38th Computational Complexity Conference

CCC 2023, July 17–20, 2023, Warwick, UK

Edited by

Amnon Ta-Shma

Editors

Amnon Ta-Shma
Tel Aviv University, Israel
amnon@tauex.tau.ac.il

ACM Classification 2012
Theory of computation

ISBN 978-3-95977-282-2

Published online and open access by
Schloss Dagstuhl – Leibniz-Zentrum für Informatik GmbH, Dagstuhl Publishing, Saarbrücken/Wadern, Germany. Online available at https://www.dagstuhl.de/dagpub/978-3-95977-282-2.

Publication date
July, 2023

Bibliographic information published by the Deutsche Nationalbibliothek
The Deutsche Nationalbibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data are available in the Internet at https://portal.dnb.de.

Digital Object Identifier: 10.4230/LIPIcs.CCC.2023.0

LIPIcs – Leibniz International Proceedings in Informatics

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

ISSN 1868-8969

https://www.dagstuhl.de/lipics

Contents

Papers

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Preface

The papers in this volume were accepted for presentation at the 38th Computational Complexity Conference (CCC 2023), held between July 17–20, 2023, in Warwick, UK. The conference is organized by the Computational Complexity Foundation (CCF) in cooperation with the ACM Special Interest Group on Algorithms and Computation Theory (SIGACT) and the European Association for Theoretical Computer Science (EATCS).

The call for papers sought original research papers in all areas of computational complexity theory. Of the 74 submissions, the program committee selected 36 for presentation at the conference.

The program committee would like to thank everyone involved in the conference, including all those who submitted papers for consideration as well as the reviewers (listed separately) for their scientific contributions; the board of trustees of the Computational Complexity Foundation for their advice and assistance; the Local Arrangements Committee chair Tom Gur; Nikolas Breuckmann, Shuichi Hirahara, and Salil Vadhan for their invited talks; and Michael Wagner for coordinating the production of these proceedings.

Amnon Ta-Shma
Program Committee Chair, on behalf of the Program Committee

Conference Organization

Program Committee

Eli Ben-Sasson, StarkWare
Andrej Bogdanov, Chinese University of Hong Kong (until Jan 2023) and University of Ottawa (after Jan 2023)
Lijie Chen, UC Berkeley
Alex Bredariol Grilo, LIP6 (CNRS/Sorbonne Université)
Mika Göös, EPFL
William Hoza, Simons Institute for the Theory of Computing
Susanna F. de Rezende, Lund University
Neeraj Kayal, Microsoft Research India
Tali Kaufmann, Bar-Ilan University
François Le Gall, Nagoya University
Raghu Meka, UCLA
Amnon Ta-Shma (Chair), Tel-Aviv University

Local Arrangements Committee

Tom Gur (Chair), University of Warwick

Board of Trustees

Amit Chakrabarti, Dartmouth College
Valentine Kabanets (President), Simon Fraser University
Michal Koucký, Charles University
Nutan Limaye IIT Bombay
Meena Mahajan, The Institute of Mathematical Sciences
Pierre McKenzie, Université de Montréal
Benjamin Rossman, Duke University
Shubhangi Saraf, University of Toronto
Ryan Williams, Massachusetts Institute of Technology

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

External Reviewers

Gorjan Alagic
Yaroslav Alekseev
Eric Allender
Omar Alrabiah
Andris Ambainis
Robert Andrews
Srinivasan Arunachalam
Aleksandrs Belovs
Shalev Ben-David
Guy Blanc
Ilario Bonacina
Samuel Bouaziz
Mark Bun
Libor Caha
Arkadev Chattopadhyay
Sitan Chen
Kuan Cheng
Tsun Ming Cheung
Shawn Cui
Radu Curticapean
Yotam Dikstein
Michael Dinitz
Irit Dinur
Pavel Dvořák
Prateek Dwivedi
Bill Fefferman
Dmytro Gavinsky
Alexandru Gheorghiu
Halley Goldberg
Oded Goldreich
Alexander Golovnev
Louis Golowich
Jesse Goodman
Spencer Gordon
Roy Gotlib
Fernando Granha Jeronimo
Kasper Green Larsen
Joshua Grochow
Ziyi Guan
Zeyu Guo
Tom Gur
Tuomas Hakoniemi
Nathaniel Harms
Pooya Hatami
Tal Herman
Shuichi Hirahara
Alexandros Hollender
Christian Ikenmeyer
Rahul Ilango
Yuval Ishai
Siddhartha Jain
Ce Jin
Chris Jones
Amitay Kamber
Robbie King
Leszek Kolodziejczyk
Alexis Korb
Sajin Koroth
Oliver Korten
Robin Kothari
William Kretschmer
Mrinal Kumar
Srijita Kundu
Victor Lecomte
Chin Ho Lee
Nutan Limaye
Jiahui Liu
Yunchao Liu
Shachar Lovett
Xin Lyu
Nathan Manohar
Peter Manohar
Or Meir
Dimitrios Myrisiotis
Mikito Nanashima
Oded Nir
Igor Carboni Oliveira
Michael Oliveira
Izhar Oppenheim
Benedikt Pago
Tomáš Peitl
Naty Peter
Supartha Podder
Aaron Potechin
Kevin Pratt
Dömötör Pálvölgyi
Luowen Qian
Youming Qiao
Mikhail Raskin
Alexander Razborov

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany
COMPUTATIONAL COMPLEXITY CONFERENCE

Daniel Reichman
Hanlin Ren
Artur Riazanov
Kilian Risse
Robert Robere
Ansis Rosmanis
Benjamin Rossman
Ron Rothblum
Dorian Rudolph
Rahul Santhanam
Ramprasad Saptharishi
Pascal Schweitzer
Ronen Shaltiel
Makrand Sinha
Dmitry Sokolov
Harsha Srimath Tirumala
Srikanth Srinivasan
Sathyawageeswar Subramanian
Avishay Tal
Roei Tell
Neil Thapen
Iddo Tzameret
Neekon Vafa
Prashant Vasudevan
Emanuele Viola
Ben Lee Volk
Daochen Wang
James Watson
Ryan Williams
Freek Witteveen
Hongxun Wu
Tal Yankovitz
Amir Yehudayoff
Huacheng Yu
Weiqiang Yuan
Mark Zhandry
Jeroen Zuiddam

Separation of the Factorization Norm and Randomized Communication Complexity

Tsun-Ming Cheung
School of Computer Science, McGill University, Montreal, Canada

Hamed Hatami
School of Computer Science, McGill University, Montreal, Canada

Kaave Hosseini
Department of Computer Science, University of Rochester, NY, USA

Morgan Shirley
Department of Computer Science, University of Toronto, Canada

Abstract

In an influential paper, Linial and Shraibman (STOC '07) introduced the factorization norm as a powerful tool for proving lower bounds against randomized and quantum communication complexities. They showed that the logarithm of the *approximate* γ_2-factorization norm is a lower bound for these parameters and asked whether a stronger lower bound that replaces approximate γ_2 norm with the γ_2 norm holds.

We answer the question of Linial and Shraibman in the negative by exhibiting a $2^n \times 2^n$ Boolean matrix with γ_2 norm $2^{\Omega(n)}$ and randomized communication complexity $O(\log n)$.

As a corollary, we recover the recent result of Chattopadhyay, Lovett, and Vinyals (CCC '19) that deterministic protocols with access to an Equality oracle are exponentially weaker than (one-sided error) randomized protocols. In fact, as a stronger consequence, our result implies an exponential separation between the power of unambiguous nondeterministic protocols with access to Equality oracle and (one-sided error) randomized protocols, which answers a question of Pitassi, Shirley, and Shraibman (ITSC '23).

Our result also implies a conjecture of Sherif (Ph.D. thesis) that the γ_2 norm of the Integer Inner Product function (`IIP`) in dimension 3 or higher is exponential in its input size.

2012 ACM Subject Classification Theory of computation → Communication complexity

Keywords and phrases Factorization norms, randomized communication complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.1

Funding *Hamed Hatami*: Supported by an NSERC grant.
Morgan Shirley: Supported by an NSERC grant.

1 Introduction

The γ_2-factorization norm is an important notion of matrix complexity that was initially developed in Banach Space theory. In an influential paper, Linial and Shraibman [12] introduced this norm to communication complexity. Subsequently, the factorization norm and its approximate version found numerous applications in communication complexity and other adjacent areas such as discrepancy theory [13] and differential privacy [14, 3, 7].

▶ **Definition 1** (γ_2-factorization norm). *The* γ_2 norm *of a real matrix* A *is*

$$\|A\|_{\gamma_2} := \min_{X,Y:A=XY} \|X\|_{\text{row}} \|Y\|_{\text{col}},$$

where $\|X\|_{\text{row}}$ *and* $\|Y\|_{\text{col}}$ *denote the largest* ℓ_2*-norm of a row in* X *and the largest* ℓ_2 *norm of a column in* Y*, respectively.*

▶ **Definition 2** (Approximate γ_2 norm). *The* approximate γ_2 norm *of $A \in \mathbb{R}^{k\times\ell}$ with error ϵ, denoted by $\widetilde{\gamma}_2^{\epsilon}(A)$, is the minimum $\|B\|_{\gamma_2}$ over all matrices $B \in \mathbb{R}^{k\times\ell}$ with $\|A-B\|_\infty \leq \epsilon$.*

We use the notation $\widetilde{\gamma}_2^{\epsilon}(\cdot)$ to emphasize that unlike the γ_2 norm $\|\cdot\|_{\gamma_2}$, the approximate γ_2 norm is not a norm. The choice of the error parameter ϵ is mostly unimportant in the context of communication complexity. Indeed, a constant-factor reduction in the error parameter increases $\log \widetilde{\gamma}_2^{\epsilon}(A)$ by a constant factor [1, Lemma 21]. Therefore, we use the standard choice of $\epsilon = 1/3$ and write $\widetilde{\gamma}_2$ for $\widetilde{\gamma}_2^{1/3}$. Both of the quantities $\widetilde{\gamma}_2$ and γ_2 are polynomial-time computable using semi-definite programming [12].

Linial and Shraibman [12] showed that $\log \widetilde{\gamma}_2(A)$ provides a lower bound on the public-coin randomized communication complexity $\mathrm{R}(A)$ and the quantum communication complexity with shared entanglement $\mathrm{Q}^*(A)$:

$$\log \widetilde{\gamma}_2(A) \lesssim \mathrm{Q}^*(A) \leq \mathrm{R}(A). \tag{1}$$

These lower bounds subsume the most well-known lower bounds on randomized and quantum communication complexity, such as discrepancy, approximate trace norm [17], and entropy of singular values [9].

Linial and Shraibman [12] state that "they cannot rule out the intriguing possibility that the first inequality in Equation (1) is a tip of something bigger and randomized communication complexity and the quantum communication complexity with shared entanglement are in fact polynomially equivalent to $\log \|A\|_{\gamma_2}$."

▶ **Question 1** ([12]). *Is $\log \|A\|_{\gamma_2} \leq \widetilde{O}(\mathrm{R}(A))$ for every a Boolean matrix $A : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$?*

Here, the notation $\widetilde{O}(\cdot)$ hides a factor of $\mathrm{polylog}(n)$, which is common in communication complexity since the communication cost of $\mathrm{polylog}(n)$ is considered efficient.

Another motivation for Question 1 comes from the following observation. It is well-known that the Equality function $\textsc{eq} : \{0,1\}^n \times \{0,1\}^n$ has $\mathrm{R}(\textsc{eq}) = O(1)$ (see e.g. [10]) but its rank over the reals is 2^n, and therefore EQ witnesses the strongest possible separation ($O(1)$ versus 2^n) between R and rank. On the other hand, as mentioned before, the γ_2 norm can be viewed as a smooth analogue of rank. However, the γ_2 norm of the Equality function is 1, and therefore, one naturally wonders whether there is a strong separation between $\mathrm{R}(\cdot)$ and the γ_2 norm.

The purpose of the present paper is to give a strong negative answer to Question 1. In fact, we work with a stronger parameter of $\mathrm{R}_0^1(A)$ instead of $\mathrm{R}(A)$. This parameter is the minimum cost of a *one-sided* public-coin randomized protocol. The protocol is not allowed to have any error on 1 entries of A, but on the 0 entries, it can have a probability of error as big as 1/3.

1.1 Main Result

Our main result establishes a strong separation between the γ_2 norm and R_0^1.

▶ **Theorem 3** (Main Theorem). *There is a Boolean matrix $M : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with $\|M\|_{\gamma_2} \geq 2^{n/32}$ and $\mathrm{R}_0^1(M) \leq O(\log n)$.*

The construction in Theorem 3 is based on the *point-line incidence matrix* over the integers. For integers $1 \leq q \leq p$, let $\mathtt{PL}$ be the $qp \times qp$ Boolean matrix whose rows and columns are indexed by the elements of $[q] \times \{0,\ldots,p-1\}$ and its entries are given as $\mathtt{PL}[(x,x'),(y,y')] = 1$ iff $xy + x' = y'$. We also define a variant of $\mathtt{PL}$ over $\mathbb{Z}_p$ to simplify the analysis. The matrix $\mathtt{PL}_{\mathbb{Z}_p}$ is the $qp \times qp$ Boolean matrix whose rows and columns are indexed by $[q] \times \mathbb{Z}_p$ and its entries are given as $\mathtt{PL}_{\mathbb{Z}_p}[(x,x'),(y,y')] = 1$ iff $xy + x' \equiv y' \mod p$.

Recall that the trace norm of a matrix is the sum of its singular values (see Section 2.1).Theorem 3 is immediate from the following theorem, which is our main technical contribution.

▶ **Theorem 4** (Technical Statement of the Main Theorem). *Let p be a prime.*

(i) *For $1 \le q \le \sqrt{p}$, we have*

$$\|\mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}} = \Omega(pq^{9/8}) \quad \textit{and} \quad \|\mathtt{PL}_{\mathbb{Z}_p}\|_{\gamma_2} = \Omega(q^{1/8}) \quad \textit{and} \quad \mathrm{R}_0^1(\mathtt{PL}_{\mathbb{Z}_p}) = O(\log\log p).$$

(ii) *For $1 \le q \le p^{1/3}$, we have*

$$\|\mathtt{PL}\|_{\mathrm{Tr}} = \Omega(pq^{9/8}) \quad \textit{and} \quad \|\mathtt{PL}\|_{\gamma_2} = \Omega(q^{1/8}) \quad \textit{and} \quad \mathrm{R}_0^1(\mathtt{PL}) = O(\log\log p).$$

▶ Remark 5. The condition $1 \le q \le p^{1/3}$ in (ii) allows us to deduce (ii) from (i) since $\|\mathtt{PL}_{\mathbb{Z}_p} - \mathtt{PL}\|_{\mathrm{Tr}} = o(pq^{9/8})$ in this range (see Lemma 15). On the other hand, the condition $1 \le q \le \sqrt{p}$ in (i) is to guarantee $\mathrm{R}_0^1(\mathtt{PL}_{\mathbb{Z}_p}) = O(\log\log p)$. Indeed, unlike $\mathtt{PL}$, whose randomized communication complexity is always small, the randomized communication complexity of $\mathtt{PL}_{\mathbb{Z}_p}$ is large when q is close to p. For example, for $q = p$, this follows from the fact that all nontrivial eigenvalues of $\mathtt{PL}_{\mathbb{Z}_p}$ are at most $\sqrt{3p}$ [19].

