $\mathbb{F}_q$ for $d = \Theta(\log n/(\log\log n + \log(1/\varepsilon)))$ (note there are at least $q^d > n$ such polynomials). We set $(C_i)_j = p_i(j)$. The relative distance is as desired since $p_i - p_j$ has at most d roots over $\mathbb{F}_q$ and thus can be 0 at most $d \leq \varepsilon t$ times. This yields $qt = O(\varepsilon^{-2}(\log n/(\log\log n + \log(1/\varepsilon))^2)$, which surpasses the GV bound for $\varepsilon < 2^{-\Omega(\sqrt{\log n})}$, and is always better than the Chinese remainder code. We note that this construction of a binary matrix based on Reed-Solomon codes is identical to one used by Kautz and Singleton in the different context of group testing [33].

Time	m	Details	Explicit?
$O((n\log n)/\varepsilon^2)$	$O(\varepsilon^{-2}\log n)$	$A \in \{-1/\sqrt{m}, 1/\sqrt{m}\}^{m\times n}$ [1,41]	yes
$O((n\log n)/\varepsilon)$	$O(\varepsilon^{-2}\log n)$	sparse JL [31], GV code	no
$O(nd\log^2 d\log\log d/\varepsilon)$	$O(d^2/\varepsilon^2)$	Reed-Solomon code	yes
$O_\gamma(n\log m + m^{2+\gamma})$	$O(\varepsilon^{-2}\log n)$	FFT-based JL [3]	no
$O(n\log n)$	$O(\varepsilon^{-2}\log^5 n)$	FFT-based JL [4,34]	no

Fig. 1. Implications for point query from JL matrices and codes. Time indicates the running time to compute Ax given x. In the case of Reed-Solomon, $d = O(\log n/(\log\log n + \log(1/\varepsilon)))$. We say the construction is "explicit" if A can be computed in deterministic time poly(n); otherwise we only provide a polynomial time Las Vegas algorithm to construct A.

In Figure 1 we elaborate on what known constructions of codes and JL matrices provide for us in terms of point query. In the case of running time for the Reed-Solomon construction, we use that degree-d polynomials can be evaluated on $d+1$ points in a total of $O(d\log^2 d\log\log d)$ field operations over $\mathbb{F}_q$ [43, Ch. 10]. In the case of [3], the constant $\gamma > 0$ can be chosen arbitrarily, and the constant in the big-Oh depends on $1/\gamma$. We note that except in the case of Reed-Solomon codes, the construction of A is randomized (though once A is generated, incoherence can be verified in polynomial time, thus providing a poly(n)-time Las Vegas algorithm).

Note that Lemma 1 did not just give us error $\varepsilon\|x\|_1$, but actually gave us $|x_i - x'_i| \leq \varepsilon\|x_{-i}\|_1$, which is stronger. We now show that an even stronger guarantee is possible. We will show that in fact it is possible to obtain $\|x - x'\|_\infty \leq \varepsilon\|x_{tail(1/\varepsilon^2)}\|_1$ while increasing m by only an additive $O(\varepsilon^{-2}\log(\varepsilon^2 n))$, which is less than our original m except potentially in the Reed-Solomon construction. The idea is to, in parallel, recover a good approximation of $x_{head(1/\varepsilon^2)}$ with error proportional to $\|x_{tail(1/\varepsilon^2)}\|_1$ via compressed sensing, then to subtract from Ax before running our recovery procedure. We now give details.

We in parallel run a *k-sparse recovery* algorithm which has the following guarantee: there is a pair (B, Out') such that for any $x \in \mathbb{R}^n$, we have that $x' = Out'(Bx) \in \mathbb{R}^n$ satisfies $\|x' - x\|_2 \leq O(1/\sqrt{k})\|x_{tail(k)}\|_1$. Such a matrix

B can be taken to have the *restricted isometry property of order* k (k-RIP), i.e. that it preserves the ℓ_2 norm up to a small multiplicative constant factor for all k-sparse vectors in $\mathbb{R}^n$.[1] It is known [26] that any such x' also satisfies the guarantee that $\|x'_{head(k)} - x\|_1 \leq O(1)\|x_{tail(k)}\|_1$, where $x'_{head(k)}$ is the vector which agrees with x' on the top k coordinates in magnitude and is 0 on the remaining coordinates. Moreover, it is also known [10] that if B satisfies the JL lemma for a particular set of $N = (en/k)^{O(k)}$ points in $\mathbb{R}^n$, then B will be k-RIP. The associated output procedure Out' takes Bx and outputs $\text{argmin}_{z|Bx=Bz}\|z\|_1$ by solving a linear program [12]. All the JL matrices in Figure 1 provide this guarantee with $O(k\log(en/k))$ rows, except for the last row which satisfies k-RIP with $O(k\log(en/k)\log^2 k\log(k\log n))$ rows [40].

Theorem 3. *Let A be an incoherent matrix A with error parameter ε, and let B be k-RIP. Then there is an output procedure Out which for any $x \in \mathbb{R}^n$, given only Ax, Bx, outputs a vector x' with $\|x' - x\|_\infty \leq \varepsilon\|x_{tail(k)}\|_1$.*

Proof: Given Bx, we first run the k-sparse recovery algorithm to obtain a vector y with $\|x - y\|_1 = O(1)\|x_{tail(k)}\|_1$. We then construct our output vector x' coordinate by coordinate. To construct x'_i, we replace y_i with 0, obtaining the vector z^i. Then we compute $A(x - z^i)$ and run the point query output procedure associated with A and index i. The guarantee is that the output w^i of the point query algorithm satisfies $|w^i_i - (x - z^i)_i| \leq \varepsilon\|(x - z^i)_{-i}\|_1$, where

$$\|(x - z^i)_{-i}\|_1 = \|(x - y)_{-i}\|_1 \leq \|x - y\|_1 = O(1)\|x_{tail(k)}\|_1,$$

and so $|(w^i + z^i)_i - x_i| = O(\varepsilon)\|x_{tail(k)}\|_1$. If we define our output vector by $x'_i = w^i_i + z^i_i$ and rescale ε by a constant factor, this proves the theorem. ∎

By setting $k = 1/\varepsilon^2$ in Theorem 3 and stacking the rows of a k-RIP and incoherent matrix each with $O((\log n)/\varepsilon^2)$ rows, we obtain the following corollary.

Corollary 1. *There is an $m \times n$ matrix A and associated output procedure Out which for any $x \in \mathbb{R}^n$, given Ax, outputs a vector x' with $\|x' - x\|_\infty \leq \varepsilon\|x_{tail(1/\varepsilon^2)}\|_1$. Here $m = O((\log n)/\varepsilon^2)$.*

It is also possible to obtain a tail-error guarantee for inner product.

Theorem 4. *Suppose $1/\varepsilon^2 < n/2$. There is an (A, Out) with $A \in \mathbb{R}^{m\times n}$ for $m = O(\varepsilon^{-2}\log n)$ such that for any $x, y \in \mathbb{R}^n$, $Out(Ax, Ay)$ gives an output which is $\langle x, y\rangle \pm \varepsilon(\|x\|_2\|y_{tail(1/\varepsilon^2)}\|_1 + \|x_{tail(1/\varepsilon^2)}\|_1\|y\|_2) + \varepsilon^2\|x_{tail(1/\varepsilon^2)}\|_1\|y_{tail(1/\varepsilon^2)}\|_1$.*

Here we state a lower bound for the point query problem. The proof can be found in the full version and follows from the works [20,27] and volume arguments as used in [9].

[1] Unfortunately currently the only known constructions of k-RIP constructions with the values of m we discuss are Monte Carlo, forcing our algorithms in this section with the k-tail guarantee to only be Monte Carlo polynomial time when constructing the measurement matrix.

Theorem 5. *Let $0 < \varepsilon < \varepsilon_0$ for some universal constant $\varepsilon_0 < 1$. Suppose $1/\varepsilon^2 < n/2$, and A is an $m \times n$ matrix for which given Ax it is always possible to produce a vector x' such that $\|x - x'\|_\infty \leq \varepsilon \|x_{tail(k)}\|_1$. Then $m = \Omega(k \log(n/k)/\log k + \varepsilon^{-2} + \varepsilon^{-1} \log n)$.*

3 Lower Bounds for ℓ_1/ℓ_1 Recovery

Recall in the ℓ_1/ℓ_1-recovery problem, we would like to design a matrix $A \in \mathbb{R}^{m \times n}$ such that for any $x \in \mathbb{R}^n$, given Ax we can recover $x' \in \mathbb{R}^n$ such that $\|x - x'\|_1 \leq (1+\varepsilon)\|x_{tail(k)}\|_1$. We now show two lower bounds.

Theorem 6. *Let $0 < \varepsilon < 1/\sqrt{8}$ be arbitrary, and k be an integer. Suppose $k/\varepsilon^2 < (n-1)/2$. Then any matrix $A \in \mathbb{R}^{m \times n}$ which allows ℓ_1/ℓ_1-recovery with the k-tail guarantee with error ε must have $m \geq \min\{n/2, (1/16)k/\varepsilon^2\}$.*

Proof: Without loss of generality we may assume that the rows of A are orthonormal. This is because first we can discard rows of A until the rows remaining form a basis for the rowspace of A. Call this new matrix with potentially fewer rows A'. Note that any dot products of rows of A with x that the recovery algorithm uses can be obtained by taking linear combinations of entries of $A'x$. Next, we can then find a matrix $T \in \mathbb{R}^{m \times m}$ so that TA' has orthonormal rows, and given $TA'x$ we can recover $A'x$ in post-processing by left-multiplication with T^{-1}. We henceforth assume that the rows of A are orthonormal. Since $A \cdot 0 = 0$, and our recovery procedure must in particular be accurate for $x = 0$, the recovery procedure must output $x' = 0$ for any $x \in ker(A)$. We consider $x = (I - A^T A)y$ for $y = \sum_{i=1}^{k} \sigma_i e_{\pi(i)}$. Here π is a random permutation on n elements, and $\sigma_1, \ldots, \sigma_k$ are independent and uniform random variables in $\{-1, 1\}$. Since $x \in ker(A)$, which follows since $AA^T = I$ by orthonormality of the rows of A, the recovery algorithm will output $x' = 0$. Nevertheless, in the full version we show that unless $m \geq \min\{n/2, (1/16)k/\varepsilon^2\}$, we will have $\|x\|_1 > (1+\varepsilon)\|x_{tail(k)}\|_1$ with positive probability so that by the probabilistic method there exists $x \in ker(A)$ for which $x' = 0$ is not a valid output. ■

We now give another lower bound via a different approach. As in [9,39], we use 2-party communication complexity to prove an $\Omega((k/\varepsilon)\log(\varepsilon n/k))$ bound on the number of rows of any ℓ_1/ℓ_1 sparse recovery scheme. The main difference from prior work is that we use deterministic communication complexity and a different communication problem.

We show how to use a pair (A, Out) with the property that for all vectors z, the output z' of $Out(Az)$ satisfies $\|z - z'\|_1 \leq (1+\varepsilon)\|z_{tail(k)}\|_1$, to construct a correct protocol for the equality function on strings $x, y \in \{0,1\}^r$ for $r = \Theta((k/\varepsilon)\log n \log(\varepsilon n/k))$, where the communication is an $O(\log n)$ factor larger than the number of rows of A. We then show how this implies the number of rows of A is $\Omega((k/\varepsilon)\log(\varepsilon n/k))$. Details are in the full version.

Theorem 7. *Any matrix A which allows ℓ_1/ℓ_1-recovery with the k-tail guarantee with error ε satisfies $m = \Omega((k/\varepsilon)\log(\varepsilon n/k))$.*

4 Deterministic Norm Estimation and the Gelfand Width

Theorem 8. *For $1 \leq p < q \leq \infty$, let m be the minimum number such that there is an $n - m$ dimensional subspace S of $\mathbb{R}^n$ satisfying $\sup_{v \in S} \frac{\|v\|_q}{\|v\|_p} \leq \varepsilon$. Then there is an $m \times n$ matrix A and associated output procedure Out which for any $x \in \mathbb{R}^n$, given Ax, outputs an estimate of $\|v\|_q$ with additive error at most $\varepsilon\|v\|_p$. Moreover, any matrix A with fewer rows fails to perform this task.*

Proof: Consider a matrix A whose kernel is such a subspace. For any sketch z, we need to return a number in the range $[\|x\|_q - \varepsilon\|x\|_p, \|x\|_q + \varepsilon\|x\|_p]$ for any x satisfying $Ax = z$. Assume for contradiction that it is not possible. Then there exist x and y such that $Ax = Ay$ but $\|x\|_q - \varepsilon\|x\|_p > \|y\|_q + \varepsilon\|y\|_p$. However, since $x - y$ is in the kernel of A, $\|x\|_q - \|y\|_q \leq \|x - y\|_q \leq \varepsilon\|x - y\|_p \leq \varepsilon(\|x\|_p + \|y\|_p)$. Thus, we have a contradiction. This argument also shows that given the sketch z, the output procedure can return $\min_{x:Ax=z} \|x\|_q + \varepsilon\|x\|_p$. This is a convex optimization problem that can be solved in polynomial time using the ellipsoid algorithm; details are in the full version.

For the lower bound, consider a matrix A with fewer than m rows. Then in the kernel of A, there exists v such that $\|v\|_q > \varepsilon\|v\|_p$. Both v and the zero vector give the same sketch (a zero vector). However, by the stated requirement, we need to output 0 for the zero vector but some positive number for v. Thus, no matrix A with fewer than m rows can solve the problem. ■

The subspace S of highest dimension of $\mathbb{R}^n$ satisfying $\sup_{v \in S} \frac{\|v\|_q}{\|v\|_p} \leq \varepsilon$ is related to the *Gelfand width*, a well-studied notion in functional analysis. For $p < q$, the Gelfand width of order m of ℓ_p and ℓ_q unit balls in $\mathbb{R}^n$ is defined as the infimum over all subspaces $A \subseteq \mathbb{R}^n$ of codimension m of $\sup_{v \in A} \frac{\|v\|_q}{\|v\|_p}$. Using known bounds for the Gelfand width for $p = 1$ and $q = 2$ [19,23], we obtain the following corollary.

Corollary 2. *Assume that $1/\varepsilon^2 < n/2$. There is an $m \times n$ matrix A and associated output procedure Out which for any $x \in \mathbb{R}^n$, given Ax, outputs an estimate e such that $\|x\|_2 - \varepsilon\|x\|_1 \leq e \leq \|x\|_2 + \varepsilon\|x\|_1$. Here $m = O(\varepsilon^{-2} \log(\varepsilon^2 n))$ and this bound for m is tight.*

Acknowledgments. We thank Raghu Meka for answering several questions about almost k-wise independent sample spaces. We thank an anonymous reviewer for pointing out the connection between incoherent matrices and ε-biased spaces.