1.2 Consequences of the Main Theorem

As an immediate consequence, combining Theorem 3 with Equation (1) implies an exponential separation between $\widetilde{\gamma}_2(\cdot)$ and $\|\cdot\|_{\gamma_2}$. This corollary answers a question of Pitassi, Shirley, and Shraibman [16, Open Question 3].

▶ **Corollary 6.** *There is a Boolean matrix $M : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with $\|M\|_{\gamma_2} \ge 2^{n/32}$ and $\widetilde{\gamma}_2(M) \le O(\mathrm{poly}(n))$.*

Another corollary of Theorem 3 concerns the deterministic communication complexity with oracle access to the Equality function. We formally define this model in Section 2.2 and denote the corresponding complexity measure by $\mathrm{D}^{\mathrm{EQ}}(\cdot)$. The equality function, which corresponds to the identity matrix, is the standard example of a problem with $O(1)$ randomized communication complexity but large deterministic communication complexity. This fact makes $\mathrm{D}^{\mathrm{EQ}}(\cdot)$ an interesting complexity measure between randomized and deterministic communication complexities.

$$\log \widetilde{\gamma}_2(A) \lesssim \mathrm{Q}^*(A) \le \mathrm{R}(A) \lesssim \mathrm{D}^{\mathrm{EQ}}(A) \le \mathrm{D}(A). \tag{2}$$

Since the γ_2 norm of the identity matrix is 1, it is not hard to see that [5, Proposition 3.1]

$$\frac{1}{2}\log \|A\|_{\gamma_2} \le \mathrm{D}^{\mathrm{EQ}}(A). \tag{3}$$

In light of Equation (3), Theorem 3 implies the following.

▶ **Corollary 7.** *There is a Boolean matrix $M : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with $\mathrm{R}_0^1(M) \le O(\log n)$ and $\mathrm{D}^{\mathrm{EQ}}(M) = \Omega(n)$.*

The above corollary recovers the result of Chattopadhyay, Lovett, and Vinyals [2] separating R and D^{EQ}. In fact, we obtain an exponential lower bound on a model stronger than D^{EQ}. In complexity theory, *unambiguous nondeterminism* is similar to nondeterminism but with the extra requirement that for every input, there is at most one accepting computational path. Therefore, the power of unambiguous nondeterminism lies between determinism and nondeterminism. For a Boolean matrix M, the *unambiguous nondeterministic communication complexity of M with access to an equality oracle* is denoted by $\mathrm{UP}^{\mathrm{EQ}}$ (see Section 2.2). It is immediate that $\mathrm{UP}^{\mathrm{EQ}}(\cdot) \le \mathrm{D}^{\mathrm{EQ}}(\cdot)$. Theorem 3 implies the following corollary, answering a question of Pitassi, Shirley, and Shraibman [16, Open Question 2].

▶ **Corollary 8.** *There is a Boolean matrix* $M : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ *with* $\mathrm{R}_0^1(M) \leq O(\log n)$ *and* $\mathrm{UP}^{EQ}(M) = \Omega(n)$.

The matrix PL that we consider in Theorem 4 is essentially a submatrix of the Integer Inner Product matrix (IIP) used in the work of Chattopadhyay et al. [2]; however, the proof technique here is entirely different.

▶ **Definition 9.** *Let* $t \in \mathbb{N}$ *be a fixed constant. For a positive integer* $m = 2^n$*, the Integer Inner Product function* $\mathtt{IIP}_t : \{-m, \ldots, m\}^t \times \{-m, \ldots, m\}^t \to \{0,1\}$ *is defined as*

$$\mathtt{IIP}_t[(x_1, \ldots, x_t), (y_1, \ldots, y_t)] = 1 \text{ iff } x_1y_1 + \ldots + x_ty_t = 0.$$

Since t is a fixed constant, the input size of $\mathtt{IIP}_t$ is $\Theta(n)$-bits as a communication problem. Chattopadhyay, Lovett, and Vinyals proved that $\mathrm{R}_0^1(\mathtt{IIP}_t) = O(\log n)$, and $\mathrm{D}^{\mathrm{EQ}}(\mathtt{IIP}_t) = \Omega(n)$ for $t \geq 6$.

Later, Sherif [18] conjectured $\|\mathtt{IIP}_t\|_{\gamma_2} = 2^{\Omega(n)}$ for $t \geq 6$. Since the matrix PL is a submatrix of $\mathtt{IIP}_3$, as a corollary of Theorem 4, we answer Sherif's question in the affirmative.

▶ **Corollary 10.** *For* $t \geq 3$*,*

$$\|\mathtt{IIP}_t\|_{\gamma_2} = 2^{\Omega(n)}.$$

Proof. Choose n such that $2^{n-1} \leq p \leq 2^n$ and $q = \lceil p^{1/3} \rceil$. From Theorem 4, we obtain PL as a submatrix of $\mathtt{IIP}_3$ with $m = 2^n$ such that $\|\mathtt{PL}\|_{\gamma_2} = \Omega(2^{n/32})$. Since the γ_2 norm cannot increase when restricting to a submatrix, we conclude that

$$\|\mathtt{IIP}_t\|_{\gamma_2} \geq \|\mathtt{IIP}_3\|_{\gamma_2} \geq \|\mathtt{PL}\|_{\gamma_2} = 2^{\Omega(n)}. \qquad \blacktriangleleft$$

▶ Remark 11. The condition $t \geq 3$ is necessary as $\|\mathtt{IIP}_2\|_{\gamma_2} = O(1)$. To prove the latter, we use Equation (3) and show $\mathrm{D}^{\mathrm{EQ}}(\mathtt{IIP}_2) = O(1)$. Note that if $x_1y_1 + x_2y_2 = 0$ and $y_1, x_2 \neq 0$, then $\frac{x_1}{x_2} = -\frac{y_2}{y_1}$. To check this equation, Alice and Bob can call the Equality oracle on rational inputs $\frac{x_1}{x_2}$ and $-\frac{y_2}{y_1}$.

1.3 Connections to Fourier Algebra Norm

The sum of the absolute values of the Fourier coefficients of a function $f : \mathbb{Z}_2^n \to \mathbb{R}$ is called the *algebra norm* of f:

$$\|f\|_A \coloneqq \|\widehat{f}\|_1 = \sum_{a \in \mathbb{Z}_2^n} |\widehat{f}(a)|.$$

For any error parameter $\epsilon \in (0, 1/2)$, the ϵ-*approximate algebra norm* of $f : \mathbb{Z}_2^n \to \{0,1\}$ is

$$\widetilde{A}_\epsilon(f) \coloneqq \inf\{\|g\|_A \ : \ \|f - g\|_\infty \leq \epsilon\}.$$

It is possible to use the XOR operation to lift these norms to the γ_2 norm and the approximate γ_2 norm [12]: for the matrix $F : \mathbb{Z}_2^n \times \mathbb{Z}_2^n \to \{0,1\}$ defined by $F(x,y) = f(x \oplus y)$, we have

$$\|f\|_A = \|F\|_{\gamma_2} \qquad \text{and} \qquad \widetilde{A}_\epsilon(f) = \widetilde{\gamma}_2^\epsilon(F).$$

The communication complexity measures of F are related to the parity query complexity measures of f. For example, we have

$$\mathrm{R}(F) \leq 2\,\mathrm{rdt}^{\oplus}(f),$$

where $\mathrm{rdt}^{\oplus}(f)$ denotes the randomized parity decision tree complexity of f (see [5]).

Therefore, the class of XOR-lifted Boolean functions provide a rich collection of matrices for which the questions about the factorization norm reduce to simpler questions about the Fourier algebra norm. In this setting, one can ask the analog of Question 1.

▶ **Question 2** (Open Question). *Is* $\log \|f\|_A = \widetilde{O}(\mathrm{rdt}^{\oplus}(f))$ *for every Boolean function* $f : \mathbb{Z}_2^n \to \{0,1\}$*?*

By the above discussion, if we find a counter-example f to Question 2, then $F(x,y) \coloneqq f(x \oplus y)$ would be a counter-example to Question 1. However, Question 2 remains open. Indeed, our counter-example to Question 1 is not an XOR-lift.

Finally, let us comment on the stronger versions of Question 1 and Question 2, where we do not tolerate a polylog(n) factor, i.e., replace $\widetilde{O}(\cdot)$ with $O(\cdot)$. Let $B(n,r) \subseteq \{0,1\}^n$ denote the Hamming ball of radius r around the origin, i.e.,

$$B(n,r) \coloneqq \left\{ x \in \{0,1\}^n \ : \ \sum_{i=1}^{n} x_i \le r \right\}.$$

Note that the lifted function $F_{n,r}(x,y) = \mathbf{1}_{B(n,r)}(x \oplus y)$ corresponds to the hamming distance problem, whose communication complexity is well-understood. We have [8]

$$\mathrm{rdt}^{\oplus}(\mathbf{1}_{B(n,r)}) \le O(r \log r) \qquad \text{and} \qquad \mathrm{R}(F_{n,r}) \le O(r \log r).$$

On the other hand, for $r \le n/2$, the following bounds are known [5, Lemma 2.15] about the Fourier algebra norm of $\mathbf{1}_{B(n,r)}$:

$$e^{-r} \sqrt{\sum_{i=0}^{r} \binom{n}{i}} \le \left\|\mathbf{1}_{B(n,r)}\right\|_A = \|F_{n,r}\|_{\gamma_2} \le \sqrt{\sum_{i=0}^{r} \binom{n}{i}}.$$

Therefore, in the context of Question 2 and Question 1, taking $r = O(1)$ provides examples of $f : \{0,1\}^n \to \{0,1\}$ and $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ with

$$\mathrm{rdt}^{\oplus}(f) = O(1) \qquad \text{and} \qquad \log \|f\|_A = \Theta(\log n),$$

and

$$\mathrm{R}(F) = O(1) \qquad \text{and} \qquad \log \|F\|_{\gamma_2} = \Theta(\log n).$$

Paper Organization

In Section 2, we discuss the preliminaries of matrix norms, communication complexity, and Fourier analysis. We give a brief overview of the proof strategy in Section 3. We present the proof of Theorem 4 in Sections 4 and 5. Finally, we discuss several open problems in Section 6.

2 Notations and Preliminaries

For a positive integer k, we denote $[k] \coloneqq \{1, \dots, k\}$. We use the shorthand notations $a \equiv_p b$ to denote $a \equiv b \bmod p$. For a set S, we use the indicator function notation $\mathbf{1}_S$, which is evaluated to 1 on x if $x \in S$ and 0 otherwise. All the logarithms in this paper are in base 2.

We adopt the standard computer science asymptotic notations and use the tilde asymptotic notations to hide poly-logarithmic factors. We write $f \lesssim g$ to denote $f(n) = O(g(n))$.

For a vector $v \in \mathbb{C}^k$, we denote the ℓ_2-norm of v by $\|v\|_2 = \sqrt{\sum_i |v_i|^2}$. We denote the all-1 matrix by $\mathbf{J}$.

2.1 Matrix Norms

For a complex-valued matrix $A \in \mathbb{C}^{k\times\ell}$, we denote the singular values of A by

$$\sigma_1(A) \geq \sigma_2(A) \geq \ldots \geq \sigma_{\min(k,\ell)}(A) \geq 0.$$

We primarily work with the matrix norm family of *Schatten norms*. For $p \in [1,\infty]$, the Schatten-p norm of a matrix is the ℓ_p norm of the vector of its singular values. The particular cases of $p = 1, 2, \infty$ are frequently used, and these norms are commonly known as *trace norm*, *Frobenius norm*, and *spectral norm* respectively:

$$\|A\|_{\mathrm{Tr}} = \|A\|_{S_1} = \sum_i \sigma_i$$
$$\|A\|_F = \|A\|_{S_2} = \sqrt{\sum_i \sigma_i^2} = \sqrt{\sum_{i,j} |A_{ij}|^2}$$
$$\|A\| = \|A\|_{S_\infty} = \sigma_1 = \max_{x\in\mathbb{C}^\ell : \|x\|_2=1} \|Ax\|_2 = \max_{\substack{u\in\mathbb{C}^k, v\in\mathbb{C}^\ell \\ \|u\|_2=\|v\|_2=1}} u^*Av$$

Viewing Schatten p-norm as the ℓ_p norm of the singular value vector, one can obtain several useful properties inherited from ℓ_p norms. One such property is the monotonicity of Schatten p-norm in p: $\|A\|_{S_p} \geq \|A\|_{S_q}$ for $1 \leq p < q \leq \infty$.

Similar to the case of ℓ_p norm, for $p, q \in [1,\infty]$ with $\frac{1}{p} + \frac{1}{q} = 1$, the dual norm of $\|\cdot\|_{S_p}$ is $\|\cdot\|_{S_q}$. With the inner product on the matrix space $\mathbb{C}^{k\times\ell}$ defined by $\langle A, B\rangle = \mathrm{Tr}(A^*B) = \sum_{ij} \overline{A_{ij}} B_{ij}$, the Schatten p-norm admits the following dual norm characterization:

$$\|A\|_{S_p} = \max_{\|B\|_{S_q}=1} |\langle A, B\rangle|.$$

For the particular case of $p = 1$, this yields

$$|\langle A, B\rangle| \leq \|A\|_{\mathrm{Tr}} \|B\|.$$

In particular, by setting $B = A$, we have

$$\|A\|_F^2 \leq \|A\|_{\mathrm{Tr}} \|A\|. \tag{4}$$

Next, we discuss a reformulation of the γ_2 norm in terms of the trace norm. As shown in [11], for $A \in \mathbb{R}^{k\times\ell}$, we have

$$\|A\|_{\gamma_2} = \max_{\substack{u\in\mathbb{R}^k, v\in\mathbb{R}^\ell \\ \|u\|_2=\|v\|_2=1}} \|A \circ uv^T\|_{\mathrm{Tr}}.$$

Here $\circ$ denotes the Hadamard (or entrywise) product of two matrices: for $B, C \in \mathbb{R}^{k\times\ell}$, their product $B \circ C$ is the $m \times n$ matrix defined by $[B \circ C]_{ij} = B_{ij}C_{ij}$ for all i, j. It follows from the trace norm formulation of the γ_2 norm that

$$\|A\|_{\gamma_2} \geq \frac{1}{\sqrt{k\ell}} \|A\|_{\mathrm{Tr}}. \tag{5}$$

2.2 Communication Complexity

In the standard communication model, there are two parties and problems are modelled by functions $f : \mathcal{X} \times \mathcal{Y} \to \{0, 1\}$ on finite domains $\mathcal{X}, \mathcal{Y}$. The two parties receive $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, respectively, and they exchange messages to compute $f(x, y)$. We often interpret f as a Boolean matrix indexed by $(x, y) \in \mathcal{X} \times \mathcal{Y}$.

For a given $\epsilon \in (0, 1/2)$, we denote by $\mathrm{R}_\epsilon(f)$, the randomized communication complexity of f in the public-coin model with two-sided error $\epsilon > 0$. The one-sided versions, $\mathrm{R}^1_\epsilon(f)$ and $\mathrm{R}^1_{0,\epsilon}(f)$, restrict the error to be one-sided: $\mathrm{R}^1_\epsilon(f)$ does not allow any error on the inputs in $f^{-1}(0)$. Similarly, $\mathrm{R}^1_{0,\epsilon}(f)$ does not allow any error on the inputs in $f^{-1}(1)$. We refer the reader to [10] for the formal definitions. We use the canonical choice of $\epsilon = 1/3$ and drop ϵ in the notations in such cases. This choice is without loss of generality since the probability of error can be reduced to any constant $\epsilon' > 0$ by repeating the protocol a constant number of times and outputting the majority.

As mentioned, approximate norms are useful tools for studying communication complexity. The following well-known inequalities [6, Proposition A.2] connect approximate γ_2 norm with randomized communication complexity.

$$\log \widetilde{\gamma}_2(A) \leq \mathrm{R}(A) \leq O(\widetilde{\gamma}_2(A)^2). \tag{6}$$

Next, we define the *deterministic communication complexity with access to an equality oracle*. In this model, a protocol computing a Boolean matrix $A_{\mathcal{X}\times\mathcal{Y}}$ corresponds to a binary tree. Each non-leaf node v in the tree is labelled with two functions $a_v : \mathcal{X} \to \mathcal{Z}$ and $b_v : \mathcal{Y} \to \mathcal{Z}$ for a finite set $\mathcal{Z}$. Such a node v corresponds to the query $\textsc{eq}(a_v(x), b_v(y))$, which returns 1 if $a_v(x) = b_v(y)$ and 0 otherwise. Every input (x, y) naturally corresponds to a path from the tree's root to a leaf, and the leaf must be labelled with the correct value $A(x, y)$. The cost of the protocol is the depth of the tree. The deterministic communication complexity of the matrix A with access to an equality oracle, denoted by $\mathrm{D}^{\textsc{eq}}(A)$, is the smallest depth of such a protocol for A.

Consider a node v in an equality-oracle deterministic communication protocol as described above. Note that the matrix $B_v(x, y) \coloneqq \textsc{eq}(a_v(x), b_v(y))$ consists of a collection of all-1 submatrices with rows and columns disjoint. Such matrices are dubbed *blocky matrices* by [5]. The answer to the query at the node v will inform the parties whether the input (x, y) belongs to the support of B_v or the support of $\mathbf{J} - B_v$.

Consider a leaf ℓ of the protocol tree where the protocol outputs 1, and let $v_1, \ldots, v_d = \ell$ be the set of the nodes on the corresponding path from the root. The inputs that lead the protocol to reach ℓ are the 1 entries of the matrix $M_\ell \coloneqq C_{v_1} \circ \ldots \circ C_{v_{d-1}}$ with $C_{v_i} = B_{v_i}$ or $C_{v_i} = \mathbf{J} - B_{v_i}$ according to the outcome of the query at v_i. Each matrix C_{v_i} is either a blocky matrix or the difference of two blocky matrices. Since the γ_2 norm of a Blocky matrix is at most 1, it follows that $\|C_{v_i}\|_{\gamma_2} \leq 2$. Since γ_2 is an algebra norm (i.e., $\|X \circ Y\|_{\gamma_2} \leq \|X\|_{\gamma_2}\|Y\|_{\gamma_2}$), we have $\|M_\ell\|_{\gamma_2} \leq 2^d$. Note that $A = \sum M_\ell$ where the sum is over all the leaves where the protocol outputs 1. Hence,

$$\|A\|_{\gamma_2} \leq 4^d. \tag{7}$$

An *unambiguous nondeterministic protocol with access to equality oracle* is a collection of 2^m deterministic equality-oracle protocols, each with depth at most d, such that on every input, at most one of them returns 1. The cost of such a protocol is $m + d$. Consider such a protocol for a Boolean matrix A, and let $A_1, \ldots, A_{2^m}$ be the Boolean matrices computed by the 2^m deterministic equality-oracle protocols. We must have $A = \sum_{i=1}^{2^m} A_i$, and in particular, by Equation (7), we have

$$\|A\|_{\gamma_2} \leq \sum_{i=1}^{2^m} \|A_i\|_{\gamma_2} \leq 2^m \times 4^d = 2^{m+2d}.$$

We denote by $\mathrm{UP}^{\textsc{eq}}(A)$, the smallest cost of an unambiguous nondeterministic equality-oracle protocol for A. We conclude

$$\frac{1}{2} \log \|A\|_{\gamma_2} \leq \mathrm{UP}^{\textsc{eq}}(A) \leq \mathrm{D}^{\textsc{eq}}(A). \tag{8}$$

2.3 Fourier Analysis of $\mathbb{Z}_p^k$

This section gives a basic overview of Fourier analysis on the finite Abelian group $G \coloneqq \mathbb{Z}_p^k$ for $p, k \in \mathbb{N}$. Consider the Hilbert space $L^2(G)$ with the inner product of two functions $f, g : G \to \mathbb{C}$ defined by

$$\langle f, g \rangle = \sum_{x \in G} f(x)\overline{g(x)}.$$

The inner product defines the norm $\|f\|_2 = \sqrt{\langle f, f \rangle}$.

Consider the principal p-th root of unity $\omega \coloneqq e^{2\pi i/p}$. For every element $a = (a_1, \ldots, a_k) \in \mathbb{Z}_p^k$, define the corresponding Fourier character $\chi_a : G \to \mathbb{C}$ as

$$\chi_a(x) = \omega^{\sum_{j=1}^k a_j x_j}.$$

The Fourier characters form an orthogonal basis for $L^2(G)$:

$$\langle \chi_a, \chi_b \rangle = \sum_{x \in G} \chi_{a-b}(x) = \begin{cases} |G| & \text{if } a = b \\ 0 & \text{otherwise} \end{cases}.$$

Therefore, every function $f : G \to \mathbb{C}$ has a unique expansion

$$f = \sum_{a \in G} \widehat{f}(a)\chi_a,$$

where

$$\widehat{f}(a) = \frac{1}{|G|}\langle f, \chi_a \rangle.$$

It follows from the orthogonality of the Fourier characters that for every $f : G \to \mathbb{C}$,

$$\sum_{x \in G} |f(x)|^2 = |G| \sum_{a \in G} |\hat{f}(a)|^2. \tag{9}$$

This identity is called *Parseval's identity*.

3 Overview of the Proof of the Main Theorem

Let $1 \le q \le p$, and let M be the $([q] \times \mathbb{Z}_p) \times ([q] \times \mathbb{Z}_p)$ Boolean matrix defined as $M[(x, x'), (y, y')] = 1$ iff $xy = x' + y'$. Note that $M[(x, x'), (y, y')] = \mathtt{PL}_{\mathbb{Z}_p}[(x, -x'), (y, y')]$, and thus M is just a row permutation of $\mathtt{PL}_{\mathbb{Z}_p}$. Therefore, $\|M\|_{\mathrm{Tr}} = \|\mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}}$.

Let $\sigma_1 \ge \ldots \ge \sigma_N$ be the singular values of M. Since M is a real symmetric matrix and every row of M contains exactly q ones, the largest eigenvalue of M is $\sigma_1 = q$, which corresponds to the all-1 eigenvector. If M were a "pseudo-random" matrix in the sense that all of its non-principal eigenvalues were small (i.e., $\sigma_2 < q^{1-\epsilon}$), then one could easily show that the trace norm of M is large. Indeed, the Frobenius norm of M is equal to

$$\sqrt{\sum_{(x,x'),(y,y')} M[(x, x'), (y, y')]^2} = \sqrt{qN},$$

therefore

$$\|M\|_{\mathrm{Tr}} \ge \sum_{i=2}^{N} \sigma_i \ge \frac{\sum_{i=2}^N \sigma_i^2}{\sigma_2} = \frac{qN - q^2}{\sigma_2} = \Omega\left(\frac{qN}{\sigma_2}\right). \tag{10}$$

However, we cannot expect M to be pseudo-random since pseudo-random matrices have large randomized communication complexity and this is not the case for M.

To prove a lower bound for $\|M\|_{\mathrm{Tr}}$, there is nothing special about removing only the largest singular value in Equation (10). One can take any subspace $W \subseteq \mathbb{R}^N$ and apply Equation (4) to the orthogonal projection of M to W. More precisely, let $P_W : \mathbb{R}^N \to \mathbb{R}^N$ be the orthogonal projection from $\mathbb{R}^N$ to W. By Equation (4), we have

$$\|M\|_{\mathrm{Tr}} \geq \|P_W^*\|\|M\|_{\mathrm{Tr}}\|P_W\| \geq \|P_W^* M P_W\|_{\mathrm{Tr}} \geq \frac{\|P_W^* M P_W\|_F^2}{\|P_W^* M P_W\|}.$$

Taking W as the orthogonal complement of the principal eigenvector of M yields Equation (10). The natural choice to strengthen this lower bound is to take W as the span of the eigenvectors of M that correspond to small eigenvalues. Dropping the first $k-1$ largest eigenvalues will result in the lower bound $\|M\|_{\mathrm{Tr}} \geq \frac{\sum_{i=k}^N \sigma_k^2}{\sigma_k}$. If a non-negligible mass of $\|M\|_F^2$ is on the tail $\sum_{i=k}^N \sigma_k^2$ for some $\sigma_k < q^{1-\epsilon}$, then this approach provides a strong lower bound for $\|M\|_{\mathrm{Tr}}$.

Unfortunately, the direct application of this method requires determining the eigenvectors and eigenvalues of M, which seems difficult. To circumvent this difficult task, we employ tools from Fourier analysis and show that there is a linear span of some Fourier characters $W \subseteq \mathbb{R}^N$ such that $\|P_W^* M P_W\|_F = \Omega(\|M\|_F)$ and $\|P_W^* M P_W\|$ is small.

4 Randomized Communication Complexities of PL and $\mathtt{PL}_{\mathbb{Z}_p}$

We divide the proof of Theorem 4 into two sections. In this section, we prove the upper bounds of Theorem 4 on $\mathrm{R}_0^1(\mathtt{PL}_{\mathbb{Z}_p})$ and $\mathrm{R}_0^1(\mathtt{PL})$.

▶ **Proposition 12.** *For $q \leq \sqrt{p}$, we have $\mathrm{R}_0^1(\mathtt{PL}_{\mathbb{Z}_p}) = O(\log\log p)$. For every $1 \leq q \leq p$, we have $\mathrm{R}_0^1(\mathtt{PL}) = O(\log\log p)$.*

Proof. We describe a randomized protocol that solves $\mathtt{PL}_{\mathbb{Z}_p}$ with cost $O(\log\log p)$ that never makes mistakes on inputs where $\mathtt{PL}_{\mathbb{Z}_p}$ takes value 1. The same protocol also solves PL.

Suppose Alice and Bob have inputs $(x, x'), (y, y') \in [q] \times \mathbb{Z}_p$ respectively. Since $q \leq \sqrt{p}$, we have

$$[xy + x' \equiv_p y'] \iff [xy + x' = y'] \vee [xy + x' = y' + p].$$

In the rest of the proof, we show that each of the two equations on the right-hand side can be verified with a protocol of cost at most $O(\log\log p)$ and error at most $1/6$, which then implies a protocol of cost $O(\log\log p)$ and error at most $1/3$ for the matrix $\mathtt{PL}_{\mathbb{Z}_p}$. Suppose Alice and Bob want to verify whether $xy + x' = y'$; the case for $xy + x' = y' + p$ is similar. Alice picks a uniformly random prime $\mathbf{r}$ from the set of the first $\lceil 6\log(2p) \rceil$ primes $\mathcal{P}$ and sends it to Bob. Alice and Bob exchange the values $(x \mod \mathbf{r}), (x' \mod \mathbf{r}), (y \mod \mathbf{r}), (y' \mod \mathbf{r})$ and check whether

$$(x \mod \mathbf{r})(y \mod \mathbf{r}) + (x' \mod \mathbf{r}) \equiv_{\mathbf{r}} (y' \mod \mathbf{r}),$$

or equivalently

$$xy + x' \equiv_{\mathbf{r}} y'.$$

The cost of this communication is at most $O(\log \mathbf{r}) = O(\log\log p)$. Next, we show that the probability of error (over the choice of $\mathbf{r}$) is at most $1/6$. Observe that an error can only happen when $xy + x' \neq y'$ but $xy + x' \equiv_{\mathbf{r}} y'$. We want to show that

$$\Pr_{\mathbf{r} \in \mathcal{P}}[[xy + x' \neq y'] \wedge [xy + x' \equiv_{\mathbf{r}} y']] \leq \frac{1}{6}.$$

Let $B \subseteq \mathcal{P}$ be the set of bad choices for $\mathbf{r}$, namely

$$B = \{r \in \mathcal{P} : [xy + x' \neq y'] \wedge [xy + x' \equiv_{\mathbf{r}} y']\}.$$

Suppose towards a contradiction that $|B| > \frac{|\mathcal{P}|}{6}$. Define

$$m \coloneqq \prod_{r \in B} r \geq 2^{|B|} > 2p.$$

Note that for all $r \in B$, we have $xy + x' \equiv_r y'$. By the Chinese remainder theorem, we have $xy + x' \equiv_m y'$. This implies the contradiction that $xy + x' = y'$ because $0 \leq xy + x', y' < 2p < m$. ◀

▶ Remark 13. Note that the protocol used in the proof Proposition 12 is in fact a private-coin protocol, so the bounds in Proposition 12 hold in both private-coin and public-coin models.

▶ Remark 14. Combining Proposition 12 with Equation (6), we obtain

$$\widetilde{\gamma}_2(\mathtt{PL}_{\mathbb{Z}_p}) \leq \log^{O(1)}(N). \tag{11}$$

5 Trace Norms of PL and $\mathtt{PL}_{\mathbb{Z}_p}$

This section is dedicated to proving the lower bounds on $\|\mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}}$ and $\|\mathtt{PL}\|_{\mathrm{Tr}}$ of Theorem 4. The lower bounds on $\|\mathtt{PL}_{\mathbb{Z}_p}\|_{\gamma_2}$ and $\|\mathtt{PL}\|_{\gamma_2}$ immediately follow from Equation (5).