References

1. Achlioptas, D.: Database-friendly random projections: Johnson-Lindenstrauss with binary coins. J. Comput. Syst. Sci. 66(4), 671–687 (2003)
2. Ailon, N., Chazelle, B.: The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. SIAM J. Comput. 39(1), 302–322 (2009)

3. Ailon, N., Liberty, E.: Fast dimension reduction using Rademacher series on dual BCH codes. Discrete & Computational Geometry 42(4), 615–630 (2009)
4. Ailon, N., Liberty, E.: Almost optimal unrestricted fast Johnson-Lindenstrauss transform. In: Proceedings of the 22nd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 185–191 (2011)
5. Alon, N.: Problems and results in extremal combinatorics - I. Discrete Mathematics 273(1-3), 31–53 (2003)
6. Alon, N.: Perturbed identity matrices have high rank: Proof and applications. Combinatorics, Probability & Computing 18(1-2), 3–15 (2009)
7. Alon, N., Goldreich, O., Håstad, J., Peralta, R.: Simple construction of almost k-wise independent random variables. Rand. Struct. Alg. 3(3), 289–304 (1992)
8. Alon, N., Matias, Y., Szegedy, M.: The Space Complexity of Approximating the Frequency Moments. JCSS 58(1), 137–147 (1999)
9. Ba, K.D., Indyk, P., Price, E., Woodruff, D.P.: Lower bounds for sparse recovery. In: SODA, pp. 1190–1197 (2010)
10. Baraniuk, R., Davenport, M.A., DeVore, R., Wakin, M.: A simple proof of the Restricted Isometry Property. Constructive Approximation 28(3), 253–263 (2008)
11. Barbará, D., Wu, N., Jajodia, S.: Detecting novel network intrusions using Bayes estimators. In: Proceedings of the 1st SIAM International Conference on Data Mining (2001)
12. Candès, E., Romberg, J., Tao, T.: Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. IEEE Trans. Information Theory 52(2), 489–509 (2006)
13. Charikar, M., Chen, K., Farach-Colton, M.: Finding frequent items in data streams. Theor. Comput. Sci. 312(1), 3–15 (2004)
14. Cohen, A., Dahmen, W., DeVore, R.A.: Compressed sensing and best k-term approximation. J. Amer. Math. Soc. 22, 211–231 (2009)
15. Cormode, G., Muthukrishnan, S.: An improved data stream summary: the count-min sketch and its applications. J. Algorithms 55(1), 58–75 (2005)
16. Cormode, G., Muthukrishnan, S.: What's hot and what's not: tracking most frequent items dynamically. ACM Trans. Database Syst. 30(1), 249–278 (2005)
17. Demaine, E.D., López-Ortiz, A., Munro, J.I.: Frequency Estimation of Internet Packet Streams with Limited Space. In: Möhring, R.H., Raman, R. (eds.) ESA 2002. LNCS, vol. 2461, pp. 348–360. Springer, Heidelberg (2002)
18. Donoho, D.L., Huo, X.: Uncertainty principles and ideal atomic decomposition. IEEE Trans. Inform. Th. 47, 2558–2567 (2001)
19. Foucart, S., Pajor, A., Rauhut, H., Ullrich, T.: The Gelfand widths of ℓ_p-balls for $0 < p \leq 1$. Journal of Complexity 26(6), 629–640 (2010)
20. Ganguly, S.: Lower Bounds on Frequency Estimation of Data Streams (Extended Abstract). In: Hirsch, E.A., Razborov, A.A., Semenov, A., Slissenko, A. (eds.) CSR 2008. LNCS, vol. 5010, pp. 204–215. Springer, Heidelberg (2008)
21. Ganguly, S.: Deterministically Estimating Data Stream Frequencies. In: Du, D.-Z., Hu, X., Pardalos, P.M. (eds.) COCOA 2009. LNCS, vol. 5573, pp. 301–312. Springer, Heidelberg (2009)
22. Ganguly, S., Majumder, A.: CR-PRECIS: A Deterministic Summary Structure for Update Data Streams. In: Chen, B., Paterson, M., Zhang, G. (eds.) ESCAPE 2007. LNCS, vol. 4614, pp. 48–59. Springer, Heidelberg (2007)
23. Garnaev, A.Y., Gluskin, E.D.: On the widths of the Euclidean ball. Soviet Mathematics Doklady 30, 200–203 (1984)
24. Gilbert, A.C., Kotidis, Y., Muthukrishnan, S., Strauss, M.J.: Quicksand: Quick summary and analysis of network data. DIMACS Technical Report 2001-43 (2001)

25. Gilbert, A.C., Muthukrishnan, S., Strauss, M.: Approximation of functions over redundant dictionaries using coherence. In: SODA, pp. 243–252 (2003)
26. Gilbert, A.C., Strauss, M.J., Tropp, J.A., Vershynin, R.: One sketch for all: fast algorithms for compressed sensing. In: STOC, pp. 237–246 (2007)
27. Gluskin, E.D.: On some finite-dimensional problems in the theory of widths. Vestn. Leningr. Univ. Math. 14, 163–170 (1982)
28. Indyk, P., Ružić, M.: Near-optimal sparse recovery in the L_1 norm. In: FOCS, pp. 199–207 (2008)
29. Johnson, W.B., Lindenstrauss, J.: Extensions of Lipschitz mappings into a Hilbert space. Contemporary Mathematics 26, 189–206 (1984)
30. Jowhari, H., Saglam, M., Tardos, G.: Tight bounds for L_p samplers, finding duplicates in streams, and related problems. In: PODS, pp. 49–58 (2011)
31. Kane, D.M., Nelson, J.: Sparser Johnson-Lindenstrauss transforms. In: SODA, pp. 1195–1206 (2012)
32. Karp, R.M., Shenker, S., Papadimitriou, C.H.: A simple algorithm for finding frequent elements in streams and bags. ACM Trans. Database Syst. 28, 51–55 (2003)
33. Kautz, W.H., Singleton, R.C.: Nonrandom binary superimposed codes. IEEE Trans. Inf. Theory 10, 363–377 (1964)
34. Krahmer, F., Ward, R.: New and improved Johnson-Lindenstrauss embeddings via the Restricted Isometry Property. SIAM J. Math. Anal. 43(3), 1269–1281 (2011)
35. Krishna, H., Krishna, B., Lin, K.-Y., Sun, J.-D.: Computational Number Theory and Digital Signal Processing: Fast Algorithms and Error Control Techniques. CRC, Boca Raton (1994)
36. Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Trans. Signal Process. 41(12), 3397–3415 (1993)
37. Misra, J., Gries, D.: Finding repeated elements. Sci. Comput. Program. 2(2), 143–152 (1982)
38. Naor, J., Naor, M.: Small-bias probability spaces: Efficient constructions and applications. SIAM J. Comput. 22(4), 838–856 (1993)
39. Price, E., Woodruff, D.P.: (1 + eps)-approximate sparse recovery. In: FOCS, pp. 295–304 (2011)
40. Rudelson, M., Vershynin, R.: On sparse reconstruction from Fourier and Gaussian measurements. Communications on Pure and Applied Mathematics 61, 1025–1045 (2008)
41. Sivakumar, D.: Algorithmic derandomization via complexity theory. In: STOC, pp. 619–626 (2002)
42. Soderstrand, M.A., Jenkins, W.K., Jullien, G.A., Taylor, F.J.: Residue Number System Arithmetic: Modern Applications in Digital Signal Processing. IEEE Press, New York (1986)
43. von zur Gathen, J., Gerhard, J.: Modern Computer Algebra. Cambridge University Press (1999)
44. Watson, R.W., Hastings, C.W.: Self-checked computation using residue arithmetic. Proc. IEEE 4(12), 1920–1931 (1966)

A New Upper Bound on the Query Complexity for Testing Generalized Reed-Muller Codes

Noga Ron-Zewi[1,⋆] and Madhu Sudan[2]

[1] Department of Computer Science, Technion, Haifa
nogaz@cs.technion.ac.il
[2] Microsoft Research New England, Cambridge, MA
madhu@mit.edu

Abstract. Over a finite field $\mathbb{F}_q$ the (n, d, q)-Reed-Muller code is the code given by evaluations of n-variate polynomials of total degree at most d on all points (of $\mathbb{F}_q^n$). The task of testing if a function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ is close to a codeword of an (n, d, q)-Reed-Muller code has been of central interest in complexity theory and property testing. The query complexity of this task is the minimal number of queries that a tester can make (minimum over all testers of the maximum number of queries over all random choices) while accepting all Reed-Muller codewords and rejecting words that are δ-far from the code with probability $\Omega(\delta)$. (In this work we allow the constant in the Ω to depend on d.)

For codes over a prime field $\mathbb{F}_q$ the optimal query complexity is well-known and known to be $\Theta(q^{\lceil (d+1)/(q-1) \rceil})$, and the test consists of testing if f is a degree d polynomial on a randomly chosen $(\lceil (d+1)/(q-1) \rceil)$-dimensional affine subspace of $\mathbb{F}_q^n$. If q is not a prime, then the above quantity remains a lower bound, whereas the previously known upper bound grows to $O(q^{\lceil (d+1)/(q-q/p) \rceil})$ where p is the characteristic of the field $\mathbb{F}_q$. In this work we give a new upper bound of $(cq)^{(d+1)/q}$ on the query complexity, where c is a universal constant. Thus for every p and sufficiently large q this bound improves over the previously known bound by a polynomial factor.

In the process we also give new upper bounds on the "spanning weight" of the dual of the Reed-Muller code (which is also a Reed-Muller code). The spanning weight of a code is the smallest integer w such that codewords of Hamming weight at most w span the code. The main technical contribution of this work is the design of tests that test a function by *not* querying its value on an entire subspace of the space, but rather on a carefully chosen (algebraically nice) subset of the points from low-dimensional subspaces.

1 Introduction

In this work we present new upper bounds on the query complexity of testing Reed-Muller codes, the codes obtained by evaluations of multivariate low-degree

⋆ Research conducted in part while this author was an intern at Microsoft Research New-England, Cambridge, MA, and supported in part by the Israel Ministry of Science and Technology.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 639–650, 2012.
© Springer-Verlag Berlin Heidelberg 2012

polynomials, over general fields. In the process we also give new upper bounds on the spanning weight of Reed-Muller codes. We explain these terms and our results below.

We start with the definition of Reed-Muller codes. Let $\mathbb{F}_q$ denote the finite field on q elements. Throughout we will let $q = p^s$ for prime p and integer s. The Reed-Muller codes have two parameters in addtion to the field size, namely the degree d and number of variables n. The (n, d, q)-Reed-Muller code $\mathrm{RM}[n, d, q]$ is the set of functions from $\mathbb{F}_q^n$ to $\mathbb{F}_q$ that are evaluations of n-variate polynomials of total degree at most d.

1.1 Testing Reed-Muller Codes

We define the notion of testing the "Reed-Muller" property as a special case of property testing. We let $\{\mathbb{F}_q^n \to \mathbb{F}_q\}$ denote the set of all functions mapping $\mathbb{F}_q^n$ to $\mathbb{F}_q$. A property $\mathcal{F}$ is simply a subset of such functions. For $f, g : \mathbb{F}_q^n \to \mathbb{F}_q$ we say the distance between them $\delta(f, g)$ is the fraction of points of $\mathbb{F}_q^n$ where they disagree. We let $\delta(f, \mathcal{F})$ denote the minimum distance between f and a function in $\mathcal{F}$. We say f is δ-close to $\mathcal{F}$ if $\delta(f, \mathcal{F}) \leq \delta$ and δ-far otherwise.

A (k, ϵ)-tester for the property $\mathcal{F} \subseteq \{\mathbb{F}_q^n \to \mathbb{F}_q\}$ is a randomized algorithm that makes at most k queries to an oracle for a function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ and accepts if $f \in \mathcal{F}$ and rejects $f \notin \mathcal{F}$ with probability at least $\epsilon\delta(f, \mathcal{F})$.

For fixed d and q, we consider *query complexity* of testing the property of being a degree d multivariate polynomial over $\mathbb{F}_q$. Specifically, the query complexity $k = k(d, q)$, is the minimum integer such that there exists an ϵ such that for all n there is a (k, ϵ)-tester for the $\mathrm{RM}[n, d, q]$ property. (So the error ϵ of the tester is allowed to depend on q and d, but not on n.)

The query complexity of low-degree testing is a well-studied question and has played a role in many results in computational complexity including in the PCP theorem ([ALM+98] and subsequent works), and in the works of Viola and Wigderson [VW08] and Barak et al. [BGH+11]. Many of these results depend not only on a tight analysis of $k(d, q)$ but also a tight analysis of the parameter ϵ, but in this work we only focus on the first quantity. Below we describe what was known about these quantities.

For the case when d is (sufficiently) smaller than the field size, the works of Rubinfeld and Sudan [RS96] and Friedl and Sudan [FS95] show that $k(d, q) = d + 2$ (provided $d < q - q/p$). For the case when $q = 2$ and d is arbitrary, this quantity was analyzed in the work of Alon et al [AKK+05] who show that $k(d, 2) = 2^{d+1}$ (exactly). Jutla et al [JPRZ09] and Kaufman and Ron [KR06] explored this question for general q and d (the former only considered prime q) and showed that $k(d, q) \leq q^{\lceil (d+1)/(q-q/p) \rceil}$. In [KR06] it is also shown that the bound is tight (to within a factor of q) if q is a prime. However for the non-prime case the only known lower bound on the query complexity was $k(d, q) \geq q^{(d+1)/(q-1)}$ (which is roughly the upper bound raised to the power of $(p-1)/p$). (In the following sections we describe the conceptual reason for this gap in knowledge.)

In this work we give a new upper bound on $k(d,q)$ which is closer to the lower bound when p is a constant and d and q are going to infinity. We state our main theorem below.

Theorem 1 (Main). *Let $q = p^s$ for prime p and positive integer s. Then there exists a constant $c_q \leq 3q^4$ such that for every d and n, the Reed-Muller code $\mathrm{RM}[n,d,q]$ has a $(k, \Omega(1/k^2))$-local tester, for $k = k(d,q) \leq c_q \cdot \left(2^{p-1}+p-1\right)^{(d+1)/(q(p-1))} q^{(d+1)/q}$. In particular $k(d,q) \leq 3q^4 \cdot (3q)^{(d+1)/q}$.*

We note that when p goes to infinity the bound on $k(d,q)$ tends to $c_q \cdot (3q)^{(d+1)/q}$. We also note that the constant c_q is not optimized in our proofs and it seems quite plausible that it can be improved using more careful analysis. The more serious factor (especially when one considers a constant q and $d \to \infty$) is the constant factor multiplying q in the base of the exponent. Our techniques do seem to be unable to improve this beyond $(2^{p-1}+p-1)^{1/(p-1)}$ which is always between 2 and 3 (while the lower bounds suggest a constant which is close to 1).

We note that the above result does not compare well with previous bounds if one take the "soundness" parameter (ϵ) into account. Previous results by Bhattacharyya et al. [BKS+10] for $q = 2$ and Haramaty et al. [HSS11] for general q give a (k', ϵ_0)-local tester for ϵ_0 depending only on q (but independent of d) and $k' = q^{\lceil \frac{d+1}{q-q/p} \rceil}$. To get such a soundness independent of d, Theorem 1 yields a (k^3, ϵ_1)-local tester for ϵ_1 being some universal constant. Thus for small q and growing d this is worse than the results of [BKS+10, HSS11]. However for d and q growing at the same rate (for instance) our result does give the best bounds even if we want the soundness to be some absolute constant.

Theorem 1 is proved by proving that the Reed-Muller code $\mathrm{RM}[n,d,q]$ has a "k-single-orbit characterization" (a notion we will define later, see Definition 2 and Theorem 3). This will imply the testing result immediately by a result of Kaufman and Sudan [KS07].

1.2 Spanning Weight

It is well-known (cf. [BHR05]) that the query complexity of testing a linear code C is lower bounded by the "minimum distance" of its dual, where the minimum distance of a code is the minimum weight of a non-zero codeword. (The weight of a word is simply the number of non-zero coordinates.) Applied to the Reed-Muller code $\mathrm{RM}[n,d,q]$ this suggests a lower bound via the minimum distance of its dual, which also turns out to be a Reed-Muller code. Specifically the dual of $\mathrm{RM}[n,d,q]$ is $\mathrm{RM}[n, n(q-1)-d-1, q]$. The minimum distance of the latter is well-known and is (roughly) $q^{(d+1)/(q-1)}$ and this leads to the tight analysis of the query complexity of Reed-Muller codes over prime fields.

Over non-prime fields however this bound has not been matched, so one could turn to potentially stronger lower bounds. A natural such bound would be the "spanning weight" of the dual code, namely the minimum weight w such that codewords of the dual of weight at most w span the dual code. It is easy to show that to achieve any positive ϵ (even going to 0 as $n \to \infty$) a (k,ϵ)-local

tester must make at least w queries (on some random choices), where w is the spanning weight of the dual. Somewhat surprisingly, the spanning weight of the Reed-Muller code does not seem well-understood. (Some partial understanding comes from [DK00].). Since for a linear code, the spanning weight of its dual code is a lower bound on the query complexity of the code, our result gives new upper bounds on this spanning weight. Specifically, we have

Corollary 1. *Let $q = p^s$ for prime p and positive integer s. Then there exists a constant $c_q \leq 3q^4$ such that for every d and n, the Reed-Muller code* $\mathrm{RM}[n, n(q-1) - d - 1, q]$ *has a spanning weight of at most $c_q \cdot \left(2^{p-1} + p - 1\right)^{(d+1)/(q(p-1))} \cdot q^{(d+1)/q} \leq 3q^4 \cdot (3q)^{(d+1)/q}$.*

1.3 Qualitative Description and Techniques

Our tester differs from previous ones in some qualitative ways. All previously analyzed testers for low-degree testing roughly worked as follows: They picked a large enough dimension t (depending on q and d, but not n) and verified that the function to be tested was a degree d polynomial on a random t-dimensional affine subspace. The final aspect was verified by querying the function on the entire t-dimensional space, thus leading to a query complexity of q^t. The minimal choice of the dimension t that allows this test to detect functions that are not degree d polynomials with positive probability is termed the "testing dimension" (see, for instance, [HSS11]), and this quantity is well-understood, and equals $t_{q,d} = \lceil (d+1)/(q - q/p) \rceil$.

Any improvement to the query complexity of the test above requires two features: (1) For some choices of the tester's randomness, the set of queried points should span a $t_{q,d}$ dimensional space. (2) For all choices of the tester's randomness, it should make $o(q^{t_{q,d}})$ queries. Finding such a useful subset of $\mathbb{F}_q^n$ turns out to be a non-trivial task. The fortunate occurence that provides the basis for our tester is that such sets of points can indeed be found, and even (in retrospect) systematically.

To illustrate the central idea, consider the setting of $n = 2$, $d = q - 1$ and $q = 2^s$ for some large s. While the naive test would query the given function $f : \mathbb{F}_q^2 \to \mathbb{F}_q$ at all q^2 points, we wish to query only $O(q)$ points. Our test, for this simple setting is the following: We pick a random affine-transformation $T : \mathbb{F}_q^2 \to \mathbb{F}_q^2$ and test that the function $f \circ T$ has a zero "inner-product" with the function $g : \mathbb{F}_q^2 \to \mathbb{F}_q$ given by $g(x, y) = \frac{1}{y}((x + y)^{q-1} - x^{q-1})$. Here "inner-product" is simply the quantity $\sum_{\alpha,\beta \in \mathbb{F}_q} (f \circ T)(\alpha, \beta) g(\alpha, \beta)$. It can be verified that the function g is zero very often and indeed takes on non-zero values on at most $3q = O(q)$ points in $\mathbb{F}_q^2$. So querying $f(\alpha, \beta)$ at these $O(q)$ points suffices. The more interesting question is: Why is this test complete and sound?

Completeness is also easy to verify. It can be verified, by some simple manipulations that any monomial of the form $x^i y^j$ with $i + j < q$ has a zero inner product with g and by linearity of the test it follows that all polynomials of total degree at most d have a zero inner product with g. Since the degree of functions

is preserved under affine-transformations, it then follows that $f \circ T$ also has zero inner product with g for every polynomial f of total degree at most d.

Finally, we turn to the soundness. Here we appeal to the emerging body of work on affine-invariant linear properties (linear properties that are preserved under affine-transformations), which allows us to focus on very specific monomials and to verify that their inner product with g is non-zero. In particular, we use a "monomial extraction" lemma (from [KS07]) which allows us to focus on the behavior of our tests only on monomials, as opposed to general polynomials. Further the theory also allows us to focus on specific monomials due to a "monomial spread" lemma which we use to prove that every affine-invariant family which contains some monomials of degree greater than d also contains some canonical monomials of degree slightly larger than d. In the special case of polynomials of degree at most $q-1$, these lemmas allow us to focus on only bivariate monomials of degree q, namely the monomials $x^i y^{q-i}$ for $1 \leq i \leq q-1$ and for these monomials one can again verify that their inner product with g is non-zero. Using the general methods in the theory of affine-invariant property testing, one can conclude that all polynomials of degree greater than d are rejected with positive probability.

Extending the above result to the general case turns out relatively clean, again using methods from the study of testing of affine-invariant linear properties. The extension to general n is immediate. Extending to other degrees involves some intuitive ways of combining tests, with analysis that get simplified by the emerging theory. These combinations yield the query complexity of roughly $(3q)^{(d+1)/q}$. We however attempt to reduce the constant in front of q in the base of this expression and manage to get an expression that tends to 2 when p goes to infinity. In order to do so we abstract the function g as being the derivative of the function x^{q-1} in direction y, and extend it to use iterative derivatives. This yields the best tests we give in the paper.

Organization. In Section 2 we introduce some of the standard background material from the study of affine-invariant linear properties and use the theory to provide restatements of our problem. In Section 3 we introduce the main novelty of our work, which provides a restricted version of our test while achieving significant savings over standard tests. In the full version of this paper [RS12] we show how to build on the test from Section 3 to get a tester for the general case. Due to space limitations many proofs are omitted from this version. They may also be found in the full version.

2 Background and Restatement of Problem

We start by introducing some of the background material that leads to some reformulations of the main theorem we wish to prove. We first introduce the notions of "constraints" and "(single-orbit) characterizations", which leads to a first reformulation of our main theorem (see Theorem 3). We then give some sufficient conditions to recognize such characterizations, and this leads to a second reformulation of our main theorem (see Theorem 4).

2.1 Single-Orbit Characterizations

In this section we use the fact that Reed-Muller codes form a "linear, affine-invariant property". We recall these notions first. Given a finite field $\mathbb{F}_q$ a property is a set of functions $\mathcal{F}$ mapping $\mathbb{F}_q^n$ to $\mathbb{F}_q$. The property is said to be *linear* if it is an $\mathbb{F}_q$-vector space, i.e., $\forall f, g \in \mathcal{F}$ and $\alpha \in \mathbb{F}_q$ we have $\alpha f + g \in \mathcal{F}$. The property is said to be *affine-invariant* if it is invariant under affine-transformations of the domain, i.e., $\forall f \in \mathcal{F}$ it is the case that $f \circ T$ is also in $\mathcal{F}$ for every affine-transformation $T : \mathbb{F}_q^n \to \mathbb{F}_q^n$ given by $T(x) = A \cdot x + \beta$ for $A \in \mathbb{F}_q^{n \times n}$, $\beta \in \mathbb{F}_q^n$.[1] It can be easily verified that $\mathrm{RM}[n, d, q]$ is linear and affine-invariant for every n, d, q.

The main tool used so far for constructing testers for affine-invarinat linear properties is a structural theorem which shows that every linear affine-invariant property that is k-single characterizable is also k-locally testable. In order to describe the notion of single-orbit characterizability we start with a couple of definitions.

Definition 1 (k-constraint, k-characterization). *A* k-constraint $C = (\overline{\alpha}, \{\overline{\lambda}_i\}_{i=1}^r)$ *on* $\{\mathbb{F}_q^n \to \mathbb{F}_q\}$ *is given by a vector* $\overline{\alpha} = (\alpha_1, \ldots, \alpha_k) \in (\mathbb{F}_q^n)^k$ *together with* r *vectors* $\overline{\lambda}_i = (\lambda_{i,1}, \ldots \lambda_{i,k}) \in \mathbb{F}_q^k$ *for* $1 \le i \le r$. *We say that the constraint* C accepts *a function* $f : \mathbb{F}_{q^n} \to \mathbb{F}_q$ *if* $\sum_{j=1}^k \lambda_{i,j} f(\alpha_j) = 0$ *for all* $1 \le i \le r$. *Otherwise we say that* C rejects f.

Let $\mathcal{F} \subseteq \{\mathbb{F}_{q^n} \to \mathbb{F}_q\}$ *be a linear property. A* k-characterization *of* $\mathcal{F}$ *is a collection of* k*-constraints* $C_1, \ldots, C_m$ *on* $\{\mathbb{F}_q^n \to \mathbb{F}_q\}$ *such that* $f \in \mathcal{F}$ *if and only if* C_j *accepts* f, *for every* $j \in \{1, \ldots, m\}$.

It is well-known [BHR05] that every k-locally testable linear property must have a k-characterization. In the case of affine-invariant linear families some special characterizations are known to lead to k-testability. We describe these special characterizations next.

Definition 2 (k-single-orbit characterization). *Let* $C = (\overline{\alpha}, \{\overline{\lambda}_i\}_{i=1}^r)$ *be a* k*-constraint on* $\{\mathbb{F}_q^n \to \mathbb{F}_q\}$. *The* orbit *of* C *under the set of affine-transformations is the set of* k*-constraints* $\{T \circ C\}_T = \left\{ \left((T(\alpha_1), \ldots, T(\alpha_k)), \{\overline{\lambda}_i\}_{i=1}^r \right) \mid T : \mathbb{F}_q^n \to \mathbb{F}_q^n \text{ is an affine-transformation} \right\}$. *We say that* C *is a* k-single-orbit characterization *of* $\mathcal{F}$ *if the orbit of* C *forms a* k*-characterization of* $\mathcal{F}$.

The following theorem, due to Kaufman and Sudan [KS07], says that k-single-orbit characterization implies local testability.

Theorem 2 (Single-orbit characterizability implies local testability, [KS07, Lemma 2.9]). *Let* $\mathcal{F} \subseteq \{\mathbb{F}_q^n \to \mathbb{F}_q\}$ *be an affine-invariant linear family. If* $\mathcal{F}$ *has a* k*-single-orbit characterization, then* $\mathcal{F}$ *has a* $(k, \Omega(1/k^2))$*-local tester.*

[1] We note that as in [KS07] we do not require A to be non-singular. Thus the affine-transformations we consider are not necessarily permutations from $\mathbb{F}_q^n$ to $\mathbb{F}_q^n$.

In view of the above theorem, it suffices to find a single-orbit characterization of $\mathrm{RM}[n,d,q]$ to test it. The following theorem, which we prove in the rest of this paper, thus immediately implies Theorem 1.

Theorem 3. *Let $q = p^s$ for prime p, and let n, d be arbitrary positive integers. Then the Reed-Muller code $\mathrm{RM}[n,d,q]$ has a k-single-orbit characterization for $k \leq c_q \cdot \left(2^{p-1}+p-1\right)^{(d+1)/(q(p-1))} \cdot q^{(d+1)/q}$ where $c_q \leq 3q^4$.*

2.2 Constraints vs. Monomials

One of the main simplifications derived from the study of affine-invariant linear properties is that it suffices to analyze the performance of constraints on "monomials" as opposed to general polynomials. This allows us to rephrase our target (a single-orbit characterization of $\mathrm{RM}[n,d,q]$) in somewhat simpler terms. Below we describe some of the essential notions, namely the "degree set", the "border set" and the relationship of these to single-orbit characterizations. This leads to a further reformulation of our main theorem as Theorem 4. Variations of most of the results and notions presented in this section appeared in previous works [KS07, GKS09, BS11, BGM+11]. In all the above works, with the exception of [KS07], the notions were specialized to the case of univariate funcions mapping $\mathbb{F}_{q^n}$ to $\mathbb{F}_q$ that are invariant over the set of affine-transformations over $\mathbb{F}_{q^n}$. In this work we focus on these notions in the context of affine-invariant linear properties over the domain $\mathbb{F}_q^n$.

Let $\mathcal{F} \subseteq \{\mathbb{F}_q^n \to \mathbb{F}_q\}$ be a linear affine-invariant family of functions. Note that every member of $\{\mathbb{F}_q^n \to \mathbb{F}_q\}$ can be written uniquely as a polynomial in $\mathbb{F}_q[x_1, x_2, \ldots, x_n]$ of degree at most $q-1$ in each variable. For a monomial $\prod_{i=1}^n x_i^{d_i}$ over n variables, we define its degree to be the vector $\overline{d} = (d_1, d_2, \ldots, d_n)$ and we define its *total degree* to be $\sum_{i=1}^n d_i$. For a function $f : \mathbb{F}_q^n \to \mathbb{F}_q$ we denote its *support*, denoted $\mathrm{supp}(f)$, to be the set degrees in the support of the associated polynomial. I.e., $\mathrm{supp}(f) = \{\overline{d} \in \{0,\ldots,q-1\}^n | c_{\overline{d}} \neq 0\}$ where $f(x) = \sum_{\overline{d}} c_{\overline{d}} x^{\overline{d}}$. The *degree set* $\mathrm{Deg}(\mathcal{F})$ of $\mathcal{F}$ is simply the union of the supports of the functions in $\mathcal{F}$, i.e., $\mathrm{Deg}(\mathcal{F}) = \cup_{f\in\mathcal{F}} \mathrm{supp}(f)$.

While the degree set of the Reed-Muller codes are natural to study, they are also natural in more general contexts. The following lemma from [KS07] says that every affine-invariant linear property from $\mathbb{F}_q^n$ to $\mathbb{F}_q$ is uniquely determined by its degree set.