▶ **Lemma 15.** *For $1 \leq q \leq p$, we have*

$$\|\mathtt{PL} - \mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}} \leq q^4.$$

In particular, if $q \leq p^{1/3}$, then

$$\|\mathtt{PL} - \mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}} = O(pq).$$

Proof. For $(x, x'), (y, y') \in [q] \times \{0, \ldots, p-1\}$, we have

$$\mathtt{PL}_{\mathbb{Z}_p}[(x, x'), (y, y')] = 1 \text{ iff } [xy + x' = y'] \vee [xy + x' = y' + p].$$

Therefore, we can write $\mathtt{PL}_{\mathbb{Z}_p} = \mathtt{PL} + A$, where A is defined as

$$A[(x, x'), (y, y')] = 1 \text{ iff } xy + x' = y' + p.$$

Because $xy \leq q^2$ and $x' < p$, $xy + x' = y' + p$ implies $y' < q^2$. Therefore, A has at most q^4 non-zero entries. Consequently $\|A\|_{\mathrm{Tr}} \leq q^4$. ◀

By Lemma 15, to complete the proof of Theorem 4, it suffices to prove $\|\mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}} = \Omega(pq^{9/8})$. Since we want to apply Fourier analysis to study the trace norm of $\mathtt{PL}_{\mathbb{Z}_p}$, it is more convenient to extend the rows and columns of $\mathtt{PL}_{\mathbb{Z}_p}$ to $G \coloneqq \mathbb{Z}_p^2$ by adding all-zero rows and columns. That is, we consider $M : G \times G \to \{0, 1\}$, defined as

$$M[(x, x'), (y, y')] = \begin{cases} 1 & \text{if } x, y \in [q] \text{ and } xy \equiv_p x' + y' \\ 0 & \text{otherwise} \end{cases}.$$

This definition of M is slightly different from the one used in the proof overview, but all of the properties we want still hold. For $x, y \in [q]$, we have $M[(x, x'), (y, y')] = \mathtt{PL}_{\mathbb{Z}_p}[(x, -x'), (y, y')]$, and M is zero on the other entries. In other words, M is obtained

from $\mathtt{PL}_{\mathbb{Z}_p}$ by first permuting the rows according to the change of variable $x' \to -x'$, then adding several all-zero rows and columns. These operations do not change the matrix's trace, Frobenius, and spectral norm, and in particular,

$$\|\mathtt{PL}_{\mathbb{Z}_p}\|_{\mathrm{Tr}} = \|M\|_{\mathrm{Tr}}.$$

For $(\alpha, \beta) \in G$, let $\chi_{\alpha,\beta} : G \to \mathbb{C}$ denote the corresponding character in $\widehat{G}$, defined as $\chi_{\alpha,\beta} : (x, x') \mapsto \omega^{\alpha x + \beta x'}$ where $\omega = e^{2\pi i/p}$.

Let $S \subseteq \mathbb{Z}_p$, and π_S be the $G \times G$ matrix corresponding to the orthogonal projection from $L^2(G)$ to the span of $\chi_{\alpha,\beta}$ for $(\alpha, \beta) \in \mathbb{Z}_p \times S$. That is, for $f : G \to \mathbb{C}$,

$$\pi_S f = \sum_{\alpha \in \mathbb{Z}_p} \sum_{\beta \in S} \widehat{f}(\alpha, \beta) \chi_{\alpha,\beta}.$$

Denote $M_S \coloneqq \pi_S^* M \pi_S$. Since π_S is an orthogonal projection, we have $\pi_S = \pi_S^*$ and $\|\pi_S\| = \|\pi_S^*\| \leq 1$, and therefore,

$$\|M_S\|_{\mathrm{Tr}} = \|\pi_S^* M \pi_S\|_{\mathrm{Tr}} \leq \|\pi_S^*\| \|M\|_{\mathrm{Tr}} \|\pi_S\| \leq \|M\|_{\mathrm{Tr}}.$$

Hence, we can use Equation (4) to obtain a lower bound for $\|M\|_{\mathrm{Tr}}$:

$$\|M\|_{\mathrm{Tr}} \geq \|M_S\|_{\mathrm{Tr}} \geq \frac{\|M_S\|_F^2}{\|M_S\|}.$$

First, we determine the value of $\|M_S\|_F$.

▶ **Lemma 16.** *For any $S \subseteq \mathbb{Z}_p$, $\|M_S\|_F = q\sqrt{|S \cap (-S)|}$.*

Proof. Since $\frac{1}{\sqrt{|G|}}\chi_{\alpha,\beta}$'s form an orthonormal basis for $L^2(G)$, for every matrix $B \in \mathbb{C}^{G \times G}$, we have

$$\|B\|_F^2 = \frac{1}{|G|^2} \sum_{(\alpha,\beta),(\alpha',\beta') \in G} |\langle B\chi_{\alpha,\beta}, \chi_{\alpha',\beta'}\rangle|^2. \tag{12}$$

For every $\alpha, \alpha', \beta, \beta' \in \mathbb{Z}_p$, we have

$$\langle M_S \chi_{\alpha,\beta}, \chi_{\alpha',\beta'}\rangle = \langle M\pi_S\chi_{\alpha,\beta}, \pi_S\chi_{\alpha',\beta'}\rangle = \begin{cases} \langle M\chi_{\alpha,\beta}, \chi_{\alpha',\beta'}\rangle & \text{if } \beta, \beta' \in S \\ 0 & \text{otherwise} \end{cases}$$

and therefore, by Equation (12),

$$\|M_S\|_F^2 = \frac{1}{|G|^2} \sum_{(\alpha,\beta),(\alpha',\beta') \in \mathbb{Z}_p \times S} |\langle M\chi_{\alpha,\beta}, \chi_{\alpha',\beta'}\rangle|^2. \tag{13}$$

For $\beta \in S$, define the matrix $F_\beta \in \mathbb{C}^{p \times p}$ as

$$F_\beta(\alpha, \alpha') = \sum_{x,y \in [q]} \omega^{\alpha x + \alpha' y + \beta xy}. \tag{14}$$

Let $\alpha, \alpha' \in \mathbb{Z}_p$ and $\beta, \beta' \in S$. We have

$$\begin{aligned}
\langle M\chi_{\alpha,\beta}, \chi_{\alpha',\beta'}\rangle &= \sum_{x,y\in\mathbb{Z}_p}\sum_{x',y'\in\mathbb{Z}_p} M[(y,y'),(x,x')]\chi_{\alpha,\beta}(x,x')\overline{\chi_{\alpha',\beta'}(y,y')} \\
&= \sum_{x,y\in[q]}\sum_{x',y'\in\mathbb{Z}_p} M[(y,y'),(x,x')]\chi_{\alpha,\beta}(x,x')\overline{\chi_{\alpha',\beta'}(y,y')} \\
&= \sum_{x,y\in[q]}\sum_{y'\in\mathbb{Z}_p} \chi_{\alpha,\beta}(x,xy-y')\overline{\chi_{\alpha',\beta'}(y,y')} \\
&= \sum_{x,y\in[q]}\sum_{y'\in\mathbb{Z}_p} \omega^{\alpha x+\beta(xy-y')-\alpha' y-\beta' y'} \\
&= \sum_{x,y\in[q]} \omega^{\alpha x-\alpha' y+\beta xy}\sum_{y'\in\mathbb{Z}_p}\omega^{-(\beta+\beta')y'} \\
&= \begin{cases} pF_\beta(\alpha,-\alpha') & \text{if } \beta=-\beta' \\ 0 & \text{otherwise} \end{cases}.
\end{aligned}$$

Combining this with Equation (13) gives

$$\|M_S\|_F^2 = \frac{p^2}{|G|^2}\sum_{\alpha,\alpha'\in\mathbb{Z}_p}\sum_{\beta\in S\cap(-S)}|F_\beta(\alpha,-\alpha')|^2 = \frac{1}{|G|}\sum_{\beta\in S\cap(-S)}\|F_\beta\|_F^2. \tag{15}$$

Furthermore,

$$\begin{aligned}
\|F_\beta\|_F^2 &= \sum_{\alpha,\alpha'\in\mathbb{Z}_p}\sum_{x,y\in[q]}\omega^{\alpha x+\alpha' y+\beta xy}\sum_{x',y'\in[q]}\omega^{-(\alpha x'+\alpha' y'+\beta x'y')} \\
&= \sum_{\alpha,\alpha'\in\mathbb{Z}_p}\sum_{x,y,x',y'\in[q]}\omega^{\alpha(x-x')+\alpha'(y-y')+\beta(xy-x'y')} \\
&= \sum_{x,y,x',y'\in[q]}\omega^{\beta(xy-x'y')}\sum_{\alpha,\alpha'\in\mathbb{Z}_p}\omega^{\alpha(x-x')+\alpha'(y-y')}.
\end{aligned}$$

The inner sum is zero unless $x=x'$ and $y=y'$, in which case the inner sum is evaluated to p^2. Thus, for every β, we have $\|F_\beta\|_F^2 = q^2p^2$. We conclude that

$$\|M_S\|_F^2 = \frac{|S\cap(-S)|q^2p^2}{|G|} = q^2|S\cap(-S)|. \qquad \blacktriangleleft$$

Next, we turn to the upper bound of the spectral norm of M_S.

▶ **Lemma 17.** *There is a set $S\subseteq\mathbb{Z}_p$, closed under negation and of size $|S|\geq p/2$, such that $\|M_S\|\leq 2q^{7/8}$.*

Proof. We have

$$\|M_S\| = \max_{\substack{f,g:G\to\mathbb{C}\\ \|f\|_2=\|g\|_2=1}}\langle M_S f,g\rangle = \max_{\substack{f,g:G\to\mathbb{C}\\ \|f\|_2=\|g\|_2=1}}\langle M\pi_S f,\pi_S g\rangle.$$

Define $\widehat{f}_\beta,\widehat{g}_\beta\in\mathbb{C}^p$ as $\widehat{f}_\beta(\alpha):=\widehat{f}(\alpha,\beta)$ and $\widehat{g}_\beta(\alpha):=\widehat{g}(-\alpha,-\beta)$ for each $\alpha\in\mathbb{Z}_p$. Recalling the definition of F_β in Equation (14), we have

$$\begin{aligned}
\langle M\pi_S f, \pi_S g\rangle &= \sum_{\beta,\beta'\in S}\sum_{\alpha,\alpha'\in\mathbb{Z}_p} \widehat{f}(\alpha,\beta)\overline{\widehat{g}(\alpha',\beta')}\langle M\chi_{\alpha,\beta}, \chi_{\alpha',\beta'}\rangle \\
&= \sum_{\beta\in S\cap(-S)}\sum_{\alpha,\alpha'\in\mathbb{Z}_p} \widehat{f}(\alpha,\beta)\overline{\widehat{g}(\alpha',-\beta)}\langle M\chi_{\alpha,\beta}, \chi_{\alpha',-\beta}\rangle \\
&= p\sum_{\beta\in S}\sum_{\alpha,\alpha'\in\mathbb{Z}_p} \widehat{f}(\alpha,\beta)\overline{\widehat{g}(\alpha',-\beta)}F_\beta(\alpha,-\alpha') \\
&= p\sum_{\beta\in S}\sum_{\alpha,\alpha'\in\mathbb{Z}_p} \widehat{f}(\alpha,\beta)\overline{\widehat{g}(-\alpha',-\beta)}F_\beta(\alpha,\alpha') \\
&= p\sum_{\beta\in S}\sum_{\alpha,\alpha'\in\mathbb{Z}_p} \widehat{f}(\alpha,\beta)\overline{\widehat{g}(-\alpha',-\beta)}F_\beta(\alpha',\alpha) \\
&= p\sum_{\beta\in S}\sum_{\alpha,\alpha'\in\mathbb{Z}_p} \widehat{f}_\beta(\alpha)\widehat{g}_\beta(\alpha')F_\beta(\alpha',\alpha) \\
&= p\sum_{\beta\in S}\langle F_\beta\widehat{f}_\beta, \widehat{g}_\beta\rangle,
\end{aligned}$$

where at the third equality, we used the negation-closed property of S. By the definition of spectral norm and Cauchy-Schwarz inequality,

$$\begin{aligned}
|\langle M\pi_S f, \pi_S g\rangle| &\le p\sum_{\beta\in S}|\langle F_\beta\widehat{f}_\beta, \widehat{g}_\beta\rangle| \le p\sum_{\beta\in S}\|F_\beta\|\|\widehat{f}_\beta\|_2\|\widehat{g}_\beta\|_2 \\
&\le p\max_{\beta\in S}\|F_\beta\|\sqrt{\sum_{\beta\in S}\|\widehat{f}_\beta\|_2^2}\sqrt{\sum_{\beta\in S}\|\widehat{g}_\beta\|_2^2} \le \frac{p}{|G|}\max_{\beta\in S}\|F_\beta\|,
\end{aligned}$$

where the last inequality follows from Parseval's identity Equation (9) and $\|f\|_2 = \|g\|_2 = 1$:

$$\sum_{\beta\in S}\|\widehat{f}_\beta\|_2^2 \le \sum_{\beta\in\mathbb{Z}_p}\|\widehat{f}_\beta\|_2^2 = \sum_{(\alpha,\beta)\in G}|\widehat{f}(\alpha,\beta)|^2 = \frac{1}{|G|}\sum_{(x,y)\in G}|f(x,y)|^2 = \frac{1}{|G|}.$$

Next, we upper-bound the spectral norm of F_β using the 4th moment of singular values:

$$\begin{aligned}
\|F_\beta\|^4 &\le \|F_\beta\|_{S_4}^4 = \mathrm{Tr}\big(F_\beta F_\beta^* F_\beta F_\beta^*\big) \\
&= \sum_{\alpha_1,\alpha_1',\alpha_2,\alpha_2'\in\mathbb{Z}_p} F_\beta(\alpha_1,\alpha_1')\overline{F_\beta(\alpha_1,\alpha_2')}F_\beta(\alpha_2,\alpha_2')\overline{F_\beta(\alpha_2,\alpha_1')} \\
&= \sum_{\substack{\alpha_1,\alpha_1'\\ \alpha_2,\alpha_2'}}\sum_{\substack{x_1,\ldots,x_4\\ y_1,\ldots,y_4}} \omega^{\alpha_1(x_1-x_2)+\alpha_2(x_3-x_4)+\alpha_1'(y_1-y_4)+\alpha_2'(y_3-y_2)}\omega^{\beta(x_1y_1-x_2y_2+x_3y_3-x_4y_4)} \\
&= \sum_{\substack{x_1,\ldots,x_4\\ y_1,\ldots,y_4}}\sum_{\substack{\alpha_1,\alpha_1'\\ \alpha_2,\alpha_2'}} \omega^{\alpha_1(x_1-x_2)+\alpha_2(x_3-x_4)+\alpha_1'(y_1-y_4)+\alpha_2'(y_3-y_2)}\omega^{\beta(x_1y_1-x_2y_2+x_3y_3-x_4y_4)}.
\end{aligned}$$

The inner sum is zero unless $x_1 = x_2$, $x_3 = x_4$, $y_1 = y_4$ and $y_2 = y_3$. This simplifies $\|F_\beta\|_{S_4}^4$ to

$$\|F_\beta\|_{S_4}^4 = p^4\sum_{x,y,x',y'\in[q]}\omega^{\beta(xy-xy'+x'y'-x'y)} = p^4 r(\beta),$$

where

$$r(\beta) \coloneqq \sum_{\vec{u}\in[q]^4}\omega^{\beta\phi(\vec{u})} \qquad \text{with} \qquad \phi(u_1,u_2,u_3,u_4) \coloneqq u_1u_2 - u_1u_4 + u_3u_4 - u_3u_2.$$