Lemma 1 (Monomial extraction lemma, [KS07, Lemma 4.2]). *Let $\mathcal{F} \subseteq \left\{\mathbb{F}_q^n \to \mathbb{F}_q\right\}$ be an affine-invariant linear property. Then $\mathcal{F}$ has a monomial basis, that is, $\mathcal{F}$ is the set of all polynomials supported on monomials of the form $x^{\overline{d}}$ where $\overline{d} \in \mathrm{Deg}(\mathcal{F})$.*[2]

[2] Our language is somewhat different from that of [KS07]. After translation, their lemma says that all monomials $x^{\overline{d}}$ are contained in $\mathcal{F}$. The other direction saying $\mathcal{F}$ is contained in the span of such monomials is immediate from the definition of $\mathrm{Deg}(\mathcal{F})$.

One main structural feature of the degree sets of affine-invariant linear properties is that they are *p-shadow-closed*. Before giving the definition of a shadow-closed set of degrees we need to introduce a bit of notation. For a pair of integers a, b let $a = \sum_j a_j p^j$, $b = \sum_j b_j p^j$ be their base-p representation, respectively. We say that b is in the *p-shadow* of a, and denote this $b \leq_p a$, if $b_j \leq a_j$ for all j. For a pair of integer vectors $\overline{d} = (d_1, d_2, \ldots, d_n)$, $\overline{e} = (e_1, e_2, \ldots, e_n)$ we say that $\overline{e} \leq_p \overline{d}$ if $e_i \leq_p d_i$ for every i.

Definition 3 (Shadow-closed set of degrees). *For a vector of integers* $\overline{d} = (d_1, d_2, \ldots, d_n)$ *of length* n*, the* p-shadow *of* $\overline{d}$ *is the set* $\mathrm{Shadow}_p(\overline{d}) = \{\overline{e} = (e_1, e_2, \ldots, e_n) \mid \overline{e} \leq_p \overline{d}\}$*. For a subset* S *of integer vectors of length* n *we let* $\mathrm{Shadow}_p(S) = \bigcup_{\overline{d} \in S} \mathrm{Shadow}_p(\overline{d})$*. Finally, we say that* S *is* p-Shadow-closed *if* $\mathrm{Shadow}_p(S) = S$*.*

Lemma 4.6 in [KS07] says that the degree set of every affine-invariant linear property over $\mathbb{F}_q^n$ is p-shadow-closed. This motivates the notion of a "border" set, the set of minimal elements (under $\leq_p$) that are not in $\mathrm{Deg}(\mathcal{F})$.

Definition 4 (Border). *For an affine-invariant linear family* $\mathcal{F} \subseteq \left\{\mathbb{F}_q^n \to \mathbb{F}_q\right\}$*, its* border set*, denoted* $\mathrm{Border}(\mathcal{F})$*, is the set*

$$\mathit{Border}(\mathcal{F}) = \{\overline{e} \in \{0, \ldots, q-1\}^n | \overline{e} \notin \mathrm{Deg}(\mathcal{F}) \text{ but } \forall \overline{e}' \leq_p \overline{e}, \overline{e}' \neq \overline{e}, \overline{e}' \in \mathrm{Deg}(\mathcal{F})\}.$$

The relationship between the degree set and the border set of an affine-invariant linear family and single-orbit characterizability is given by the following lemma. This lemma says that for an affine-invariant linear family, in order to establish k-single-orbit characterizability it suffices to exhibit a k-constraint whose orbit accepts all monomials of the form $x^{\overline{d}}$ for $\overline{d} \in \mathrm{Deg}(\mathcal{F})$ and rejects all monomials of the form $x^{\overline{b}}$ for $\overline{b} \in \mathrm{Border}(\mathcal{F})$. It is similar in spirit to Lemma 3.2 of [BGM+11] which shows that a similar result holds for affine-invariant linear properties over $\mathbb{F}_{q^n}$.

Lemma 2. *Let* $\mathcal{F} \subseteq \{\mathbb{F}_q^n \to \mathbb{F}_q\}$ *be an affine-invariant linear property and let* C *be a constraint. Then* C *is a single-orbit characterization of* $\mathcal{F}$ *if the orbit of* C *accepts every monomial* $x^{\overline{d}}$ *for* $\overline{d} \in \mathrm{Deg}(\mathcal{F})$ *and rejects every monomial* $x^{\overline{b}}$ *for* $\overline{b} \in \mathrm{Border}(\mathcal{F})$*.*

Proof omitted in this version.

In order to describe the border of the Reed-Muller family we shall use the following definition.

Definition 5. *For integer* d*, let* $d_0, d_1, \ldots,$ *be its expansion in base-*p*, i.e.,* d_j*'s satisfy* $0 \leq d_j < p$ *and* $d = \sum_{j=0}^{\infty} d_j p^j$*. Let* $b_i(d) = p^i + \sum_{j=i}^{\infty} d_j p^j$*.*

Note that $b_i(d) > d$ for every i and conversely, for every integer $e > d$ there exists an i such that $b_i(d) \leq_p e$. The $b_i(d)$'s are useful in describing the border monomials of the Reed-Muller family, as formalized below.

Proposition 1. *For every n, d, q, where $q = p^s$ for a prime p, we have*

$$\mathrm{Deg}(\mathrm{RM}[n,d,q]) = \left\{ \overline{d} = (d_1, \ldots, d_n) \in \{0, \ldots, q-1\}^n \mid \sum_{j=1}^{n} d_j \leq d \right\} \text{ and}$$

$$\mathrm{Border}(\mathrm{RM}[n,d,q]) \subseteq$$

$$\left\{ \overline{e} = (e_1, \ldots, e_n) \in \{0, \ldots, q-1\}^n \mid \sum_{j=1}^{n} e_j = b_i(d) \text{ for some } 0 \leq i \leq s \right\}.$$

Proof omitted.

Combining Lemma 2 and Proposition 1 we have that Theorem 3 follows immediately from Theorem 4 below.

Theorem 4. *Let $q = p^s$ for a prime p. Then there exists a k-constraint C whose orbit accepts all monomials of total degree at most d and rejects all monomials of total degree $b_i(d)$ for $0 \leq i \leq s$, for $k \leq 3q^4 \cdot \left(2^{p-1} + p - 1\right)^{(d+1)/(q(p-1))} \cdot q^{(d+1)/q}$.*

The rest of this paper will be devoted to proving Theorem 4.

3 Canonical Monomials and a New Constraint

In this section we introduce the notion of "canonical monomials" of a given degree — very simplified monomials that appear in every affine-invariant linear property containing monomials of a given degree. We then give a constraint that rejects canonical monomials of some special degrees, while accepting all monomials of lower degrees. In the full version of this paper [RS12], we show how to use this to build a constraint whose orbit accepts all monomials of total degree at most d while rejecting all monomials of total degree $b_i(d)$, which suffices to get Theorem 4.

Definition 6 (Canonical monomials). *Let $q = p^s$ for a prime p. The canonical monomial of (total) degree d over $\mathbb{F}_q$ is the monomial $\prod_{i=1}^{\ell} x_i^{d_i}$ which satisfies $\sum_{i=1}^{\ell} d_i = d$, $d_i = q - q/p$ for all $2 \leq i \leq \ell$, $0 \leq d_1 \leq q-1$ and $d_1 + q - q/p > q - 1$.*

We note that [HSS11] used a different canonical monomial (cf. Definition 4.1., [HSS11]) for the construction of their improved tester for the Reed-Muller codes. Our different choice of canonical monomial is needed to construct single-orbit characterizations which improve on those given in [HSS11] in terms of the number of queries. The main property of the canonical monomial, that we will use in the full version of tihs paper to prove Theorem 4 is that every affine-invariant linear family that contains any monomial of total degree d also contains the canonical monomial of degree d. This will imply in turn that if we can find constraints that *reject* this canonical monomial their orbit will reject every monomial of total degree d.

3.1 A New Constraint on Monomials of Total Degree $< p(q - q/p)$

The main technical novelty in our paper is a k-constraint C that accepts all monomials of total degree strictly less than $p(q-q/p)$ in p variables but rejects the canonical monomial of degree $p(q-q/p)$ (note that the latter monomial also has p variables) for $k = (2^{p-1}+p-1)q^{p-1}$. We state the lemma below and devote the rest of this section to proving this lemma.

Lemma 3 (Main technical lemma). *For every q which is a power of a prime p there exists a k-constraint C which accepts all monomials of total degree smaller than $p(q-q/p)$ in p variables and rejects the canonical monomial (in p variables) of degree $p(q-q/p)$ over $\mathbb{F}_q$, where $k = (2^{p-1}+p-1)q^{p-1}$.*

It will be convenient for us to represent the constraint C as a p-variate polynomial over $\mathbb{F}_q$. More precisely, suppose that $g(x)$ is a p-variate polynomial $g(x) \in \mathbb{F}_q[x_1, x_2, \ldots, x_p]$ that is non-zero on at most k points in $\mathbb{F}_q^p$. We associate with $g(x)$ the k-constraint $C = (\overline{\alpha}, \overline{\lambda})$, $\overline{\alpha} = (\alpha_1, \ldots, \alpha_k) \in (\mathbb{F}_q^p)^k$, $\overline{\lambda} = (\lambda_1, \ldots, \lambda_k) \in \mathbb{F}_q^k$, where the vector $\overline{\alpha}$ consists of all points in $\mathbb{F}_q^p$ on which $g(x)$ is non-zero and $\lambda_j = g(\alpha_j)$ for all $1 \le j \le k$. Clearly, for every function $f : \mathbb{F}_q^p \to \mathbb{F}_q$ it holds that

$$\sum_{j=1}^{k} \lambda_j f(\alpha_j) = \sum_{\beta_1,\ldots,\beta_p \in \mathbb{F}_q} g(\beta_1, \ldots, \beta_p) \cdot f(\beta_1, \ldots, \beta_p) \tag{1}$$

Thus we reduce the task of finding a k-constraint which accepts all monomials of total degree smaller than $p(q-q/p)$ and rejects the canonical monomial of degree $p(q-q/p)$ to the task of finding a p-variate polynomial $g(x) \in \mathbb{F}_q[x_1, x_2, \ldots, x_p]$ with at most k non-zero points in $\mathbb{F}_q^p$ such that $\sum_{\beta_1,\ldots,\beta_p \in \mathbb{F}_q} g(\beta_1, \ldots, \beta_p) \cdot M(\beta_1, \ldots, \beta_p) = 0$ for every monomial in p variables of total degree smaller than $p(q-q/p)$ and $\sum_{\beta_1,\ldots,\beta_p \in \mathbb{F}_q} g(\beta_1, \ldots, \beta_p) \cdot M(\beta_1, \ldots, \beta_p) \neq 0$ when $M(x)$ is the canonical monomial of degree $p(q-q/p)$.

We start by describing a polynomial $P(x)$ that will satisfy the conditions we expect in g above. The best way to describe this polynomial is via the notion of *directional derivatives*. Let $f : \mathbb{F}_q \to \mathbb{F}_q$ be a function. Define the derivative of f in direction $y \in \mathbb{F}_q$ as $f_y(x) = f(x+y) - f(x)$. Define the iterated derivatives as

$$f_{y_1,\ldots,y_d}(x) = (f_{y_1,\ldots,y_{d-1}})_{y_d}(x) = \sum_{I \subseteq [d]} (-1)^{|I|+1} f\left(x + \sum_{i \in I} y_i\right).$$

Let $f(x)$ be the polynomial $f(x) = x_p^{q-1}$. Our polynomial $P(x)$ will be defined as follows.

$$P(x) = \frac{f_{x_1,\ldots,x_{p-1}}(x_p)}{x_1 \cdots x_{p-1}} = \frac{\sum_{I \subseteq [p-1]} (-1)^{|I|+1} (x_p + \sum_{i \in I} x_i)^{q-1}}{x_1 \cdots x_{p-1}}. \tag{2}$$

To see that $P(x)$ is indeed a polynomial we need to show that $f_{x_1,\ldots,x_{p-1}}(x_p)$ is divisible by $x_1 \cdots x_{p-1}$. We omit the proof here.

In order to prove our main technical Lemma 3 it suffices to show that the number of non-zero points of $P(x)$ in $\mathbb{F}_q^p$ is at most $(2^{p-1}+p-1)q^{p-1}$, that it accepts all monomials in p variables of total degree smaller $p(q-q/p)$, and that it rejects the canonical monomial of degree $p(q-q/p)$. We assert these three claims in Lemmas 4, 5 and 6 below, respectively. Given these three lemmas our main technical Lemma 3 is immediate. We start with bounding the number of non-zeros of $P(x)$.

Lemma 4. *The number of non-zero points of $P(x)$ in $\mathbb{F}_q^p$ is at most $(2^{p-1}+p-1)q^{p-1}$.*

Lemma 5. *Let C be the constraint associated with $P(x)$. Then C accepts all monomials in p variables of total degree smaller than $p(q-q/p)$.*

Lemma 6. *Let C be the constraint associated with $P(x)$. Then C rejects the canonical monomial of degree $p(q-q/p)$ over $\mathbb{F}_q$.*

Given Lemmas 4, 5 and 6 the proof of Lemma 3 is immediate.

Proof (Proof of Lemma 3). Let $P(x)$ be the polynomial given in (2), and let C be the constraint on $\left\{\mathbb{F}_q^p \to \mathbb{F}_q\right\}$ associated with $P(x)$. From Lemma 4 we have that the number of non-zero points of $P(x)$ in $\mathbb{F}_q^p$ is at most $(2^{p-1}+p-1)q^{p-1}$, and hence C is a $\left((2^{p-1}+p-1)q^{p-1}\right)$-constraint. Lemma 5 implies that C accepts all monomials of total degree smaller than $p(q-q/p)$, while Lemma 6 implies that C rejects the canonical monomial of degree $p(q-q/p)$.

Acknowledgements. We would like to thank Amir Shpilka for suggesting that our tests are related to directional derivatives.

References

[AKK+05] Alon, N., Kaufman, T., Krivelevich, M., Litsyn, S., Ron, D.: Testing Reed-Muller codes. IEEE Transactions on Information Theory 51(11), 4032–4039 (2005)

[ALM+98] 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)

[BGH+11] Barak, B., Gopalan, P., Håstad, J., Meka, R., Raghavendra, P., Steurer, D.: Making the long code shorter, with applications to the unique games conjecture. CoRR, abs/1111.0405 (2011)

[BGM+11] Ben-Sasson, E., Grigorescu, E., Maatouk, G., Shpilka, A., Sudan, M.: On Sums of Locally Testable Affine Invariant Properties. In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) APPROX/RANDOM 2011. LNCS, vol. 6845, pp. 400–411. Springer, Heidelberg (2011)

[BHR05] Ben-Sasson, E., Harsha, P., Raskhodnikova, S.: Some 3CNF properties are hard to test. SICOMP: SIAM Journal on Computing 35 (2005)

[BKS+10] Bhattacharyya, A., Kopparty, S., Schoenebeck, G., Sudan, M., Zuckerman, D.: Optimal testing of Reed-Muller codes. In: FOCS, pp. 488–497. IEEE Computer Society (2010)

[BS11] Ben-Sasson, E., Sudan, M.: Limits on the Rate of Locally Testable Affine-Invariant Codes. In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) APPROX/RANDOM 2011. LNCS, vol. 6845, pp. 412–423. Springer, Heidelberg (2011)

[DK00] Ding, P., Key, J.D.: Minimum-weight codewords as generators of generalized Reed-Muller codes. IEEE Transactions on Information Theory 46(6), 2152–2158 (2000)

[FS95] Friedl, K., Sudan, M.: Some improvements to total degree tests. In: Proceedings of the 3rd Annual Israel Symposium on Theory of Computing and Systems, January 4-6, pp. 190–198. IEEE Computer Society, Washington, DC (1995) Corrected version available online at, http://people.csail.mit.edu/madhu/papers/friedl.ps

[GKS09] 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)

[HSS11] Haramaty, E., Shpilka, A., Sudan, M.: Optimal testing of multivariate polynomials over small prime fields. In: Ostrovsky, R. (ed.) FOCS, pp. 629–637. IEEE (2011)

[JPRZ09] 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)

[KR06] Kaufman, T., Ron, D.: Testing polynomials over general fields. SIAM Journal of Computing 36(3), 779–802 (2006)

[KS07] Kaufman, T., Sudan, M.: Algebraic property testing: The role of invariance. Electronic Colloquium on Computational Complexity (ECCC) 14(111) (2007)

[RS96] Rubinfeld, R., Sudan, M.: Robust characterizations of polynomials with applications to program testing. SIAM Journal on Computing 25(2), 252–271 (1996)

[RS12] Ron-Zewi, N., Sudan, M.: A new upper bound on the query complexity for testing generalized Reed-Muller codes. Electronic Colloquium on Computational Complexity (ECCC) 19, 46 (2012)

[VW08] Viola, E., Wigderson, A.: Norms, xor lemmas, and lower bounds for polynomials and protocols. Theory of Computing 4(1), 137–168 (2008)

A Combination of Testability and Decodability by Tensor Products*

Michael Viderman

Computer Science Department
Technion — Israel Institute of Technology
Haifa, 32000, Israel
viderman@cs.technion.ac.il

Abstract. Ben-Sasson and Sudan (RSA 2006) showed that taking the repeated tensor product of linear codes with very large distance results in codes that are locally testable. Due to the large distance requirement the associated tensor products could be applied only over sufficiently large fields.

In this paper we improve the result of Ben-Sasson and Sudan and show that for *any* linear codes the associated tensor products are locally testable.

Moreover, a combination of our result with the result of Spielman (IEEE IT, 1996) implies a construction of linear codes (over any field) that combine the following properties:

- have constant rate and constant relative distance;
- have blocklength n and are testable with n^ϵ queries, for any constant $\epsilon > 0$;
- linear time encodable and linear-time decodable from a constant fraction of errors.

Furthermore, a combination of our result with the result of Guruswami et al. (STOC 2009) implies a similar corollary for list-decodable codes.

Keywords: Tensor products, locally testable codes, locally correctable codes, efficient decoding.

1 Introduction

Over the last decades coding theory and complexity theory have benefited from numerous interesting interconnections. Recent major achievements in complexity theory, e.g., showing IP = PSPACE [38,45,46] and giving a PCP characterization of NP [3,4] have strongly relied on connections with coding theory either explicitly or implicitly.

Most of the well-studied and practically used codes are linear codes. A linear code $C \subseteq \mathbf{F}^n$ is a linear subspace over the field $\mathbf{F}$, where n is called the blocklength of C and $\dim(C)$ denotes the dimension of the code. The rate of the code is

* The research was partially supported by the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 240258 and by grant number 2006104 by the US-Israel Binational Science Foundation.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 651–662, 2012.
© Springer-Verlag Berlin Heidelberg 2012

defined by $rate(C) = \frac{\dim(C)}{n}$. We define the distance between two words $x, y \in \mathbf{F}^n$ to be $\Delta(x,y) = |\{i \mid x_i \neq y_i\}|$ and the relative distance to be $\delta(x,y) = \frac{\Delta(x,y)}{n}$. The distance of the code C is defined by $\Delta(C) = \min_{x \neq y \in C} \Delta(x,y)$ and its the relative distance is denoted $\delta(C) = \frac{\Delta(C)}{n}$. Typically, one is interested in the codes whose distance is linear in the blocklength.

The central algorithmic problem in coding theory is the explicit construction of error-correcting codes with best possible parameters together with fast encoding and decoding algorithms. These features were proved to be useful also in cryptography and computational complexity (see e.g., [50, Section 1]).

Besides the efficient encoding/decoding algorithms there are interesting well-studied properties: local testing and local decoding (correction). The combination of these properties is highly useful, e.g., PCPs based on the Hadamard code [3] relied on the fact that the Hadamard code is testable with 3 queries [35] and locally decodable (correctable) with 2 queries.

Given the fact that error-correcting codes play an important role in complexity theory, and in particular, in different interactive protocols (see e.g., [7]), it might be helpful to develop a general scheme for constructing error-correcting codes that combine several different properties. E.g., it might be helpful to have high-rate codes which combine such properties as local testing, efficient encoding and decoding from a constant fraction of errors. This is what we do in this paper. In the rest of the introduction we provide a brief background and explain our contribution.

Locally Testable Codes. Locally testable codes (LTCs) are error correcting codes that have a tester, which is a randomized algorithm with oracle access to the received word x. The tester reads a sublinear amount of information from x and based on this "local view" decides if $x \in C$ or not. It should accept codewords with probability one, and reject words that are far (in Hamming distance) from the code with noticeable probability.

LTCs were implicit already in [6] (cf. [22, Sec. 2.4]) and they were explicitly studied by Goldreich and Sudan [24]. By now several different constructions of LTCs are known including codes based on low-degree polynomials over finite fields [2,35,3], constructions based on PCPs of proximity/assignment testers [9,19],[1] and sparse random linear codes [15], [29], [31]. In this paper we study a different family of LTC constructions, namely, *tensor codes*. Given two linear error correcting codes $C \subseteq \mathbf{F}^{n_1}, R \subseteq \mathbf{F}^{n_2}$ over a finite field $\mathbf{F}$, we define their *tensor product* to be the subspace $R \otimes C \subseteq \mathbf{F}^{n_1 \times n_2}$ consisting of $n_1 \times n_2$ matrices M with entries in $\mathbf{F}$ having the property that every row of M is a codeword of R and every column of M is a codeword of C. In this case, we say that C and R are *base-codes*. If $C = R$ we use C^2 to denote $C \otimes C$ and for $i > 2$ define $C^i = C \otimes C^{i-1}$. Note that the blocklength of C^i is n_1^i.

[1] As was pointed out in [24], not all PCP constructions are known to yield LTCs, but some of them (e.g., PCPs of proximity/assignment testers) can be adapted to yield LTCs.

Recently, tensor products were used to construct new families of LTCs [12,37], new families of list-decodable codes [25], and to give an alternative proof [38][2] for IP=PSPACE theorem of [45,46].

Ben-Sasson and Sudan [12] suggested to use tensor product codes as a means to construct LTCs combinatorially. Let $C \subseteq \mathbf{F}^{n_1}$ be a linear code and let us consider the following approach. Suppose that the task is to test whether an input word $M \in \mathbf{F}^{n_1 \times n_1}$ belongs to C^2, where M is far from C^2. One could expect that in this case the typical row/column of M is far from C, and hence the tester for C^2 can choose a random row (or column) of M. Then this selected row/column could be tested on being in C. However, as was shown in [51,23,17] this approach fails in general and is known to work only under assumptions that C has some non-trivial properties [20,13,14] (see also [36]).

In spite of this fact, Ben-Sasson and Sudan [12] showed that taking the repeated tensor products of any code $C \subseteq \mathbf{F}^n$ with sufficiently large distance results in a locally testable code with sublinear query complexity. Although it was not explicitly stated in [12], it follows that [12, Theorem 2.6] gives the following result.

Theorem 1 (Informal). *For every $\epsilon > 0$ there exists a sufficiently large field $\mathbf{F} = \mathbf{F}(\epsilon)$ such that letting $m = \lceil \frac{2}{\epsilon} \rceil$ for every $C \subseteq \mathbf{F}^n$, if $\left(\frac{\Delta(C)-1}{n}\right)^m \geq \frac{7}{8}$ then C^m is testable with N^ϵ queries, where $N = n^m$ is the blocklength of C^m. The rejection probability of the tester depends on m.*

It remained unclear if the assumption about the very large distance of the base codes is necessary. Moreover, the requirement on the distance of the base code ($\Delta(C)$) is dependent on the number of tensor products (m) one should apply. Note that for smaller query complexity (relative to the blocklength) more tensor product operations should be applied. Thus the distance of the base code must be increased when the number of queries is decreased. We notice also that the assumption of larger $\Delta(C)$ implies a larger underlying field $\mathbf{F}$. As a consequence, a similar theorem to Theorem 1 could not be argued for a fixed field, like the binary field.

We show that no assumptions about the base codes (or underlying fields) are needed. I.e., we prove the following result (stated formally in Theorem 3).

Theorem 2 (Informal). *For every $\epsilon > 0$ and for every field $\mathbf{F}$ letting $m = \lceil \frac{2}{\epsilon} \rceil$ it holds that for every $C \subseteq \mathbf{F}^n$ we know that C^m is testable with N^ϵ queries, where $N = n^m$ is the blocklength of C^m. The rejection probability of the tester depends on $\Delta(C)$ and m.*

This contrasts with the previous works on the combinatorial constructions of LTCs due to Ben-Sasson and Sudan [12] and Meir [37] which required very large

[2] Meir [38] showed that the "multiplication" property and the "sum-check" protocol can be designed by tensor products. We consider this surprising, since previously such features were achieved only by low degree polynomials.

base-code distance, and as a consequence required the large field size. Furthermore, our proof is much simpler than the proof provided in [12] and simultaneously we obtain some quantitative improvements in the related parameters.

Efficient encoding and decoding. Let us ask the following natural question. Whether tensor products of codes can be encoded efficiently? It is quite simple to show that if the code C has an efficient (linear time) encoder then C^m has an efficient (linear time) encoder.

Let us turn to the decoding properties of the tensor products, e.g., the natural question here would be whether tensor products of codes preserve the decoding properties provided that the base codes are efficiently decodable. Gopalan et al. [25] showed that tensor products preserve the list-decoding properties, i.e., if C is list-decodable in polynomial time then C^m is list-decodable in polynomial time.[3] Our contribution to this question is as follows. In the full version we show that if C is decodable from a constant fraction of errors in linear time then C^m is decodable from a constant fraction of errors in linear time.

Then, we show (Corollaries 2, 3) that a combination of our results with the results of [48,25] implies the construction of constant-rate codes which are both testable with sublinear query complexity, linear-time encodable and efficiently decodable (or list-decodable) from the constant fraction of errors.

Tensor product of codes preserves the local decoding (correction) properties. Informally, locally decodable codes (LDCs) and locally correctable codes (LCCs) are error-correcting codes that allow to retrieve each message (codeword) bit using a small number of queries even after a constant fraction of it is adversely corrupted. The most famous LDCs (LCCs) include Hadamard and Reed-Muller codes [41]. In theoretical computer science, locally decodable codes have played an important part in the Proof-Checking Revolution [33], [34], [45], [6], [7], [4], [3] as well as in other fundamental results in complexity theory [8], [28], [5], [49], [44].

In Section 3.3 we prove that tensor product of codes preserve the local correction property. That means if C is an LCC with query complexity q then C^2 is an LCC with query complexity q^2. On the one hand, this observation discovers additional families of locally correctable codes and on the other hand, it suggests a simple way to combine two different properties: local correction and local testing.

2 Preliminaries

All codes discussed in this paper are linear. Throughout this paper, we let $[n] = \{1, \ldots, n\}$. For $w \in \mathbf{F}^n$ let $supp(w) = \{i | w_i \neq 0\}$, $|w| = |supp(w)|$ and $\mathrm{w}t(w) = \frac{|w|}{n}$. For $x \in \mathbf{F}^n$ and a linear code $C \subseteq \mathbf{F}^n$, let $\delta(x, C) = \min_{y \in C} \{\delta(x, y)\}$

[3] The main focus in [25] was done on the designing polynomial-time list-decoding algorithms and on the combinatorial bounds for the list-decoding tensor products of codes and interleaved codes.

denote the relative distance of x from the code C. If $\delta(x, C) \geq \epsilon$ we say that x is ϵ-far from C, and otherwise we say that x is ϵ-close to C. We let $C^{\perp} = \{u \in \mathbf{F}^n \mid \forall c \in C : \langle u, c\rangle = 0\}$ be the dual code of C, where $\langle u, c\rangle$ denotes the vector inner product between u and c.

For $w \in \mathbf{F}^n$ and $S = \{j_1, j_2, \ldots, j_m\} \subseteq [n]$, where $j_1 < j_2 < \ldots < j_m$, we let $w|_S = (w_{j_1}, \ldots, w_{j_m})$ be the *restriction* of w to the subset S. We let $C|_S = \{c|_S \mid c \in C\}$ denote the restriction of the code C to the subset S.

2.1 Tensor Product Codes

The definitions appearing here are standard in the literature on tensor-based LTCs (e. g. [20], [12], [37], [14], [51]).

For $x \in \mathbf{F}^I$ and $y \in \mathbf{F}^J$ we let $x \otimes y$ denote the tensor product of x and y (i. e., the matrix M with entries $M_{(i,j)} = x_i \cdot y_j$ where $(i, j) \in I \times J$). Let $R \subseteq \mathbf{F}^I$ and $C \subseteq \mathbf{F}^J$ be linear codes. We define the tensor product code $R \otimes C$ to be the linear space spanned by words $r \otimes c \in \mathbf{F}^{J \times I}$ for $r \in R$ and $c \in C$.

We let $C^1 = C$ and $C^t = C^{t-1} \otimes C$ for $t > 1$. Note by this definition, $C^{2^0} = C$ and $C^{2^t} = C^{2^{t-1}} \otimes C^{2^{t-1}}$ for $t > 0$. We also notice that for a code $C \subseteq \mathbf{F}^n$ and $m \geq 1$ it holds that $rate(C^m) = (rate(C))^m$, $\delta(C^m) = (\delta(C))^m$ and the blocklength of C^m is n^m.

The main drawback of the tensor product operation is that this operation strongly decreases the rate and the distance of the base codes. We refer the reader to [37] which showed how one can use tensor products and avoid the decrease in the distance and the strong decrease in the rate.[4]

2.2 Locally Testable Codes (LTCs)

A *standard q-query tester* for a linear code $C \subseteq \mathbf{F}^n$ is a randomized algorithm that on the input word $w \in \mathbf{F}^n$ picks non-adaptively a subset $I \subseteq [n]$ such that $|I| \leq q$. Then T reads all symbols of $w|_I$ and accepts if $w|_I \in C|_I$, and rejects otherwise (see [10, Theorem 2]). Hence a q-query tester can be associated with a distribution over subsets $I \subseteq [n]$ such that $|I| \leq q$.