For every $z \in \mathbb{Z}_p$ and $y \in \mathbb{Z}_p \setminus \{0\}$, we have $\Pr_{x\in[q]}[xy \equiv_p z] \in \{0, 1/q\}$. Note that the event $\{\phi(\vec{u}) \equiv_p \phi(\vec{v})\}$ is equivalent to $\{u_1(u_2 - u_4) \equiv_p z\}$, where $z = v_1v_2 - v_1v_4 + v_3v_4 - v_3v_2 - u_3u_4 + u_3u_2$. Consider uniform independent random variables $\vec{u}, \vec{v} \in [q]^4$. Conditioned on $u_2 \neq u_4$, which happens with probability $1 - 1/q$, the probability that $u_1(u_2 - u_4) \equiv_p z$ is at most $1/q$. Therefore,

$$\Pr[\phi(\vec{u}) \equiv_p \phi(\vec{v})] \leq \left(1 - \frac{1}{q}\right) \times \frac{1}{q} + \frac{1}{q} \times 1 \leq \frac{2}{q},$$

implying that $|\{(\vec{u}, \vec{v}) : \phi(\vec{u}) \equiv_p \phi(\vec{v})\}| \leq 2q^7$. Hence

$$\mathop{\mathbb{E}}_{\beta} |r(\beta)|^2 = \mathop{\mathbb{E}}_{\beta} \left[\sum_{\vec{u},\vec{v}\in[q^4]} \omega^{\beta(\phi(\vec{u})-\phi(\vec{v}))} \right] = \sum_{\vec{u},\vec{v}} \mathop{\mathbb{E}}_{\beta} \left[\omega^{\beta(\phi(\vec{u})-\phi(\vec{v}))}\right] = \sum_{\vec{u},\vec{v}} \mathbf{1}_{\{\phi(\vec{u})\equiv_p\phi(\vec{v})\}} \leq 2q^7.$$

From the above inequality, for $t \coloneqq 2q^{7/2}$, by Markov's inequality we have

$$\Pr_{\beta}[|r(\beta)| \geq t] \leq \frac{2q^7}{t^2} = \frac{1}{2}.$$

As $\beta\phi(u_1, u_2, u_3, u_4) = -\beta\phi(u_1, u_4, u_3, u_2)$, we have $r(\beta) = r(-\beta)$ for any β, and so

$$S \coloneqq \{\beta \in \mathbb{Z}_p : |r(\beta)| < t\}$$

is a subset of $\mathbb{Z}_p$ closed under negation with $|S| \geq p/2$. Therefore,

$$\|M_S\| \leq \frac{p}{|G|} \max_{\beta\in S} \|F_\beta\| \leq \frac{p}{|G|} \max_{\beta\in S} \|F_\beta\|_{S_4} \leq \frac{p}{|G|} \max_{\beta\in S} \{p|r(\beta)|^{1/4}\} < t^{1/4} \leq 2q^{7/8}. \qquad \blacktriangleleft$$

By combining Lemma 16 and Lemma 17, we conclude that

$$\|M\|_{\mathrm{Tr}} \geq \|M_S\|_{\mathrm{Tr}} \geq \frac{\|M_S\|_F^2}{\|M_S\|} \geq \frac{q^2 \times p/2}{2q^{7/8}} = \Omega(pq^{9/8}).$$

6 Concluding Remarks

We showed the existence of Boolean matrices $M_{N\times N}$ with $\|M\|_{\gamma_2} \geq \Omega(N^{1/32})$ and $\mathrm{R}(M) \leq \mathrm{R}^1_0(M) \leq O(\log\log N)$, displaying a double exponential separation between γ_2 norm and randomized communication complexity. We did not attempt to optimize the power of N in the lower bound, and there is no reason to suspect that $1/32$ is the best possible.

▶ **Question 3.** *What is the largest c such that there exist Boolean matrices $M_{N\times N}$ with $\mathrm{R}(M) \leq O(\log\log N)$ and $\|M\|_{\gamma_2} \geq \Omega(N^c)$?*

It is also natural to ask the analogue of Question 3 regarding the approximate γ_2 norm.

▶ **Question 4.** *What is the largest c such that there exist Boolean matrices $M_{N\times N}$ with $\widetilde{\gamma}_2(M) \leq \mathrm{polylog}(N)$ and $\|M\|_{\gamma_2} \geq \Omega(N^c)$?*

We remark that in Question 4, one cannot hope to obtain a lower bound stronger than $\Omega(N^{1/2})$ as $\|M\|_{\gamma_2} \leq \widetilde{\gamma}_2(M) + O(\sqrt{N})$ for all M (see [12, Lemma 15]).

Whether or not the upper bounds in Question 3 and Question 4 can be improved is also an interesting open problem. As we discussed in Section 1.3, there are Boolean matrices $M_{N\times N}$ with $\mathrm{R}(M) = O(1)$ but $\|M\|_{\gamma_2} = \mathrm{polylog}(N)$.

▶ **Question 5.** *Is there a Boolean matrix* $M_{N\times N}$ *with* $\mathrm{R}(M) = O(1)$ *and* $\|M\|_{\gamma_2} = N^{\Omega(1)}$*?*

We could not overrule the possibility that PL and IIP are such examples. Nevertheless, we make the following conjecture.

▶ **Conjecture 18.** $\mathrm{R}(\mathtt{PL}) = \omega(1)$.

Another intriguing question is about the relationship between γ_2 norm and the *unbounded error* randomized communication complexity, denoted by $\mathrm{U}(\cdot)$. It is well-known [15] that $\mathrm{U}(M) = \log \mathrm{rank}_{\pm}(M) \pm O(1)$ where $\mathrm{rank}_{\pm}(\cdot)$ denotes the sign-rank a.k.a. dimension complexity. The reader is referred to [6] for the definitions of $\mathrm{U}(\cdot)$ and $\mathrm{rank}_{\pm}$. It is natural to ask whether one can obtain an upper bound on sign-rank based solely on γ_2 norm. In other words, the following conjecture is intriguing.

▶ **Conjecture 19.** *Suppose* $\|M\|_{\gamma_2} = O(1)$*. Then* $\mathrm{rank}_{\pm}(M) = O(1)$.

A viable approach to settle the above question in the positive is by using the parameter $\mathrm{D}^{\mathrm{EQ}}(\cdot)$. Hatami et al. [6] showed that if $\mathrm{D}^{\mathrm{EQ}}(M) = O(1)$, then $\mathrm{rank}_{\pm}(M) = O(1)$. On the other hand, the following was conjectured in [5], which, if true, would imply Conjecture 19.

▶ **Conjecture 20 ([5]).** *Suppose* $\|M\|_{\gamma_2} = O(1)$*. Then* $\mathrm{D}^{EQ}(M) = O(1)$.

In the special case where M is an XOR function, it is shown in [6] that Conjecture 20 is true. The authors show that this follows from Green-Sanders' quantitative version of Cohen's idempotent theorem [4].

References

1. Shalev Ben-David, Adam Bouland, Ankit Garg, and Robin Kothari. Classical lower bounds from quantum upper bounds. *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 339–349, 2018.
2. Arkadev Chattopadhyay, Shachar Lovett, and Marc Vinyals. Equality alone does not simulate randomness. In *34th Computational Complexity Conference (CCC 2019)*, 2019.
3. Alexander Edmonds, Aleksandar Nikolov, and Jonathan Ullman. The power of factorization mechanisms in local and central differential privacy. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 425–438, New York, NY, USA, June 2020. Association for Computing Machinery.
4. Ben Green and Tom Sanders. Boolean functions with small spectral norm. *Geometric and Functional Analysis*, 18(1):144–162, 2008.
5. Lianna Hambardzumyan, Hamed Hatami, and Pooya Hatami. Dimension-free bounds and structural results in communication complexity. *Israel J. Math.*, 2022. doi:10.1007/s11856-022-2365-8.
6. Hamed Hatami, Pooya Hatami, William Pires, Ran Tao, and Rosie Zhao. Lower bound methods for sign-rank and their limitations. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms a*, volume 245, pages 22:1–22:24, 2022.
7. Monika Henzinger and Jalaj Upadhyay. Constant matters: Fine-grained complexity of differentially private continual observation using completely bounded norms. *arXiv preprint*, 2022. arXiv:2202.11205.
8. Wei Huang, Yaoyun Shi, Shengyu Zhang, and Yufan Zhu. The communication complexity of the Hamming distance problem. *Inform. Process. Lett.*, 99(4):149–153, 2006.
9. Hartmut Klauck. Lower bounds for quantum communication complexity. In *Proceedings 42nd IEEE Symposium on Foundations of Computer Science*, pages 288–297. IEEE, 2001.
10. Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, Cambridge, 1997.

11 Troy Lee, Adi Shraibman, and Robert Špalek. A direct product theorem for discrepancy. In *2008 23rd Annual IEEE Conference on Computational Complexity*, pages 71–80, 2008. doi:10.1109/CCC.2008.25.

12 Nati Linial and Adi Shraibman. Lower bounds in communication complexity based on factorization norms. *Random Structures & Algorithms*, 34(3):368–394, 2009.

13 Jiri Matousek, Aleksandar Nikolov, and Kunal Talwar. Factorization norms and hereditary discrepancy. *arXiv preprint*, 2014. arXiv:1408.1376.

14 Shanmugavelayutham Muthukrishnan and Aleksandar Nikolov. Optimal private halfspace counting via discrepancy. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1285–1292, 2012.

15 Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33(1):106–123, 1986.

16 Toniann Pitassi, Morgan Shirley, and Adi Shraibman. The strength of equality oracles in communication. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 89:1–89:19, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.89.

17 Alexander A Razborov. Quantum communication complexity of symmetric predicates. *Izvestiya: Mathematics*, 67(1):145, 2003.

18 Suhail Sherif. *Communication Complexity and Quantum Optimization Lower Bounds via Query Complexity*. PhD thesis, Tata Institute of Fundamental Research, Mumbai, 2021.

19 József Solymosi. Incidences and the spectra of graphs. In *Combinatorial number theory and additive group theory*, pages 299–314. Springer, 2009.

Border Complexity of Symbolic Determinant Under Rank One Restriction

Abhranil Chatterjee
ACM Unit, Indian Statistical Institute, Kolkata, India

Sumanta Ghosh
Department of Computing and Mathematical Sciences, Caltech, Pasadena, CA, USA

Rohit Gurjar
Department of Computer Science and Engineering, IIT Bombay, India

Roshan Raj
Department of Computer Science and Engineering, IIT Bombay, India

Abstract

VBP is the class of polynomial families that can be computed by the determinant of a symbolic matrix of the form $A_0 + \sum_{i=1}^{n} A_i x_i$ where the size of each A_i is polynomial in the number of variables (equivalently, computable by polynomial-sized algebraic branching programs (ABP)). A major open problem in geometric complexity theory (GCT) is to determine whether VBP is closed under *approximation* i.e. whether $\mathsf{VBP} \stackrel{?}{=} \overline{\mathsf{VBP}}$. The power of approximation is well understood for some restricted models of computation, e.g. the class of depth-two circuits, read-once oblivious ABPs (ROABP), monotone ABPs, depth-three circuits of bounded top fan-in, and width-two ABPs. The former three classes are known to be closed under approximation [4], whereas the approximative closure of the last one captures the entire class of polynomial families computable by polynomial-sized formulas [6].

In this work, we consider the subclass of VBP computed by the determinant of a symbolic matrix of the form $A_0 + \sum_{i=1}^{n} A_i x_i$ where for each $1 \leq i \leq n$, A_i is of rank one. This class has been studied extensively [12, 13, 21] and efficient identity testing algorithms are known for it [17, 15]. We show that this class is closed under approximation. In the language of algebraic geometry, we show that the set obtained by taking coordinatewise products of pairs of points from (the Plücker embedding of) a Grassmannian variety is closed.

2012 ACM Subject Classification Theory of computation

Keywords and phrases Border Complexity, Symbolic Determinant, Valuated Matroid

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.2

Funding *Abhranil Chatterjee*: Supported by DST-INSPIRE Faculty Fellowship.

1 Introduction

The determinant polynomial plays a central role in the study of complexity theory. It is known to be a *complete polynomial* i.e. every polynomial can be computed by some affine projection of the determinant of a symbolic matrix. More precisely, for any polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$, there is some m and $A_0, A_1, \ldots, A_n$ in $\mathbb{F}^{m \times m}$ such that $f = \det_m(A_0 + \sum_{i=1}^{n} A_i x_i)$. VBP is defined as the class of polynomial families for which the size of such determinantal representation is polynomially bounded in the number of variables (equivalently, such polynomial families can be computed by polynomial-size *algebraic branching programs* (ABP)).

The other polynomial of significant interest is the permanent polynomial, a close cousin of the determinant polynomial. The permanent polynomial is also known to be a *complete polynomial*. VNP is defined as the class of polynomial families for which the size of the permanental representation is polynomially bounded in the number of variables. It is known

that $\mathsf{VBP} \subseteq \mathsf{VNP}$. The goal of algebraic complexity theory is to separate VBP and VNP, equivalently, to show a super-polynomial lower bound on the determinantal representation of the permanent polynomial.

Even though we have witnessed some outstanding progress in our understanding of the lower bound problem on various restricted models of computation in the last few years, the fundamental problem in the general setting remains elusive. *Geometric Complexity Theory* (GCT) was proposed as a possible approach to settle this question by showing $\mathsf{VNP} \not\subseteq \overline{\mathsf{VBP}}$ [19] where $\overline{\mathsf{VBP}}$ denotes the *approximative closure* of VBP. Let $\mathcal{C}$ be a circuits class over $\mathbb{F}$, $\mathbb{F}[\varepsilon]$ be the polynomial ring and $\mathbb{F}(\varepsilon)$ be the fraction field of $\mathbb{F}[\varepsilon]$. We can define $\overline{\mathcal{C}}$, the (approximative) closure of the circuit class $\mathcal{C}$ in the following equivalent ways.

(a) Approximative closure. A polynomial family $\{f_n\}$ is in the approximative closure of $\mathcal{C}$ over $\mathbb{F}$ if there is a polynomial family $\{g_n\}$ in $\mathbb{F}[\varepsilon][x_1, \ldots, x_n]$ computable in $\mathcal{C}$ over $\mathbb{F}(\varepsilon)$, such that for every n,

$$g_n(x_1, \ldots, x_n) = f_n(x_1, \ldots, x_n) + \varepsilon \cdot h_n(x_1, \ldots, x_n)$$

for some polynomial h_n in $\mathbb{F}[\varepsilon][x_1, \ldots, x_n]$. We say, the polynomial family $\{f_n\}$ is approximated by the family $\{g_n\}$.

(b) Euclidean closure. A polynomial family $\{f_n\}$ is in the Euclidean closure of $\mathcal{C}$ over $\mathbb{F}$ if, for every n, there exists an infinite sequence of polynomials $\{g_{n,i}\}$ in $\mathcal{C}$ over $\mathbb{F}$ such that the limit point of the sequence of coefficient vectors corresponding to $\{g_{n,i}\}$ is the coefficient vector of f_n. This definition is known to be equivalent to the previous definition when $\mathbb{F}$ is $\mathbb{R}$ or $\mathbb{C}$ [7].

(c) Zariski closure. Another equivalent way is to define the approximative closure as a *Zariski closure* [20]. For a circuit class $\mathcal{C}$, consider the system of all polynomial equations which are satisfied by the coefficient vector corresponding to each polynomial in $\mathcal{C}$. Then, the Zariski closure $\overline{\mathcal{C}}$ consists of the polynomials such that the corresponding coefficient vectors are satisfying assignments of the system of polynomial equations.