Definition 1 (Tester of C and Test View). *A q-*query tester $\mathbf{D}$ *is a distribution $\mathbf{D}$ over subsets $I \subseteq [n]$ such that $|I| \leq q$. Let $w \in \mathbf{F}^n$ (think of the task of testing whether $w \in C$) and let $I \subseteq [n]$ be a subset. We call $w|_I$ the* view *of a tester. If $w|_I \in C|_I$ we say that this view is* consistent *with C, or when C is clear from the context we simply say $w|_I$ is* consistent.

Although the tester in Definition 1 does not output accept or reject, the way a standard tester does, it can be converted to output accept, reject as follows.

[4] Meir [37] demonstrated how one can combine the tensor product operation with two additional operations: random projections and distance amplification. In this way, on the one hand repeated tensor products could be applied, while on the other hand these supplementary operations prevent the distance loss and the strong rate reduction.

Whenever the task is to test whether $w \in C$ and a subset $I \subseteq [n]$ is selected by the tester, the tester can output accept if $w|_I \in C|_I$ and otherwise output reject.

When considering a tensor code $C^m \subseteq \mathbf{F}^{n^m}$, an associated tester will be a distribution over subsets $I \subseteq [n]^m$. We identify $[n^m]$ with $[n]^m$.

Definition 2 (LTCs). *A code $C \subseteq \mathbf{F}^n$ is a (q, ϵ)-LTC if it has a q-query tester $\mathbf{D}$ such that for all $w \in \mathbf{F}^n$, we have $\Pr_{I \sim \mathbf{D}}[w|_I \notin C|_I] \geq \epsilon \cdot \delta(w, C)$.*

Note that given a code $C \subseteq \mathbf{F}^n$, the subset $I \subseteq [n]$ uniquely defines $C|_I$.

3 Main Results

The main result of this paper is stated in Theorem 3. Informally, Theorem 3 says that tensor products of third and higher powers of *any linear code* over any field are locally testable with sublinear query complexity. This theorem is quite powerful and we shall use it later to conclude that tensor products of linear codes can enjoy the combination of local testability and decodability in a new range of parameters, which was not previously known.

Theorem 3 (Main Theorem). *Let $C \subseteq \mathbf{F}^n$ be a linear code and $m \geq 3$ be an integer. Then C^m is a (n^2, α_m)-LTC, where $\alpha_m = \frac{(\delta(C))^{2m}}{18^{\log_{1.5} m}}$. Note that the blocklength of C^m is n^m.*

The proof of Theorem 3 is omitted due to the space limitations. As was mentioned earlier, our analysis is more tight and much simpler than [12].

Remark 1. We would like to point out that for any linear code $C \subseteq \mathbf{F}^n$ it holds that C^2 is a $(n, \frac{1}{2})$-LTC. Note blocklength of C^2 is n^2. So, in this way we can easily obtain a simple construction of an LTC with query complexity equal to the square root of the blocklength. Nevertheless, that is a much more difficult task to obtain a smaller query complexity via tensor products (see e.g., [12], [13], [37] for more information).

Usually, in the areas of locally testable and locally decodable codes the main interest was given to the constant query complexity. Recently, Kopparty et al. [32] showed the construction of locally decodable codes with sublinear query complexity and arbitrary high rate (see [32] for the motivation behind this range of parameters). Since then, the interest in the other range of parameters, and in particular, in sublinear query complexity has increased.

Tensor Products of Codes Can Have Large Distance. As was said in Section 2.1, Meir [37] explained that one of the standard procedures for distance amplification of the code [1] can be combined together with the repeated tensor product operations. He also proved that this procedure preserves the local testability of the underlying code. The simplest way to see this is as follows. Let DistAmp$(\cdot)$ be a procedure that increases the relative distance of the code $C' \subseteq \mathbf{F}_2^n$, e.g.,

from 0.001 to 0.49. I.e., if $\delta(C') \geq 0.001$ then $\delta(\mathrm{DistAmp}(C')) \geq 0.49$. Moreover, it holds that if C' was locally testable then $\mathrm{DistAmp}(C')$ is locally testable, where the query complexity of the code $\mathrm{DistAmp}(C')$ is increased by only a constant factor, independent on the other parameters of the code). It can be readily verified that the distance amplification procedure preserves the encoding time, and in particular, if C' was linear-time encodable then $\mathrm{DistAmp}(C')$ is linear-time encodable. Thus, one can pick any linear-time encodable code C with linear distance, obtain a linear-time encodable LTC $C' = C^{10}$ and then increase its distance by $\mathrm{DistAmp}(C')$. We refer the reader to [37, Section 4.3] for further information about distance amplification procedures and its affect on local testability.

In this paper we won't use any distance amplification procedures and restrict our attention only to the tensor product operation.

We proceed as follows. In Section 3.1 we explain how local testability can be combined with decodability, and in particular, we show that tensor products can be used to provide linear codes of high rate which are locally testable, and at the same time can be efficiently encoded and decoded. Then, in Section 3.2 we show that a combination of Theorem 3 with a result of [25] implies asymptotically good codes that can be encodable in linear time, testable with sublinear query complexity and list-decodable in polynomial time. Finally, in Section 3.3 we argue that tensor products preserve the local decoding (correction) properties. Thus a tensor product of a locally decodable (correctable) code combines both properties: local testing and local decoding (correction).

3.1 Locally Testable and Linear-Time Encodable and Decodable Codes

We continue to investigate the "encoding" and "decoding" properties of tensor products. We show in Corollary 1 a simple construction of LTCs with arbitrary small sublinear query complexity and arbitrary high rate from any linear code with sufficiently high rate.

Corollary 1. *Let* $\mathbf{F}$ *be any field. Let* $C \subseteq \mathbf{F}^n$ *be a linear code and let* $m \geq 3$ *be a constant. Then* $C^m \subseteq \mathbf{F}^{n^m}$ *is a* (n^2, α_m)*-LTC, where* $\alpha_m > 0$ *is a constant that depends only on* m *and* $\delta(C)$*. In particular, for every* $\epsilon > 0$*,* $m = \lceil \frac{2}{\epsilon} \rceil$*,* $N = n^m$ *and* $C \subseteq \mathbf{F}^n$ *such that* $\mathrm{rate}(C) \geq (1-\epsilon)^{1/m}$ *we have that* $C^m \subseteq \mathbf{F}^N$ *is a* (N^ϵ, α)*-LTC and* $\mathrm{rate}(C^m) \geq 1 - \epsilon$*, where* $\alpha > 0$ *is a constant that depends only on* ϵ*. Moreover, if* C *is a linear-time encodable then* C^m *is a linear-time encodable.*

Remark 2. We notice that there are linear error-correcting codes with arbitrary high rate that can be encodable in the linear time (see e.g., [43][5]). Thus Corollary 1 provides a construction of high-rate LTCs with constant relative

[5] This result improves the previous result of [27] and presents the construction of linear codes that lie close to the singleton bound, and have linear time encoding/decoding algorithms.

distance and arbitrary low sublinear query complexity that can be encoded in linear time. Moreover, this construction can be taken over any field. To the best of our knowledge no such results were known before.

We also notice that any simple approach, based on testing of (low-degree) polynomials [2], to achieve the similar result to Corollary 1 fails. In particular, let us consider the testing of Reed-Muller codes of degree d and recall that informally, Reed-Muller codes of degree d can be tested by making $\approx 2^d$ queries. If d is large then the associated codes must be constructed over a very large field (depending on the blocklength of the code), since otherwise cannot have constant relative distance. However, if d is small then the rate of the associated code is very low. It could also be verified that concatenation of a Reed-Muller code with a good binary code does not obtain the combination of properties presented in Corollary 1. Furthermore, the *linear-time* encoding of the codes based on high-degree polynomials is problematic.

Next we turn to the decoding properties of tensor products. Let us first recall the definition of decodable codes.

Definition 3 (Decodable codes). *Let $C \subseteq \mathbf{F}^n$ be a code and let $\alpha < \delta(C)/2$. We say that C is decodable from αn errors in time T if there exists a decoder D_C which on the input word $w \in \mathbf{F}^n$ such that $\delta(w, C) \leq \alpha$ outputs $c \in C$ such that $\delta(w, c) \leq \alpha$ and its running time is upper-bounded by T. If $T = O(n)$ we say that C is decodable in linear time.*

In the full version we show that the tensor product operation preserves the decoding property. In particular, if $C \subseteq \mathbf{F}^n$ is a linear code that is linear time decodable from $\alpha \cdot n$ errors then C^m is linear-time decodable from $\alpha^m \cdot n^m$ errors (for every constant $m \geq 1$).

A combination of our Theorem 3 together with the results of Spielman [48] and Guruswami and Indyk [27] implies the following corollary.

Corollary 2. *For every constant $\epsilon > 0$:*

1. *There exists an (explicit) family of linear error correcting codes $C \subseteq \mathbf{F}_2^N$ (obtained by tensor products on the codes from [48]) that*
   - *have rate and relative distance $\Omega_\epsilon(1)$,*
   - *linear time encodable and linear time decodable from the constant fraction $(\Omega_\epsilon(1))$ of errors,*
   - *are (N^ϵ, α)-LTCs, where $\alpha = \alpha(\epsilon) > 0$ is a constant.*
2. *There exist a field $\mathbf{F}$ and an (explicit) family of linear error correcting codes $C \subseteq \mathbf{F}^N$ (obtained by tensor products on the codes from [43]) that*
   - *have rate at least $1 - \epsilon$ and relative distance $\Omega_\epsilon(1)$,*
   - *linear time encodable and linear time decodable from the constant fraction $(\Omega_{\epsilon(1)})$ of errors,*
   - *are (N^ϵ, α)-LTCs, where $\alpha = \alpha(\epsilon) > 0$ is a constant.*

The proof of Corollary 2 is omitted due to the space limitations. Note that Corollary 2 presents a construction of error-correcting codes that combines local

testability with efficient encoding and decoding algorithms. The difference between these two bullets of the corollary is in the binary field versus a larger field and the constant rate versus arbitrary high rate.

3.2 Locally Testable and List-Decodable Codes

In this section we recall some constructions of list-decodable codes. We start by defining list-decodable codes.

Definition 4 (List-decodable codes). *A code C is a (α, L)-list decodable if for every word $w \in \mathbf{F}^n$ we have $|\{c \in C \mid \delta(c, w) \leq \alpha\}| \leq L$. The code is said to be (α, L)-list decodable in time T if there exists algorithm which on the input $w \in \mathbf{F}^n$ outputs all codewords $c \in C$ such that $\delta(c, w) \leq \alpha$ (at most L codewords).*

Gopalan et al. [25] showed that the list-decodability and the running time of the list-decoder are pretty much preserved in the tensor product operation. In particular, they proved the following theorem, stated in [25, Theorem 5.8], which says that tensor products of linear codes that are list-decodable in polynomial time enjoy this property as well. We use the combination of [25, Theorem 5.8] and Corollary 1 to conclude Corollary 3.

Corollary 3. *Let $\mathbf{F}$ be any field. For every constant $\epsilon > 0$ there exists a code $C \subseteq \mathbf{F}^N$ such that*

- *C is a (N^ϵ, α)-LTC, where $\alpha = \alpha(\epsilon) > 0$ is a constant,*
- *C is encodable in linear time and list-decodable (constant list size) in polynomial time from the constant fraction of errors (depending on ϵ),*
- *$rate(C) = \Omega_\epsilon(1)$ and $\delta(C) = \Omega_\epsilon(1)$.*

The proof of Corollary 3 is omitted.

3.3 Tensor Products Preserve Local Correction Properties

Any linear error correcting code $\mathcal{C} \subseteq \mathbf{F}^n$ can be associated with an encoding function $E_\mathcal{C} : \mathbf{F}^k \to \mathbf{F}^n$ that on the message $x \in \mathbf{F}^k$ returns the codeword $E_\mathcal{C}(x) \in \mathbf{F}^n$. The words $x \in \mathbf{F}^k$ are called messages and the elements x_i for $i \in [k]$ are called message symbols. Informally, locally decodable codes (LDCs) allow to recover each message entry with high probability by reading only a few entries of the codeword even if a constant fraction of it is adversely corrupted. These codes are related to private information retrieval protocols, initiated by [16]. The best known constructions of LDCs are due to Yekhanin [52] and Efremenko [21]. On the other hand, locally correctable codes (LCCs) are error-correcting codes that allow to retrieve each codeword symbol using a small number of queries even after a constant fraction of it is adversely corrupted. So, the difference between LDCs and LCCs is local decoding of message entries vs. codeword entries. It is also worth pointing out that all linear LCCs are LDCs, however, the opposite does not hold [30].

In the full version we prove that the tensor product of codes preserves the local decoding (correction) properties as well as local testability.

Acknowledgements. The author thanks Eli Ben-Sasson for many invaluable discussions about the "robustness" concept and the possible connections to the work [40]. We would like to thank Or Meir for helpful discussions. The author thanks Ronny Roth for pointers to the literature. We thank the anonymous referees for valuable comments on an earlier version of this article.

References

1. Alon, N., Bruck, J., Naor, J., Naor, M., Roth, R.M.: Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. IEEE Transactions on Information Theory 38(2), 509 (1992)
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. Arora, S., Sudan, M.: Improved low-degree testing and its applications. Combinatorica 23(3), 365–426 (2003)
6. Babai, L., Fortnow, L., Levin, L.A., Szegedy, M.: Checking computations in polylogarithmic time. In: Proc. 23rd STOC, pp. 21–31. ACM (1991)
7. Babai, L., Fortnow, L., Lund, C.: Non-deterministic exponential time has two-prover interactive protocols. Computational Complexity 1, 3–40 (1991)
8. Babai, L., Fortnow, L., Nisan, N., Wigderson, A.: BPP has subexponential time simulations unless EXPTIME has publishable proofs. Computational Complexity 3, 307–318 (1993)
9. Ben-Sasson, E., Goldreich, O., Harsha, P., Sudan, M., Vadhan, S.P.: Robust PCPs of proximity, shorter PCPs, and applications to coding. SIAM Journal on Computing 36(4), 889–974 (2006)
10. Ben-Sasson, E., Harsha, P., Raskhodnikova, S.: Some 3CNF Properties Are Hard to Test. SIAM Journal on Computing 35(1), 1–21 (2005)
11. Ben-Sasson, E., Sudan, M.: Simple PCPs with poly-log rate and query complexity. In: STOC, pp. 266–275. ACM (2005)
12. Ben-Sasson, E., Sudan, M.: Robust locally testable codes and products of codes. Random Struct. Algorithms 28(4), 387–402 (2006)
13. Ben-Sasson, E., Viderman, M.: Composition of Semi-LTCs by Two-Wise Tensor Products. In: Dinur, I., Jansen, K., Naor, J., Rolim, J.D.P. (eds.) APPROX and RANDOM 2009. LNCS, vol. 5687, pp. 378–391. Springer, Heidelberg (2009)
14. Ben-Sasson, E., Viderman, M.: Tensor Products of Weakly Smooth Codes are Robust. Theory of Computing 5(1), 239–255 (2009)
15. Ben-Sasson, E., Viderman, M.: Low Rate Is Insufficient for Local Testability. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J.D.P. (eds.) APPROX and RANDOM 2010, LNCS, vol. 6302, pp. 420–433. Springer, Heidelberg (2010)
16. Chor, B., Goldreich, O., Kushilevitz, E., Sudan, M.: Private information retrieval. JACM: Journal of the ACM 45 (1998)
17. Coppersmith, D., Rudra, A.: On the Robust Testability of Product of Codes. Electronic Colloquium on Computational Complexity (ECCC) (104) (2005)
18. Dinur, I.: The PCP theorem by gap amplification. Journal of the ACM 54(3), 12:1–12:44 (2007)

19. Dinur, I., Reingold, O.: Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem. SIAM Journal on Computing 36(4), 975–1024 (2006)
20. Dinur, I., Sudan, M., Wigderson, A.: Robust Local Testability of Tensor Products of LDPC Codes. In: Díaz, J., Jansen, K., Rolim, J.D.P., Zwick, U. (eds.) APPROX and RANDOM 2006. LNCS, vol. 4110, pp. 304–315. Springer, Heidelberg (2006)
21. Efremenko, K.: 3-query locally decodable codes of subexponential length. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31-June 2, pp. 39–44. ACM (2009)
22. Goldreich, O.: Short locally testable codes and proofs (survey). Electronic Colloquium on Computational Complexity (ECCC) (014) (2005)
23. Goldreich, O., Meir, O.: The Tensor Product of Two Good Codes Is Not Necessarily Robustly Testable. Electronic Colloquium on Computational Complexity (ECCC) 14(062) (2007)
24. Goldreich, O., Sudan, M.: Locally testable codes and PCPs of almost-linear length. Journal of the ACM 53(4), 558–655 (2006)
25. Gopalan, P., Guruswami, V., Raghavendra, P.: List decoding tensor products and interleaved codes. In: Mitzenmacher, M. (ed.) Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31-June 2, pp. 13–22. ACM (2009)
26. Guruswami, V., Indyk, P.: Linear time encodable and list decodable codes. In: STOC, pp. 126–135. ACM (2003)
27. Guruswami, V., Indyk, P.: Linear-time encodable/decodable codes with near-optimal rate. IEEE Transactions on Information Theory 51(10), 3393–3400 (2005)
28. Impagliazzo, R., Wigderson, A.: P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In: STOC, pp. 220–229 (1997)
29. Kaufman, T., Sudan, M.: Sparse random linear codes are locally decodable and testable. In: FOCS, pp. 590–600. IEEE Computer Society (2007)
30. Kaufman, T., Viderman, M.: Locally Testable vs. Locally Decodable Codes. In: Serna, M., Shaltiel, R., Jansen, K., Rolim, J.D.P. (eds.) APPROX And RANDOM 2010, LNCS, vol. 6302, pp. 670–682. Springer, Heidelberg (2010)
31. Kopparty, S., Saraf, S.: Local list-decoding and testing of random linear codes from high error. In: Schulman, L.J. (ed.) Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, June 5-8, pp. 417–426. ACM (2010)
32. Kopparty, S., Saraf, S., Yekhanin, S.: High-rate codes with sublinear-time decoding. ECCC - TR10-148 (2010)
33. Lipton, R.J.: Efficient Checking of Computations. In: Choffrut, C., Lengauer, T. (eds.) STACS 1990. LNCS, vol. 415, pp. 207–215. Springer, Heidelberg (1990)
34. Lund, C., Fortnow, L., Karloff, H.J., Nisan, N.: Algebraic methods for interactive proof systems. Journal of the ACM 39(4), 859–868 (1992)
35. Blum, M., Luby, M., Rubinfeld, R.: Self-Testing/Correcting with Applications to Numerical Problems. JCSS: Journal of Computer and System Sciences 47 (1993)
36. Meir, O.: On the rectangle method in proofs of robustness of tensor products. Electronic Colloquium on Computational Complexity (ECCC) 14(061) (2007)
37. Meir, O.: Combinatorial Construction of Locally Testable Codes. SIAM J. Comput. 39(2), 491–544 (2009)
38. Meir, O.: IP = PSPACE using Error Correcting Codes. Electronic Colloquium on Computational Complexity (ECCC) 17, 137 (2010)
39. Moshkovitz, D., Raz, R.: Two-query PCP with subconstant error. J. ACM 57(5) (2010)

40. Raz, R., Safra, S.: A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In: STOC, pp. 475–484 (1997)
41. Reed, I.S.: A class of multiple-error-correcting codes and the decoding scheme. IEEE Transactions on Information Theory 4(4), 38–49 (1954)
42. Roth, R.M.: Introduction to coding theory. Cambridge University Press (2006)
43. Roth, R.M., Skachek, V.: Improved Nearly-MDS Expander Codes. IEEE Transactions on Information Theory 52(8), 3650–3661 (2006)
44. Shaltiel, R., Umans, C.: Simple extractors for all min-entropies and a new pseudo-random generator. J. ACM 52(2), 172–216 (2005)
45. Shamir, A.: IP = PSPACE. J. ACM 39(4), 869–877 (1992)
46. Shen, A.: IP = PSPACE: Simplified proof. J. ACM 39(4), 878–880 (1992)
47. Sipser, M., Spielman, D.A.: Expander Codes. IEEE Transactions on Information Theory 42(6), 1710–1722 (1996); Preliminary version appeared in FOCS 1994
48. Spielman, D.A.: Linear-time Encodable and Decodable Error-Correcting Codes. IEEE Transactions on Information Theory 42(6), 1723–1731 (1996); Preliminary version appeared in STOC 1995
49. Sudan, M., Trevisan, L., Vadhan, S.P.: Pseudorandom generators without the XOR lemma. J. Comput. Syst. Sci. 62(2), 236–266 (2001)
50. Trevisan, L.: Some applications of coding theory in computational complexity. Electronic Colloquium on Computational Complexity (ECCC) (043) (2004)
51. Valiant, P.: The Tensor Product of Two Codes Is Not Necessarily Robustly Testable. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX 2005 and APPROX and RANDOM 2005. LNCS, vol. 3624, pp. 472–481. Springer, Heidelberg (2005)
52. Yekhanin, S.: Towards 3-query locally decodable codes of subexponential length. J. ACM 55(1) (2008)

Extractors for Turing-Machine Sources

Emanuele Viola*

Northeastern University, Boston MA 02115, USA
viola@ccs.neu.edu
http://www.ccs.neu.edu/home/viola/

Abstract. We obtain the first deterministic randomness extractors for n-bit sources with min-entropy $\geq n^{1-\alpha}$ generated (or sampled) by single-tape Turing machines running in time $n^{2-16\alpha}$, for all sufficiently small $\alpha > 0$. We also show that such machines cannot sample a uniform n-bit input to the Inner Product function together with the output.

The proofs combine a variant of the crossing-sequence technique by Hennie [SWCT 1965] with extractors for block sources, especially those by Chor and Goldreich [SICOMP 1988] and by Kamp, Rao, Vadhan, and Zuckerman [JCSS 2011].

Keywords: turing machine, independent source, deterministic randomness extractor, sampling lower bound, complexity of distributions.

1 Introduction

Turing machines may be the most studied model of computation even after decades of work on circuits. Following a first wave of worst-case lower bounds starting in the 60's (cf. [13]) and continuing to this date, researchers in the 90's have produced a second type of results. Specifically, Impagliazzo, Nisan, and Wigderson obtain in [14] average-case lower bounds and pseudorandom generators.

In this work we are interested in what we see as a third type of lower bounds: *sampling* lower bounds. We seek to understand what distributions can be sampled by randomized Turing machines (which take no input).

The first work on sampling complexity may be the one by Jerrum, Valiant, and Vazirani [15] who define sampling complexity classes and prove reductions among various problems. An unconditional communication complexity lower bound for sampling disjointness appears in the work [2] by Ambainis, Schulman, Ta-Shma, Vazirani, and Wigderson. Goldreich, Goldwasser, and Nussboim study the complexity of sampling in [11] as part of a general study of the implementation of huge random objects. Aaronson proves in [1] a connection between sampling and searching problems.

The complexity of sampling is being revisited in a series of recent works [26,19,9,25,6]. These works establish the first unconditional lower bounds for several computational models, such as bounded-depth circuits, and draw several new

* Supported by NSF grant CCF-0845003.

A. Gupta et al. (Eds.): APPROX/RANDOM 2012, LNCS 7408, pp. 663–671, 2012.
© Springer-Verlag Berlin Heidelberg 2012

connections to problems in data structures, combinatorics, and randomness extractors. The connection to randomness extractors in particular makes progress along the research direction initiated by Trevisan and Vadhan in [24], and continued by Kamp, Rao, Vadhan, and Zuckerman in [16], which aims to construct deterministic randomness extractors for efficiently-samplable distributions.

1.1 Our Results

Our main result is an extractor for sources samplable by Turing machines running in subquadratic time. For clarity we first review randomized Turing machines.

In this work, Turing machines have exactly one read-write tape, infinite to the right only, with exactly one head on it. One may choose $\{0,1\}$ as tape alphabet. The tape is initially blank, that is, all zeros. In one time step, the machine reads the content of the cell, tosses a coin, and then writes the cell, updates the state, and moves the head to an adjacent location. Machines never halt, and we are only interested in a portion of their computation table. A $t \times t$ computation table is a $t \times t$ matrix corresponding to a valid computation according to such rules, with rows being configurations. Each entry specifies the content of the corresponding tape cell, whether the head is on that cell, and if so what is the current state and the current coin toss. Since we store the coin tosses in the entries, all $t \times t$ computation tables have equal probability 2^{-t}.

A Turing machine source on n bits running in time t is sampled as follows. First sample uniformly the $t \times t$ computation table. Then output the bottom left n tape bits.

Theorem 1 (Extractors for Turing-machine sources). *For all sufficiently small $\alpha > 0$,*
there is an explicit extractor $E : \{0,1\}^n \to \{0,1\}^m$ with output length $m = n^{\Omega(1)}$ and error $2^{-n^{\Omega(1)}}$ for n-bit sources with min-entropy $\geq k := n^{1-\alpha/16}$ that are sampled by Turing machines with $\leq 2^q := 2^{n^{\alpha/16}}$ states and running in time $\leq t := n^{2-\alpha}$.

The above theorem implies sampling lower bounds for somewhat complicated functions. The next one obtains one for the inner-product function IP.

Theorem 2 (Sampling lower bound for Turing machines). *For every $\alpha \in (0,1]$ and all sufficiently large even n*

no Turing machine with $\leq 2^q := 2^{n^{\alpha/2}}$ states and running in time $\leq t := n^{2-\alpha}$ can sample the distribution

$$(X_1, X_2, IP(X_1, X_2))$$

where X_1 and X_2 are uniform and independent over $\{0,1\}^{n/2}$.

Note that this result depends on the ordering of the input bits – if the bits of X_1 and X_2 are interleaved then a Turing machine can sample the distribution in linear time.

1.2 Overview of the Proofs

To prove our results we show that any Turing-machine source contains an independent source. More specifically, divide the n bits of the source into r blocks (or runs) of length ℓ separated by blocks of length b, as in Figure 1. We show that any Turing-machine source running in subquadratic time is a convex combination of sources $Y_1 Y_2 \cdots Y_r$ where the Y_i are independent, and each Y_i covers exactly one of the ℓ-bit blocks:

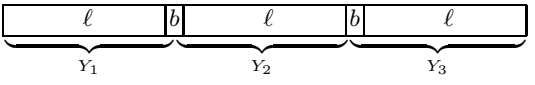

Fig. 1. Decomposition of Turing-machine source in $r = 3$ blocks (or runs) or ℓ bits separated by blocks of b bits

Lemma 1 (Turing-machine sources contain independent sources). *Let X be a Turing machine source on n bits running in time $t \geq n$ with 2^q states and min-entropy k.*

For any ℓ, b such that $(r-1)(\ell+b)+\ell = n$, X is a convex combination of $J \leq 2^{r \cdot O(q(\lg t)t/b)}$ n-bit sources S_j where each S_j is

$$S_j = Y_1 Y_2 \ldots Y_r,$$

where the Y_i are independent, and for every $i < r$ we have $\ell i + b(i-1) \leq |Y_1 Y_2 \ldots Y_i| \leq \ell i + bi$.

One can then extract using extractors for independent sources, developed in an exciting, ongoing line of research; see e.g. [22,8,3,4,21,16,20,7,5,18]. One gets different results depending on which extractors one uses. However, many of the available extractors for independent sources require a guarantee on the min-entropy of each source. By contrast, our given guarantee on the min-entropy of the Turing-machine source only translates into a guarantee on the *total* min-entropy of the independent sources. Thus for our extractor in Theorem 1 we use the extractors by Kamp, Rao, Vadhan, and Zuckerman [16] which only require that.

The sampling lower bound for IP in Theorem 2 is obtained by using instead the result by Chor and Goldreich that the inner product function $IP : \{0,1\}^\ell \times \{0,1\}^\ell \to \{0,1\}$ is a two-source extractor with error ϵ if the sum of the entropies of the two sources is $> \ell + 2\lg(1/\epsilon)$.[8]

We now elaborate on how we prove that any Turing-machine source contains an independent source. First, we introduce a variant of the classical crossing-sequence technique due to Hennie [12] that is suitable for sampling tasks. This allows us to sample the Turing-machine source by a one-way low-communication protocol among r players. This is explained in more detail below. Compared to previous simulations [17, §12] ours has the advantage of incurring no error. Another difference is that in our setting it is advantageous to have a large number

of players. (This is because the number of players corresponds to the number of independent blocks, and in general the more the independent blocks the easier the extraction.)

We then use the fact that a source sampled by a one-way low-communication protocol is a convex combination of independent sources. For 2 players, this fact originates from the work [2, §7] of Ambainis, Schulman, Ta-Shma, Vazirani, and Wigderson. Alternatively, one may view the sources sampled by such protocols as the extension of the source model in [16] where we output blocks instead of bits.

This concludes the high-level view of the proof. In the next paragraph we elaborate on how to sample a Turing-machine source by a low-communication protocol.

From Turing's Machines to Yao's Protocols. Let $T := (C_1, C_2, \ldots, C_t)$ be a distribution on $t \times t$ computation tables, where C_i represents the ith column of the table. We first describe an alternative way to sample T; then we explain how this alternative way can be implemented as a low-communication protocol.