As all these definitions are equivalent, without loss of generality, we define $\overline{\mathcal{C}}$ to be the approximative closure of $\mathcal{C}$. If $\mathcal{C} = \overline{\mathcal{C}}$, we say $\mathcal{C}$ is closed under approximation.

One of the main objectives of geometric complexity theory is to decide whether VBP is closed under approximation or not. Showing $\mathsf{VBP} = \overline{\mathsf{VBP}}$ would imply that showing $\mathsf{VBP} \neq \mathsf{VNP}$ is equivalent to showing $\mathsf{VNP} \not\subseteq \overline{\mathsf{VBP}}$. Though the complexity of $\overline{\mathsf{VBP}}$ is not well-understood, the power of approximation has been successfully studied for various restricted models of computation. For example, it is known that the following classes are closed under approximation: (a) $\Sigma\Pi$ i.e. the sparse polynomials, (b) Monotone ABPs [4], and (c) Read-once oblivious ABPs (ROABP). Recently, the approximative closure of the depth three circuits of bounded top fan-in is shown to be contained in VBP [10]. Surprisingly, even a restricted circuit class can efficiently compute a much larger class under approximation. For example, consider VBP_2, the class of polynomials computed by the width-two ABPs. Even though, there are families of polynomials that cannot be expressed by this class [1] , the approximative closure of this class contains VF, the class of polynomials computed by a small formula. Indeed, it is known that $\overline{\mathsf{VBP}_2} = \overline{\mathsf{VF}}$ [6].

It is interesting to notice that, for the circuit classes for which the approximative closure is well-understood, we also know efficient identity testing algorithms. It motivates us to study the class VBP under some natural restriction for which we already have an efficient

identity testing algorithm. The class of our interest is the symbolic determinant under rank one restriction. Recall that any n-variate polynomial in VBP can be computed as $\det(A_0+\sum_{i=1}^{n} A_i x_i)$ where the size of each A_i is polynomially bounded in n. We consider the class of polynomials of form $\det(A_0+\sum_{i=1}^{n} A_i x_i)$ where for each $1 \leq i \leq n$, $\text{rank}(A_i) = 1$. This class has been studied extensively in contexts of polynomial identity testing, combinatorial optimization, and matrix completion (see, for example [11, 17, 21]). It admits a deterministic polynomial-time identity testing algorithm in the white-box setting [17] and a deterministic quasi-polynomial-time algorithm in the black-box setting [15]. This class is equivalent to the class of polynomial families computed by the determinant of symbolic matrices with each variable occurring at most once, also known as read-once determinants [2] (as cited in [15, Lemma 4.3]).. The expressive power of this class has also been studied. It strictly contains some well-studied classes like the polynomials computed by a small read-once formula (see, for example [3]). However, it is known that for large enough n, n-variate elementary symmetric polynomials and the permanent polynomial cannot be expressed as $\det(A_0 + \sum_{i=1}^{n} A_i x_i)$ with $\text{rank}(A_i) = 1$ for each $i \in [n]$ [3].

Another motivation to study the approximative closure of this class is the fact that the approximative closure of the orbit of this class under the action of the general linear group contains VBP [18, 24]. Therefore, understanding the approximative closure of this class may shed new light on the $\mathsf{VBP} \stackrel{?}{=} \overline{\mathsf{VBP}}$ question.

Our Results

The main result of this paper is that the class of the determinant of symbolic matrices under rank one restriction is closed under approximation. More precisely, we show the following theorem, where we use $\mathbb{F}$ to denote $\mathbb{R}$ or $\mathbb{C}$.

▶ **Theorem 1.** *Given $A_0, A_1, A_2, \ldots, A_n \in \mathbb{F}(\varepsilon)^{r\times r}$ such that for each $1 \leq i \leq n$, $\text{rank}(A_i) = 1$ over $\mathbb{F}(\varepsilon)$. Let $f = \lim_{\varepsilon\to 0} \det(A_0 + \sum_{i=1}^{n} A_i x_i)$ be defined. Then, there exists $B_0, B_1, B_2, \ldots, B_n$ in $\mathbb{F}^{(n+r)\times(n+r)}$ such that $f = \det(B_0 + \sum_{i=1}^{n} B_i x_i)$ and $\text{rank}(B_i) = 1$ over $\mathbb{F}$ for each $i \in [n]$. Moreover, if $A_0 = 0$, then the matrices $B_1, B_2, \ldots, B_n$ lie in $\mathbb{F}^{r\times r}$.*

Since this class is closed under approximation, the known hitting set and non-expressibility results for this class also hold for its approximative closure.

▶ Remark 2. By using formal power series, we can extend this result to any arbitrary field. For the sake of simplicity, we only work with $\mathbb{C}$ or $\mathbb{R}$.

An algebraic geometry perspective on the result

Consider the simpler case of Theorem 1, when $A_0 = 0$. Using known techniques, the statement can be reduced to this simpler case. Now, suppose $A_1, A_2, \ldots, A_n$ are $r \times r$ matrices of rank 1. Let us write $A_i = \mathbf{u}^i \cdot \mathbf{v}^{i\,T}$ for some vectors $\mathbf{u}^i, \mathbf{v}^i \in \mathbb{F}^r$ and define matrices $U, V \in \mathbb{F}^{r\times n}$ whose ith columns are $\mathbf{u}^i$ and $\mathbf{v}^i$, respectively. It can be verified that

$$\det(\sum_i A_i x_i) = \sum_S \det(U_S)\det(V_S)\prod_{j\in S} x_j,$$

where the sum is over all size-r subsets S of $[n]$ and U_S (or V_S) denotes the submatrix of U (or V) obtained by taking columns with indices in the set S. Hence, essentially our main result says that the image of the map

$$(\mathbb{F}^{r\times n})^2 \to \mathbb{F}^{\binom{n}{r}}, \quad (U, V) \mapsto (\det(U_S) \times \det(V_S))_S$$

is Euclidean closed (and hence, Zariski closed). A closely related map

$$\mathbb{F}^{r\times n} \to \mathbb{F}^{\binom{n}{r}}, \quad U \mapsto (\det(U_S))_S$$

has been well-studied in algebraic geometry, which gives the Plücker coordinates of elements in the Grassmannian variety. And hence, the image of this map is known to be a closed set. Putting it another way, our result says that the set obtained by taking coordinatewise products of pairs of points in the Grassmannian variety is closed.

Note that this is not a general phenomenon. It is easy to construct varieties where the set obtained by taking coordinatewise products of pairs of points from the variety is not closed. To see a simple example, consider the projective variety in $\mathbb{P}^2$ defined by

$$\{[x : y : z] \mid xz + y^2 - x^2 = 0\}.$$

Now, observe that the point $(0, 1, 0)$ cannot be obtained as a coordinatewise product of two points in the variety. On the other hand, it can be obtained as a limit of the product of two points $(\varepsilon, 1, \varepsilon - 1/\varepsilon)$ and $(1, 1, 0)$. See [5] for a related notion called Hadamard power of varieties.

Closure of a principal minor map

Our main result also implies the closure of the image of a principal minor map, as defined below. The *affine principal minor map* $\phi : \mathbb{C}^{n^2} \to \mathbb{C}^{2^n}$ is defined as

$$\phi(A) = (\det(A_I))_{I\subseteq[n]}$$

where is A_I is the principal submatrix of A with rows and columns indexed by I. Lin and Sturmfels [16] showed that for any $n > 0$, the image of ϕ on $n \times n$ matrices is closed. Our result implies the closure result for a closely related map, which we refer to as the size k principal minor map. For any $k \leq n$, let us define the map $\phi_k : \mathbb{C}^{n^2} \to \mathbb{C}^{\binom{n}{k}}$ as

$$\phi_k(A) = (\det(A_I))_{I\in\binom{[n]}{k}}$$

where $\binom{[n]}{k}$ is the set of all size-k subsets of $[n]$. We show that the image of ϕ_k on $n \times n$ rank-k matrices is closed. Formally,

▶ **Corollary 3.** *For any $n > 0$ and $k \leq n$, the image of the size k principal minor map on $n \times n$ matrices with rank at most k is closed in $\mathbb{C}^{\binom{n}{k}}$.*

One can define another similar map, where a rank-at-most-k matrix is mapped to the tuple of its size-at-most-k principal minors. Note that the closure of the image of this map follows easily from the result of Lin and Sturmfels [16]. However, to the best of our knowledge, Corollary 3 does not follow from their result.

Proof idea of the main result

As we said, our goal is to show that the image of the map

$$(U, V) \mapsto (\det(U_S) \times \det(V_S))_S$$

is closed under approximation. The idea is to start with any two given matrices $U, V \in \mathbb{F}(\varepsilon)^{r\times n}$ and construct matrices $\widehat{U}, \widehat{V} \in \mathbb{F}^{r\times n}$ such that for each size-r subset $S \subseteq [n]$, we have

$$\lim_{\varepsilon\to 0} (\det(U_S) \det(V_S)) = \det(\widehat{U}_S) \det(\widehat{V}_S).$$

Of course, we can hope to construct such matrices only when the limit exists for every S. Note that one cannot simply apply the limit operation on the matrix entries because the matrix U and V can have rational functions in ε as entries.

We can view each term like $\det(U_S)$ as a Laurent series in ε. For any Laurent series f, one can define $\mathsf{val}(f)$ as the minimum exponent of ε appearing in f. Clearly, $\lim_{\varepsilon\to 0} f$ exists if and only if $\mathsf{val}(f) \geq 0$. So let us assume that $\mathsf{val}(\det(U_S)\det(V_S)) \geq 0$ for every S. In other words,

$$\min_S\{\mathsf{val}(\det(U_S)\det(V_S))\} = \min_S\{\mathsf{val}(\det(U_S)) + \mathsf{val}(\det(V_S))\} = 0.$$

Observe that only those sets S which achieve this minimum will give a nonzero term in the limit. It would have been convenient if min operator was distributive over the sum, i.e.,

$$\min_S\{\mathsf{val}(\det(U_S)) + \mathsf{val}(\det(V_S))\} = \min_S\{\mathsf{val}(\det(U_S))\} + \min_S\{\mathsf{val}(\det(V_S))\},$$

but that is of course not true. Amazingly, it turns out that in the case of val function, it is almost true. This comes from the fact that the val function satisfies a matroid like exchange property: for any two distinct $S, T \subseteq [n]$ of size r and any $j \in T \setminus S$, there exists a $k \in S \setminus T$ such that

$$\mathsf{val}(\det(U_S)) + \mathsf{val}(\det(U_T)) \geq \mathsf{val}(\det(U_{S-k+j})) + \mathsf{val}(\det(U_{T-j+k})).$$

Based on this property, Dress and Wenzel [8] defined the so-called valuated matroids. More interestingly, Murota [22] proved the valuated matroid splitting theorem, which says that the min operator indeed distributes over the sum of two val functions, but with a "correction" term which is a linear function. To be more precise, there is a tuple $\mathbf{z} \in \mathbb{Z}^n$ such that

$$\begin{aligned}\min_S\{\mathsf{val}(\det(U_S)) + \mathsf{val}(\det(V_S))\} = &\min_S\{\mathsf{val}(\det(U_S)) + \sum_{i\in S}\mathbf{z}_i\} \\ &+ \min_S\{\mathsf{val}(\det(V_S)) - \sum_{i\in S}\mathbf{z}_i\}.\end{aligned}$$

The correction term is easy to handle because of linearity. Then basically, the problem breaks into two independent problems on U and V. That is, given any two matrices $U, V \in \mathbb{F}(\varepsilon)^{r\times n}$, construct matrices $\widehat{U}, \widehat{V} \in \mathbb{F}^{r\times n}$ such that for each size-r subset $S \subseteq [n]$, we have

$$\lim_{\varepsilon\to 0}\det(U_S) = \det(\widehat{U}_S) \text{ and } \lim_{\varepsilon\to 0}\det(V_S) = \det(\widehat{V}_S).$$

The problem now becomes tractable essentially because the image of the map $U \mapsto (\det(U_S))_S$ is known to be closed.

Discussion

As discussed earlier, showing that a class of polynomials is closed under approximation also implies that it is Zariski closed. That is, the class of polynomials must be characterized by a set of polynomial equations (in the coefficients of the polynomials in the class). It would be interesting to find the set of characterizing equations for the class of determinant of symbolic matrices under rank one restriction. Another natural class of polynomials for which we can study the closure question is that of symbolic determinant under rank 2 (or higher) restriction.

2 Preliminaries and Notations

We use $\mathbb{N}$ to denote the set of natural numbers, $\mathbb{R}$ to denote the set of real numbers, $\mathbb{C}$ to denote the set of complex numbers, $\mathbb{Z}$ to denote the set of integers, and $\mathbb{F}$ to denote field $\mathbb{R}$ or $\mathbb{C}$, respectively. For a field $\mathbb{F}$ and an indeterminate ε, $\mathbb{F}(\varepsilon)$ denotes the fractional field. For a positive integer n, $[n]$ denotes the set $\{1, 2, \dots n\}$. For a set E, 2^E denotes the family of all possible subsets of E. For a subset S of E and an element $a \in E$, $S - a$ and $S + a$ denote the set $S \setminus \{a\}$ and $S \cup \{a\}$, respectively. For any subset S of $[n]$, $\mathbf{1}_S \in \mathbb{F}^n$ denotes the characteristic vector of the subset S. For a set E and a non-negative integer r, $\binom{E}{r}$ denotes the set family consisting of all subsets of E of size r.

Every element f in the fractional field $\mathbb{F}(\varepsilon)$ is of the form g/h where $g, h \in \mathbb{F}[\varepsilon]$ with $h \neq 0$. For a nonzero polynomial $p \in \mathbb{F}[\varepsilon]$, let $\mathsf{mindeg}(p)$ be the degree of the minimum degree term in p. The function val from $\mathbb{F}(\varepsilon)$ to $\mathbb{Z}$ is defined as

$$\mathsf{val}(f) := \begin{cases} \mathsf{mindeg}(g) - \mathsf{mindeg}(h) & \text{if } f \neq 0 \\ +\infty & \text{otherwise} \end{cases}$$

▶ **Proposition 4.** *The* val *function satisfies the following properties.*

- *For any* $f, g \in \mathbb{F}(\varepsilon)$, $\mathsf{val}(fg) = \mathsf{val}(f) + \mathsf{val}(g)$.
- *For any* $f, g \in \mathbb{F}(\varepsilon)$, $\mathsf{val}(f + g) \geq \min\{\mathsf{val}(f), \mathsf{val}(g)\}$.
- *For any* $g \in \mathbb{F}(\varepsilon) \setminus \{0\}$, $\mathsf{val}(1/g) = -\mathsf{val}(g)$.
- *For an* $f \in \mathbb{F}(\varepsilon)$, $\lim_{\varepsilon \to 0} f$ *exists if and only if* $\mathsf{val}(f) \geq 0$. *Furthermore,* $\lim_{\varepsilon \to 0} f = 0$ *if and only if* $\mathsf{val}(f) > 0$.

For a polynomial $P \in \mathbb{F}(\varepsilon)[X]$ where $X = \{x_1, x_2, \dots, x_n\}$ is the set of variables, we say $\lim_{\varepsilon \to 0} P$ exists if coefficient wise limit exists for every monomial of P at $\varepsilon = 0$. In other words, for any coefficient $f \in \mathbb{F}(\varepsilon)$ of a monomial of P, $\mathsf{val}(f) \geq 0$.
For a matrix $U \in \mathbb{F}^{r \times n}$, $i \in [r]$ and $j \in [n]$, $U[i, j]$ denotes the entry at ith row and jth column of U. For a matrix $U \in \mathbb{F}^{r \times n}$ and a subset $S \subseteq [n]$, U_S denotes the submatrix of U with columns indexed by S.

Next, we describe the Cauchy-Binet formula, which is an identity for the determinant of the product of two rectangular matrices of transposed shape.

▶ **Lemma 5** (Cauchy-Binet formula, [26])**.** *Let* $n \geq r$ *be two positive integers. Let* A *and* B *are two* $r \times n$ *and* $n \times r$ *matrices over* $\mathbb{F}$*, respectively. Then*

$$\det(AB) = \sum_{S \in \binom{[n]}{r}} \det(A_S) \cdot \det(B_S),$$

where B_S *denotes the submatrix of* B *with rows indexed by* S*.*

Now we describe the Grassmann-Plücker identity.

▶ **Lemma 6** (Equation 1.3 [9])**.** *Let* $n \in \mathbb{N}$*. Let* $\mathbf{a}_0, \mathbf{a}_1, \dots, \mathbf{a}_n, \mathbf{b}_2, \mathbf{b}_3, \dots, \mathbf{b}_n$ *be* $2n$ *vectors in* $\mathbb{F}^n$*. For all* $i \in \{0, 1, \dots, n\}$*, let* U_i *and* V_i *be the matrices* $(\mathbf{a}_0, \dots, \mathbf{a}_{i-1}, \mathbf{a}_{i+1}, \dots, \mathbf{a}_n)$ *and* $(\mathbf{a}_i, \mathbf{b}_2, \dots, \mathbf{b}_n)$*, respectively. Then,*

$$\sum_{i=0}^{n} \det(U_i) \cdot \det(V_i) = 0.$$

Matroids

A matroid M is a set family $\mathcal{I}$ defined on a ground set E such that $\mathcal{I}$ satisfies the following two properties:

1. *Closure under subsets:* If $X \in \mathcal{I}$ and $Y \subset X$, then $Y \in \mathcal{I}$.
2. *Augmentation Property:* If $|X| > |Y|$ and $X, Y \in \mathcal{I}$, then there exists $x \in X \setminus Y$ such that $Y \cup \{x\} \in \mathcal{I}$.

The set family $\mathcal{I}$ is called the *independent set family* M. The augmentation property ensures that all the maximal independent sets of M have the same size. The collection $\mathcal{B}$ of all the maximal independent sets is called the *base family* of M. The base family $\mathcal{B}$ satisfies the following property:

Base exchange property: Let $B, B' \in \mathcal{B}$. Then for all $a \in B \setminus B'$ there exists a $b \in B' \setminus B$ such that $B - a + b$ is in $\mathcal{B}$.

Given the base family $\mathcal{B}$ of a matroid M, its independent set family $\mathcal{I} = \{I \mid \exists B \in \mathcal{B},\ I \subseteq B\}$. Therefore, a matroid M can be represented as $(E, \mathcal{I})$ or $(E, \mathcal{B})$. In this work, we mostly use $M = (E, \mathcal{B})$ to represent a matroid. Every matroid M is associated with a function, $\text{rank} : 2^E \to \mathbb{N}$, defined as

$$\text{rank}(S) = \max\{|Y| \mid Y \subseteq S,\ Y \in \mathcal{I}\}.$$

The rank of the ground set E is called the rank of the matroid M. It is equal to the cardinality of the bases. For more details on matroids, one can see some excellent textbooks like [23, 25].

Linear Matroids

A well-known example of matroids is the *linear matroids*. A linear matroid over a field $\mathbb{F}$ is represented by an $r \times n$ matrix U over the field $\mathbb{F}$ with the full row rank. Assume that the columns are indexed by $[n]$, which is the ground set of the matroid. Let $\mathcal{B} = \{B \subseteq [n] \mid |B| = r,\ \det(U_B) \neq 0\}$. It is not hard to prove that $M = ([n], \mathcal{B})$ is a matroid with $\mathcal{B}$ as the base family.

Matroid Intersection

Let $M_1 = (E, \mathcal{B}_1)$ and $M_2 = (E, \mathcal{B}_2)$ be two matroids defined on the same ground set E. The problem of finding a common base is called matroid intersection problem. The problem of perfect matching for bipartite graphs and many other problems can be formulated in the language of the matroid intersection problem.

In this paper, we study symbolic matrix $M = \sum_{i=1}^{n} A_i x_i$ with each A_i having rank one. Next, we give an alternate representation of such symbolic matrices.

▶ **Observation 7.** *Let $M = \sum_{i=1}^{n} A_i x_i$ where each A_i is a $r \times r$ rank one matrix over $\mathbb{F}$. Then, there exist $U, V \in \mathbb{F}^{r \times n}$ such that $M = UXV^T$ where X is the $n \times n$ diagonal matrix with x_i as its ith diagonal entry.*

Proof. Since A_i is a rank one matrix over $\mathbb{F}$, there exist $\mathbf{u}^i, \mathbf{v}^i \in \mathbb{F}^r$ such that $A_i = \mathbf{u}^i \cdot \mathbf{v}^{iT}$. Let U and V be two $r \times n$ matrices such that the ith column of U and V are $\mathbf{u}^i$ and $\mathbf{v}^i$, respectively, for all $i \in [n]$. Then, for any $p, q \in [r]$, $UXV^T[p, q] = \sum_{i=1}^{n} u_p^i v_q^i x_i = \sum_{i=1}^{n} A_i[p, q] x_i$. This implies that $UXV^T = \sum_{i=1}^{n} A_i x_i$. ◀

Valuated Matroid

Dress and Wenzel [8, 9] introduced the notion of valuated matroid. Here, we discuss it as described by Murota [22]. Suppose that $M = (E, \mathcal{B})$ is a matroid with rank r. A *valuation* on a matroid M is a function ω from $\mathcal{B}$ to $\mathbb{Z} \cup \{+\infty\}$ such that for all $B, B' \in \mathcal{B}$ and $a \in B - B'$ there exists $b \in B' - B$ such that $B - a + b \in \mathcal{B}$, $B' - b + a \in \mathcal{B}$ and

$$\omega(B) + \omega(B') \geq \omega(B - a + b) + \omega(B' - b + a) \tag{1}$$

A matroid M with a valuation function ω on it is called a *valuated matroid*, and we denote it by the 3-tuple $(E, \mathcal{B}, \omega)$. The definition of the valuated matroids by Dress and Wenzel [8, 9] and in a subsequent work by Murota [22] consider the inequality in Equation 1 in the reverse direction. The reason is that their work talks about maximization problems over valuated matroids, but for our convenience, we describe their results in terms of minimization.

For a matrix, $U \in \mathbb{F}(\varepsilon)^{r \times n}$, the following lemma defines a valuation on the linear matroid represented by U. A very similar valuation has already been studied by Dress and Wenzel [9] and by Murota [22, Example 3.2]. For an $f \in \mathbb{F}(\varepsilon)$ with $g, h \in \mathbb{F}[\varepsilon]$ and $f = g/h$, they consider $\deg_\varepsilon(f)$ instead of $\mathsf{val}(f)$ which is defined as the difference of degree of p and q.

▶ **Lemma 8.** *Let U be an $r \times n$ matrix in $\mathbb{F}(\varepsilon)^{r \times n}$ with full row rank. Let $\mathcal{B}$ be the base family of the linear matroid representable by U. Let ω be a function from $\mathcal{B}$ to $\mathbb{Z} \cup \{+\infty\}$, defined as follows: for all $B \in \mathcal{B}$,*

$$\omega(B) = \mathsf{val}(\det(U_B)).$$

Then $([n], \mathcal{B}, \omega)$ forms a valuated matroid.

Proof. We prove the above lemma using the Grassmann-Plücker identity based technique used in [9]. From Grassmann-Plücker identity (Lemma 6), for any two distinct $S, T \subseteq [n]$ of size r and any $j \in T \setminus S$,

$$\det(U_S) \cdot \det(U_T) = \sum_{i \in S \setminus T} \mu_{i,j} \det(U_{S-i+j}) \cdot \det(U_{T-j+i}),$$

where $\mu_{i,j} \in \{1, -1\}$. Then, from Proposition 4, there exists a $k \in S \setminus T$ such that

$$\mathsf{val}(\det(U_S)) + \mathsf{val}(\det(U_T)) \geq \mathsf{val}(\det(U_{S-k+j})) + \mathsf{val}(\det(U_{T-j+k})).$$

This implies that if $S, T \in \mathcal{B}$, then for any $j \in T \setminus S$ there exists $k \in S \setminus T$ such that both $T - j + k$ and $S - k + j$ are in $\mathcal{B}$ and

$$\omega(S) + \omega(T) \geq \omega(S - k + j) + \omega(T - j + k).$$

Therefore, $([n], \mathcal{B}, \omega)$ forms a valuated matroid. ◀

Suppose that $U_1 = (E, \mathcal{B}_1, \omega_1)$ and $U_2 = (E, \mathcal{B}_2, \omega_2)$ are two valuated matroids over the same ground set E. Let $\mathbf{w} : E \to \mathbb{Z}$ be a weight function. For any weight function, $\mathbf{w}$ on the ground set E, it naturally extends to all the subsets of E as follows: for any $S \subseteq E$, $\mathbf{w}(S) = \sum_{a \in S} \mathbf{w}(a)$. Then, the *valuated matroid intersection* problem asks to find a common base $B \in \mathcal{B}_1 \cap \mathcal{B}_2$ that minimizes $\mathbf{w}(B) + \omega_1(B) + \omega_2(B)$. Like Frank's weight splitting theorem for weighted matroid intersection [14], Murota [22, Theorem 4.2] gave a weight splitting theorem for the valuated matroid intersection. Here, we describe the result on the minimization version of valuated matroid intersection whose proof can be deduced from the result on the maximization version in a natural way.

▶ **Lemma 9** (Weight-splitting). *Let $U_1 = (E, \mathcal{B}_1, \omega_1)$ and $U_2 = (E, \mathcal{B}_2, \omega_2)$ be two valuated matroids and $\mathbf{w}$ be a function from E to $\mathbb{Z}$. Then, there exist $\mathbf{w}^1, \mathbf{w}^2 : E \to \mathbb{Z}$ such that a common base B minimizes $\mathbf{w}(B) + \omega_1(B) + \omega_2(B)$ if and only if the following holds:*

1. $\mathbf{w}(e) = \mathbf{w}^1(e) + \mathbf{w}^2(e)$ *for all* $e \in E$.
2. B *is a minimum weight base for the matroid* $U_1 = (E, \mathcal{B}_1)$ *with respect to* $\omega_1 + \mathbf{w}^1$.
3. B *is a minimum weight base for the matroid* $U_2 = (E, \mathcal{B}_2)$ *with respect to* $\omega_2 + \mathbf{w}^2$.

3 Proof of our closure results

In this section, we prove Theorem 1 and Corollary 3. First, we discuss some lemmas that we use in the proof of our results. One of the ingredients of our proof is the fact that the maximal minors of $r \times n$ matrices parameterize a variety (Plücker embedding of the Grassmannian). Since a variety is Euclidean closed, we get that for any $r \times n$ matrix U over $\mathbb{F}(\varepsilon)$ whose $r \times r$ minors approach a vector $\mathbf{u} \in \mathbb{F}^{\binom{n}{r}}$ as $\varepsilon \to 0$, there exists an $r \times n$ matrix $\widehat{U}$ over $\mathbb{F}$ whose $r \times r$ minors equal to $\mathbf{u}$. The next lemma shows how such a matrix $\widehat{U}$ can be constructed. For notations, see Section 2.

▶ **Lemma 10.** *Let U be a matrix in $\mathbb{F}(\varepsilon)^{r \times n}$ such that for every $S \subseteq [n]$ of size r, $\lim_{\varepsilon \to 0} \det(U_S)$ exists. Then, we can construct $\widehat{U}$ in $\mathbb{F}^{r \times n}$ such that for every $S \subseteq [n]$ of size r the following holds:*

$$\lim_{\varepsilon \to 0} \det(U_S) = \det(\widehat{U}_S).$$

Proof. First consider the trivial case when $\lim_{\varepsilon \to 0} \det(U_S)$ is zero for every $S \subseteq [n]$ of size r. In that case, $\widehat{U}$ can be defined as the matrix with all entries being zero. Now, we assume that there exists a $S \subseteq [n]$ of size r such that $\lim_{\varepsilon \to 0} \det(U_S)$ is non-zero. Without loss of generality, assume that $\lim_{\varepsilon \to 0} \det(U_{[r]})$ is nonzero. Let

$$U' = U_{[r]}^{-1} \cdot U.$$

▷ Claim 11. For every $S \subseteq [n]$ of size r, $\lim_{\varepsilon \to 0} \det(U'_S)$ exists.

Proof. Since $U' = U_{[r]}^{-1} \cdot U$, for any $S \subseteq [n]$ of size r,

$$\det(U'_S) = \det(U_{[r]}^{-1}) \cdot \det(U_S).$$

Since $\det(U_{[r]}^{-1}) = 1/\det(U_{[r]})$ and $\mathsf{val}(\det(U_{[r]})) = 0$, from Proposition 4, $\mathsf{val}(\det(U_{[r]}^{-1}))$ is also zero. Therefore, applying Proposition 4, we get that $\lim_{\varepsilon \to 0} \det(U_{[r]}^{-1})$ is non-zero. The hypothesis of the lemma ensures that $\lim_{\varepsilon \to 0} \det(U_S)$ exists. Therefore,

$$\lim_{\varepsilon \to 0} \det(U'_S) = \lim_{\varepsilon \to 0} \det(U_{[r]}^{-1}) \cdot \lim_{\varepsilon \to 0} \det(U_S).$$

This implies that $\lim_{\varepsilon \to 0} \det(U'_S)$ exists. ◁

▷ Claim 12. For every $i \in [r]$ and $j \in [n]$, $\lim_{\varepsilon \to 0} U'[i, j]$ exists.