The alternative way to sample T comes from the observation that the random variables $C_1, C_2, \ldots$ are a markov process (or chain). That is, conditioned on C_i, the random variable $C_{<i}$ of the columns before the ith is independent from the random variable $C_{>i}$ of the columns after the ith. The alternative way proceeds by sampling T from left to right one column at the time, each time conditioning only on the previous column (as opposed to the entire prefix). For example, one first samples $C_1 = c_1$, then samples $C_2 = c_2 | C_1 = c_1$, then samples $C_3 = c_3 | C_2 = c_2$, and so on. Let us call the resulting distribution $T^?$. To see that T and $T^?$ are the same distribution, note that after conditioning on a column $C_i = c_i$, T becomes a product distribution: the columns before i are independent from those after i. This holds because $T | C_i = c_i$ is uniform on its support (since each computation table has probability 2^{-t}), and by locality of computation: if $c_{<i}\ c_i\ c_{>i}$ and $c'_{<i}\ c_i\ c'_{>i}$ are in the support of $T | C_i = c_i$, then so is $c_{<i}\ c_i\ c'_{>i}$. It is now an exercise to show that for any transcript $t = (c_1, c_2, \ldots, c_t)$ we have $\Pr[T = t] = \Pr[T^? = t]$. The solution to the exercise follows.

$$\begin{aligned}
\Pr[T = t] &= \prod_i \Pr[C_i = c_i | C_{<i} = c_{<i}]; \\
\Pr[T^? = t] &= \prod_i \Pr[C_i = c_i | C_{i-1} = c_{i-1}] \\
&= \prod_i \frac{\Pr[C_i = c_i \wedge C_{<i-1} = c_{<i-1} | C_{i-1} = c_{i-1}]}{\Pr[C_{<i-1} = c_{<i-1} | C_{i-1} = c_{i-1}]} \\
&\qquad \text{(Since } T | C_{i-1} = c_{i-1} \text{ is product)} \\
&= \Pr[T = t].
\end{aligned}$$

We then exploit the above alternative way to sample T efficiently by a low-communication protocol among r players. Refer to Figure 1 for the parameters. The first player samples one column at a time. After an appropriate number ℓ

of columns, it looks for the first column that has a short description. By locality of computation, among b columns there must be one that corresponds to a tape cell that the Turing-machine head scans $\leq t/b$ times. Since modifications of a column only occur when the head scans it, this column can be described with about t/b bits, which is $< n$ for $t = n^{2-\alpha}$ and $b = n^{1-\alpha/2}$. The player can send this description to the next player, who can then continue the process.

2 Proofs

Proof (of Lemma 1). We prove this in two stages. In the first, more substantial stage we show how to sample the entire source X using a one-way low-communication protocol in which Player i outputs a sample covering Y_i but touching no Y_j for $j \neq i$. In the second stage we condition on the protocol's transcript.

We now proceed to the first stage. Let $T = (C_1, C_2, \ldots, C_t)$ be the uniform distribution over $t \times t$ computation tables.

P_1 starts sampling T from left to right, one column at the time. It stops at the first tape-cell index s_1 such that $\ell < s_1 \leq \ell + b$ and such that the sample c_{s_1} of C_{s_1} contains $\leq t/b$ states. Since each row only has the state in one cell, such an s_1 is guaranteed to exist. Because changes to tape contents only happen when the head is on that cell, this column can be described with

$$O(q(\lg t)t/b)$$

bits. The $\lg t$ term arises from specifying the times where the head is on that cell.

P_1 outputs the first s_1 output bits of the computation table. It then sends both the description of c_{s_1} and s_1 to P_2. This takes $O(q(\lg t)t/b) + O(\lg t) = O(q(\lg t)t/b)$ bits.

P_2 will then continue sampling the computation table from left to right one column at the time. It stops at the smallest tape-cell index s_2 such that $(\ell + b) + \ell < s_2 \leq 2(\ell + b)$ and such that the sample c_{s_2} of C_{s_2} contains $\leq t/b$ states. And so on.

This is the end of stage 1.

By conditioning on the communication, we can write the output distribution as a convex combination of $J \leq 2^{r \cdot O(q(\lg t)t/b)}$ distributions S_j. After conditioning on the communication, the players' output are independent and have a fixed length. Hence each S_j is a product distribution $S_j = Y_1 Y_2 \ldots Y_r$ where Y_i is the output of P_i. The bounds on the lengths of $Y_1 Y_2 \ldots Y_i$ follow by inspection.

The following standard claim bounds the entropy loss when selecting a distribution from a convex combination.

Claim (Entropy Loss in Convex Combo). Let D be a distribution with min-entropy k that is a convex combination of $J = 2^j$ distributions $D_1, D_2, \ldots, D_J$. Consider sampling D by first appropriately selecting an index $h \leq J$, and then sampling D_h. For every ϵ, the probability over the choice of h that D_h has min-entropy $\leq k - j - \lg(1/\epsilon)$ is $\leq \epsilon$.

Proof. Suppose the probability is $> \epsilon$. There is a $h \leq J$ that is picked with probability $> \epsilon/J$ such that D_h has min-entropy $\leq k - j - \lg(1/\epsilon)$. This means that there is some a such that $\Pr[D_h = a] \geq 1/2^{k-j-\lg(1/\epsilon)}$. But then $\Pr[D = a] > \epsilon/J \cdot 1/2^{k-j-\lg(1/\epsilon)} > 1/2^k$.

We use the following extractor.

Theorem 3 (Theorem 5.1 in [16]). *There is a constant $\beta > 0$ such that for every ℓ and $\delta \geq 1/\ell^\beta$ there is an explicit extractor for min-entropy $\geq \delta r \ell$ sources over $\left(\{0,1\}^\ell\right)^r$ such that the r blocks of ℓ bits are independent and with $r \geq 1/(\beta\delta^2)$, with output length $m = \ell^{\Omega(1)}$, and error $\epsilon = 2^{-\ell^{\Omega(1)}}$.*

Using the techniques in [10,23] one can derive a similar extractor where almost all the entropy is output, cf. [16, §7]. However we do not pursue this here.

We now prove our main extractor result.

Proof (of Theorem 1). For an α to be determined later, set $b := n^{1-\alpha/2}$ and $\ell := n^{1-\alpha/4}$. We assume w.l.o.g. that $\ell+b$ divides $n+b$. Note $r := (n+b)/(\ell+b) = \Theta(n^{\alpha/4})$.

Divide the n bits of the source into r runs of ℓ bits separated by $r-1$ runs of b bits. We apply the extractor from Theorem 3 to the r runs of ℓ bits.

By Lemma 1 we view the source as a convex combination of $J \leq 2^{O(rq(\lg t)t/b)}$ product sources S_j. By Claim 2 with $\epsilon := 2^{-k/2}$, if we choose a distribution in the combination, except with probability ϵ we obtain a distribution with min-entropy at least

$$\begin{aligned} &k - O(rq(\lg t)t/b) - \lg(1/\epsilon) \geq k/2 - O(rq(\lg t)t/b) \\ =&n^{1-\alpha/16}/2 - O(n^{\alpha/4+\alpha/16+1-\alpha/2} \lg n) = n^{1-\alpha/16}/2 - O(n^{1-3\alpha/16} \lg n) \\ \geq&\Omega(k). \end{aligned}$$

We assume this is the case and proceed.

By ignoring the $r-1$ runs of b bits, we drop $(r-1)b \leq O(n^{\alpha/4}n^{1-\alpha/2}) = O(n^{1-\alpha/4})$ bits. Since $k \geq n^{1-\alpha/16}$, the extractor is applied to a distribution of entropy that is still $\Omega(k)$.

Also, since we ignore the $r-1$ runs of b bits, the r runs of ℓ bits to which the extractor is applied are independent.

The parameter δ in theorem 3 is

$$\delta = \Theta(k/r\ell) = \Theta(k/n) = \Theta(1/n^{\alpha/16}).$$

We must have

$$\delta \geq 1/\ell^\beta = 1/n^{(1-\alpha/4)\beta}$$

for the constant β in the statement of Theorem 3. This is the case for α sufficiently small.

We also must have

$$r \geq 1/(\beta\delta^2) = \Theta(n^{\alpha/8}/\beta)$$

which is true because $r = \Theta(n^{\alpha/4})$ as observed above.

The output length is $m = \ell^{\Omega(1)} = n^{\Omega(1)}$. The error of the extractor is $2^{-\ell^{\Omega(1)}} = 2^{-n^{\Omega(1)}}$.

Combined with the above error of $2^{-k/2}$ arising from the convex combination, we obtain a total error of again $2^{-n^{\Omega(1)}}$.

For the lower bound for sampling inner product we make use of the following theorem.

Theorem 4 ([8]). *Let X_1 and X_2 be two independent sources on ℓ bits. Suppose the sum of the min-entropies is $\geq \ell + 2\lg(1/\epsilon)$. Then $|\Pr[IP(X_1, X_2) = 1] - 1/2| \leq \epsilon$.*

We now prove our sampling lower bound for inner product.

Proof (of Theorem 2). Suppose there was such a Turing machine. Consider the Turing machine M' that first samples $(X_1, X_2, IP(X_1, X_2))$ then if $IP(X_1, X_2) = 1$ it outputs (X_1, X_2), otherwise it outputs a uniform n-bit string. M' can be implemented, say, in time $O(t)$ with $O(2^q)$ states.

The machine M' samples a distribution (X_1', X_2') with min-entropy $k \geq n-1$. Moreover, because $\Pr[IP(X_1, X_2) = 1]$ approaches $1/2$ for large n, we see that $\Pr[IP(X_1', X_2') = 1]$ approaches $3/4$ for large n.

Set $b := 0.01n$. By Lemma 1, (X_1', X_2') is a convex combination of sources S_j such that except with probability 0.01 over the choice of an independent source from this combination, S_j has min-entropy

$$\begin{aligned}
&\geq n - O(1) - O(q(\lg t)t/b) - \lg(1/0.01) \\
&\geq n - O(n^{\alpha/2}(\lg n)n^{1-\alpha} - O(1) \\
&\geq 0.99n.
\end{aligned}$$

Moreover, each S_j is $S_j = Y_1Y_2$ for independent Y_1, Y_2 and $\ell \leq |Y_1| \leq \ell + b$, where $n = 2\ell + b$. Assume without loss of generality that $|Y_1| \geq |Y_2|$. By conditioning on the $b = 0.01n$ middle bits (each of which depends on exclusively Y_1 or Y_2), we can further write (Y_1, Y_2) as a convex combination of $\leq 2^b$ sources S_j' where each S_j' is $S_j' = Y_1'Y_2'$ where $|Y_1'| = |Y_2'| = n/2$ and Y_1', Y_2' are independent. $Y_1'Y_2'$ has min-entropy $\geq 0.99n - 0.01n = 0.98n$.

This min-entropy is larger than $n/2 + 2\lg(100)$. Hence by Theorem 4 IP will successfully extract one bit with error 0.01.

Overall, the error of the extracted bit is $\leq 0.01 + 0.01 = 0.02$. This contradicts the above remark that $\Pr[IP(X_1', X_2') = 1]$ approaches $3/4$ for large n.

In this proof the extractor is applied to the whole sample, whereas in the proof of Theorem 1 it is applied to a projection of it. That was only for convenience. One could have applied the extractor to the whole sample and then condition on the values of the runs of b bits.

References

1. Aaronson, S.: The equivalence of sampling and searching. In: Computer Science Symp. in Russia (CSR), pp. 1–14 (2011)
2. Ambainis, A., Schulman, L.J., Ta-Shma, A., Vazirani, U.V., Wigderson, A.: The quantum communication complexity of sampling. SIAM J. Comput. 32(6), 1570–1585 (2003)
3. Barak, B., Impagliazzo, R., Wigderson, A.: Extracting randomness using few independent sources. SIAM J. Comput. 36(4), 1095–1118 (2006)
4. Barak, B., Kindler, G., Shaltiel, R., Sudakov, B., Wigderson, A.: Simulating independence: New constructions of condensers, ramsey graphs, dispersers, and extractors. J. of the ACM 57(4) (2010)
5. Barak, B., Rao, A., Shaltiel, R., Wigderson, A.: 2-source dispersers for sub-polynomial entropy and Ramsey graphs beating the Frankl-Wilson construction. In: ACM Symp. on the Theory of Computing (STOC), pp. 671–680 (2006)
6. Beck, C., Impagliazzo, R., Lovett, S.: Large deviation bounds for decision trees and sampling lower bounds for AC0-circuits. Electronic Colloquium on Computational Complexity (ECCC) 19, 42 (2012)
7. Bourgain, J.: More on the sum-product phenomenon in prime fields and its applications. Int. J. of Number Theory (IJNT) 1, 1–32 (2005)
8. Chor, B., Goldreich, O.: Unbiased bits from sources of weak randomness and probabilistic communication complexity. SIAM J. on Computing 17(2), 230–261 (1988)
9. De, A., Watson, T.: Extractors and Lower Bounds for Locally Samplable Sources. In: Goldberg, L.A., Jansen, K., Ravi, R., Rolim, J.D.P. (eds.) APPROX/RANDOM 2011. LNCS, vol. 6845, pp. 483–494. Springer, Heidelberg (2011)
10. Gabizon, A., Raz, R., Shaltiel, R.: Deterministic extractors for bit-fixing sources by obtaining an independent seed. SIAM J. on Computing 36(4), 1072–1094 (2006)
11. Goldreich, O., Goldwasser, S., Nussboim, A.: On the implementation of huge random objects. SIAM J. Comput. 39(7), 2761–2822 (2010)
12. Hennie, F.C.: Crossing sequences and off-line turing machine computations. In: Symposium on Switching Circuit Theory and Logical Design (SWCT) (FOCS), pp. 168–172 (1965)
13. Hopcroft, J.E., Ullman, J.D.: Formal languages and their relation to automata. Addison-Wesley Longman Publishing Co., Inc. (1969)
14. Impagliazzo, R., Nisan, N., Wigderson, A.: Pseudorandomness for network algorithms. In: 26th ACM Symp. on the Theory of Computing (STOC), pp. 356–364 (1994)
15. Jerrum, M.R., Valiant, L.G., Vazirani, V.V.: Random generation of combinatorial structures from a uniform distribution. Theoretical Computer Science 43(2-3), 169–188 (1986)
16. Kamp, J., Rao, A., Vadhan, S.P., Zuckerman, D.: Deterministic extractors for small-space sources. J. Comput. Syst. Sci. 77(1), 191–220 (2011)
17. Kushilevitz, E., Nisan, N.: Communication complexity. Cambridge University Press (1997)
18. Li, X.: Improved constructions of three source extractors. In: IEEE Conf. on Computational Complexity, CCC (2011)
19. Lovett, S., Viola, E.: Bounded-depth circuits cannot sample good codes. Computational Complexity 21(2), 245–266 (2012)
20. Rao, A.: Extractors for low-weight affine sources. In: IEEE Conf. on Computational Complexity (CCC), pp. 95–101 (2009)

21. Raz, R.: Extractors with weak random seeds. In: ACM Symp. on the Theory of Computing (STOC), pp. 11–20 (2005)
22. Santha, M., Vazirani, U.V.: Generating quasi-random sequences from semi-random sources. J. of Computer and System Sciences 33(1), 75–87 (1986)
23. Shaltiel, R.: How to get more mileage from randomness extractors. Random Struct. Algorithms 33(2), 157–186 (2008)
24. Trevisan, L., Vadhan, S.: Extracting randomness from samplable distributions. In: IEEE Symp. on Foundations of Computer Science (FOCS), pp. 32–42 (2000)
25. Viola, E.: Extractors for circuit sources. In: IEEE Symp. on Foundations of Computer Science, FOCS (2011)
26. Viola, E.: The complexity of distributions. SIAM J. on Computing 41(1), 191–218 (2012)

Author Index