Proof. From the definition, $U' = [I_r | A]$ where I_r is the $r \times r$ identity matrix. The claim trivially follows for $i, j \in [r]$. For an $i \in [r]$ and $j \in [n] - [r]$, let $T = [r] - \{i\} + \{j\}$, and U'_T be the matrix obtained by replacing the ith column of I_r by the jth column of U'. This implies that the matrix U'_T is of the following form:

$$U'_T = \begin{bmatrix} 1 & 0 & 0 & \dots & U'[1,j] & \dots & 0 \\ 0 & 1 & 0 & \dots & U'[2,j] & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & U'[i,j] & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & U'[r,j] & \dots & 1 \end{bmatrix}.$$

Therefore, $\det(U'_T) = U'[i,j]$. From the hypothesis of the lemma combined with Proposition 4, we know that $\mathsf{val}(\det(U'_T)) \geq 0$. Hence, $\mathsf{val}(U'[i,j]) \geq 0$. Now applying Proposition 4, $\lim_{\varepsilon\to 0} U'[i,j]$ exists. ◁

Now we define the matrix $\widetilde{U} \in \mathbb{F}^{r\times n}$ as follows: for all $i \in [r]$ and $j \in [n]$,

$$\widetilde{U}[i,j] := \lim_{\varepsilon\to 0} U'[i,j].$$

From Claim 12, the entries of the matrix $\widetilde{U}$ are well defined. Since determinant is a continuous function,

$$\lim_{\varepsilon\to 0} \det(U'_S) = \det(\widetilde{U}_S). \tag{2}$$

Let $\lim_{\varepsilon\to 0} det(U_{[r]}) = \alpha$. Consider the matrix $\widehat{U} \in \mathbb{F}^{r\times n}$ which exists by multiplying the first row of $\widetilde{U}$ by α, that is for all $i \in [r]$ and $j \in [n]$,

$$\widehat{U}[i,j] = \begin{cases} \alpha \cdot \widetilde{U}[i,j] & \text{if } i = 1 \\ \widetilde{U}[i,j] & \text{otherwise.} \end{cases}$$

The definition of $\widehat{U}$ implies that for any $S \subseteq [n]$ of size r,

$$\det(\widehat{U}_S) = \alpha \cdot \det(\widetilde{U}_S). \tag{3}$$

From the definition of U',

$$\lim_{\varepsilon\to 0} \det(U_S) = \lim_{\varepsilon\to 0} (\det(U_{[r]} \cdot \det(U'_S)).$$

Applying Claim 11, $\lim_{\varepsilon\to 0} \det(U'_S)$ exists. Therefore,

$$\begin{aligned} \lim_{\varepsilon\to 0} \det(U_S) &= \lim_{\varepsilon\to 0} \det(U_{[r]}) \cdot \lim_{\varepsilon\to 0} \det(U'_S) \\ &= \alpha \cdot \det(\widetilde{U}_S) && \text{[from Equation 2]} \\ &= \det(\widehat{U}_S) && \text{[from Equation 3].} \end{aligned}$$

This completes the proof of our lemma. ◀

Suppose that U, V are two matrices in $\mathbb{F}(\varepsilon)^{r\times n}$ with full row rank. Let $\lim_{\varepsilon\to 0}(\det(U_S) \cdot \det(V_S))$ exists for all $S \subseteq [n]$ of size r. However, the limit value of $\det(U_S)$ and $\det(V_S)$ at $\varepsilon = 0$ individually may not exist for all S. Our next lemma shows that there exists two $r \times n$ matrices $\widetilde{U}$ and $\widetilde{V}$ such that the limit value of both $\det(U_S)\cdot\det(V_S)$ and $\det(\widetilde{U}_S)\cdot\det(\widetilde{V}_S)$ at $\varepsilon = 0$ are same and also the limit value of $\det(\widetilde{U}_S)$ and $\det(\widetilde{V}_S)$ at $\varepsilon = 0$ individually exists.

For a matrix $U \in \mathbb{F}(\varepsilon)^{r\times n}$ with full row rank, let us define

$$\mathsf{minval}(U) := \min_{S\in\binom{[n]}{r}} \mathsf{val}(\det(U_S)).$$

▶ **Lemma 13.** *Let U, V in $\mathbb{F}(\varepsilon)^{r\times n}$ with full row rank. Let $\lim_{\varepsilon\to 0}\det(U_S)\cdot\det(V_S)$ exists for all $S\subseteq[n]$ of size r. Then, there exist $\widetilde{U},\widetilde{V}$ in $\mathbb{F}(\varepsilon)^{r\times n}$ such that for every $S\subseteq[n]$ of size r the following holds:*

$$\lim_{\varepsilon\to 0}\det(U_S)\det(V_S)=\left(\lim_{\varepsilon\to 0}\det(\widetilde{U}_S)\right)\cdot\left(\lim_{\varepsilon\to 0}\det(\widetilde{V}_S)\right).$$

Proof. When $\lim_{\varepsilon\to 0}\det(U_S)\det(V_S)=0$ for all $S\subseteq[n]$ of size r, the lemma is trivial to prove. Now we consider the case when there exists an $S\subseteq[n]$ of size r such that $\lim_{\varepsilon\to 0}\det(U_S)\det(V_S)\neq 0$. Next, we show that there exists a vector $\mathbf{z}\in\mathbb{Z}^n$ such that

$$\mathsf{minval}(U\cdot\mathrm{Diag}(\varepsilon^{\mathbf{z}}))+\mathsf{minval}(V\cdot\mathrm{Diag}(\varepsilon^{-\mathbf{z}}))=0,$$

where $\mathrm{Diag}(\varepsilon^{\mathbf{z}})$ is the diagonal matrix with (i,i)th entry as ε^{z_i}. Let $\mathcal{B}_1$ and $\mathcal{B}_2$ be the base families for the linear matroid represented by U and V, respectively. Let ω_1 be a function from $2^{[n]}$ to $\mathbb{Z}\cup\{+\infty\}$ defined as follows: for all $B\in 2^{[n]}$,

$$\omega_1(B)=\begin{cases}\mathsf{val}(\det(U_B)) & \text{if } B\in\mathcal{B}_1\\ +\infty & \text{otherwise}\end{cases}$$

Similarly, we can define $\omega_2:2^{[n]}\to\mathbb{Z}\cup\{+\infty\}$ for the matrix V. Now, from Lemma 8, both $([n],\mathcal{B}_1,\omega_1)$ and $([n],\mathcal{B}_2,\omega_2)$ are valuated matroids. Therefore, applying Lemma 9 with $\mathbf{w}$ as the zero function on $[n]$, there exists a weight function $\mathbf{z}:[n]\to\mathbb{Z}$ such that a common base $B\in\mathcal{B}_1\cap\mathcal{B}_2$ minimizes $\omega_1(B)+\omega_2(B)$ if and only if the following holds:

1. B is a minimum weight base for the matroid $([n],\mathcal{B}_1)$ with respect to $\omega_1+\mathbf{z}$.
2. B is a minimum weight base for the matroid $([n],\mathcal{B}_2)$ with respect to $\omega_2-\mathbf{z}$.

Abusing notation, let $\mathbf{z}$ also denote a vector in $\mathbb{Z}^n$ with ith coordinate as $\mathbf{z}(i)$. Let $U'=U\cdot\mathrm{Diag}(\varepsilon^{\mathbf{z}})$ and $V'=V\cdot\mathrm{Diag}(\varepsilon^{-\mathbf{z}})$. From the definitions, $\mathsf{minval}(U')$ is the minimum weight of a base of $([n],\mathcal{B}_1)$ with respect to $\omega_1+\mathbf{z}$. Similarly, $\mathsf{minval}(V')$ is the minimum weight of a base of $([n],\mathcal{B}_2)$ with respect to $\omega_2-\mathbf{z}$. Since for every $S\subseteq[n]$ of size r, $\lim_{\varepsilon\to 0}\det(U_S)\det(V_S)$ exists, for all $B\in\mathcal{B}_1\cap\mathcal{B}_2$, $\mathsf{val}(\det(U_B))+\mathsf{val}(\det(V_B))\geq 0$. On the other hand, from our assumption, there exists an $S\subseteq[n]$ of size r such that $\lim_{\varepsilon\to 0}\det(U_S)\det(V_S)\neq 0$. Therefore,

$$\min_{B\in\mathcal{B}_1\cap\mathcal{B}_2}\mathsf{val}(\det(U_B))+\mathsf{val}(\det(V_B))=0.$$

This implies that

$$\begin{aligned}\mathsf{minval}(U')+\mathsf{minval}(V')&=\min_{B\in\mathcal{B}_1}(\omega_1+\mathbf{z})(B)+\min_{B\in\mathcal{B}_2}(\omega_2-\mathbf{z})(B)\\&=\min_{B\in\mathcal{B}_1\cap\mathcal{B}_2}\omega_1(B)+\omega_2(B)\\&=\min_{B\in\mathcal{B}_1\cap\mathcal{B}_2}\mathsf{val}(\det(U_B))+\mathsf{val}(\det(V_B))\\&=0.\end{aligned}$$

Let $c=\mathsf{minval}(U')=-\mathsf{minval}(V')$. Let $\widetilde{U}$ and $\widetilde{V}$ be the matrix obtained by multiplying the first row of U' and V' by ε^{-c} and ε^{c}, respectively. Thus, for all $S\subseteq[n]$ of size r, we have that

$$\det(U_S)\cdot\det(V_S)=\det(U'_S)\cdot\det(V'_S)=\det(\widetilde{U}_S)\cdot\det(\widetilde{V}_S),$$

and $\mathsf{minval}(\widetilde{U})=\mathsf{minval}(U')-c=0$. Similarly, $\mathsf{minval}(\widetilde{V})=0$. This implies that for all $S\subseteq[n]$ of size r,

$$\lim_{\varepsilon\to 0}\det(U_S)\cdot\det(V_S)=\left(\lim_{\varepsilon\to 0}\det(\widetilde{U}_S)\right)\cdot\left(\lim_{\varepsilon\to 0}\det(\widetilde{V}_S)\right).$$ ◀

3.1 Proof of Theorem 1

In this subsection, we give the proof of Theorem 1. First, we prove for the case when $A_0 = 0$. From Observation 7, we get $U, V \in \mathbb{F}(\varepsilon)^{r\times n}$ such that $\sum_{i=1}^{n} A_i x_i = UXV^T$ where X is the diagonal matrix with x_i as its ith diagonal entry. Abusing notation, we use X_S to denote $\prod_{i\in S} x_i$. Therefore,

$$\begin{aligned} f &= \lim_{\varepsilon\to 0} \det\left(\sum_{i=1}^{n} A_i x_i\right) = \lim_{\varepsilon\to 0} \det(UXV^T) \\ &= \lim_{\varepsilon\to 0} \sum_{S\subseteq[n],|S|=r} \det(U_S)\det(V_S)X_S \qquad \text{[from Lemma 5]} \\ &= \sum_{S\subseteq[n],|S|=r} \left(\lim_{\varepsilon\to 0} \det(U_S)\det(V_S)\right) X_S. \end{aligned}$$

In the last equality above, we can take the limit inside as f is defined if and only if the limit exists for the coefficient of every monomial. Applying Lemma 13,

$$f = \sum_{S\subseteq[n],\,|S|=r} \left(\lim_{\varepsilon\to 0} \det(\widetilde{U}_S)\right)\left(\lim_{\varepsilon\to 0} \det(\widetilde{V}_S)\right) X_S.$$

From Lemma 10, we have two $r \times n$ matrices $\widehat{U}$ and $\widehat{V}$ in $\mathbb{F}^{r\times n}$ such that

$$\begin{aligned} f &= \sum_{S\subseteq[n],\,|S|=r} \det(\widehat{U}_S)\det(\widehat{V}_S)X_S \\ &= \det(\widehat{U}X\widehat{V}^T). \end{aligned}$$

For all $i \in [n]$, let B_i be the $r \times r$ rank one matrix defined as $\widehat{U}[i] \cdot \widehat{V}[i]^T$, where $\widehat{U}[i]$ and $\widehat{V}[i]$ are the ith columns of $\widehat{U}$ and $\widehat{V}$ respectively. Then,

$$f = \det(\widehat{U}X\widehat{V}^T) = \det(\sum_{i=1}^{n} B_i x_i).$$

This completes the proof of Theorem 1 where $A_0 = 0$.

For the case when $A_0 \neq 0$, we first give the following lemma that essentially reduces it to the previous case. This particular proof idea comes from Anderson, Shpilka, and Volk [2] (see [15, Lemma 4.3]).

For positive integers m and n, let I_n denote the $n \times n$ identity matrix and $0_{m,n}$ denote the $m \times n$ rectangular matrix with all zeros.

▶ **Lemma 14.** *Let $P = \det\left(A_0 + UXV^T\right)$ for some U, V in $\mathbb{F}(\varepsilon)^{r\times n}$, $A_0 \in \mathbb{F}(\varepsilon)^{r\times r}$ and X is an $n \times n$ diagonal matrix with $x_1, x_2, \ldots, x_n$ in the diagonal. Let X' be a $(2n+r)\times(2n+r)$ diagonal matrix with $x_1, x_2, \ldots x_{2n+r}$ in the diagonal. Then, there exist rectangular matrices $U', V' \in \mathbb{F}(\varepsilon)^{(n+r)\times(2n+r)}$ such that the following holds:*

- *Let Q be the polynomial in $x_1, x_2, \ldots x_n$ obtained by putting $x_{n+1}, \ldots, x_{2n+r}$ equal to 1 in $\det(U'X'V'^T)$. Then, $P = Q$.*
- *If $\lim_{\varepsilon\to 0} P$ exists, then $\lim_{\varepsilon\to 0} \det(U'X'V'^T)$ also exists.*

Proof. Let us define

$$U' = \left[\begin{array}{c|c|c} 0_{n,n} & I_n & V^T \\ \hline -U & 0_{r,n} & A_0 \end{array}\right] \quad \text{and, } V' = \left[\begin{array}{c|c|c} I_n & I_n & 0_{n,r} \\ \hline 0_{r,n} & 0_{r,n} & I_r \end{array}\right].$$

Let X_1 be a $n \times n$ diagonal matrix with $x_{n+1}, \ldots, x_{2n}$ in the diagonal and X_2 be a $r \times r$ diagonal matrix with $x_{2n+1}, \ldots, x_{2n+r}$ in the diagonal. We now consider $U'X'V'^T$. Notice that,

$$U'X'V'^T = \left[\begin{array}{c|c|c} 0_{n,n} & X_1 & V^TX_2 \\ \hline -UX & 0_{r,n} & A_0X_2 \end{array}\right] \cdot \left[\begin{array}{c|c} I_n & 0_{n,r} \\ \hline I_n & 0_{n,r} \\ \hline 0_{r,n} & I_r \end{array}\right] = \begin{bmatrix} X_1 & V^TX_2 \\ -UX & A_0X_2 \end{bmatrix}.$$

Let A, B, C, D be matrices where A and D are square matrices and A is invertible. Then, we have

$$\det\begin{pmatrix} A & B \\ C & D \end{pmatrix} = \det(A).\det(D - CA^{-1}B)$$

Therefore,

$$\begin{aligned} \det(U'X'V'^T) &= \det(X_1).\det(A_0X_2 + UXX_1^{-1}V^TX_2) \\ &= \det(X_1).\det(A_0 + UXX_1^{-1}V^T).\det(X_2). \end{aligned}$$

It is easy to see that if we put the value of 1 to $x_{n+1}, \ldots, x_{2n+r}$, we get $\det(A_0 + UXV^T)$. Also,

$$\lim_{\varepsilon\to 0} \det(U'X'V'^T) = \det(X_1).\det(X_2).\lim_{\varepsilon\to 0} \det(A_0 + UXX_1^{-1}V^T).$$

The second part of the lemma follows from the fact that if $\lim_{\varepsilon\to 0} P$ exists, then $\lim_{\varepsilon\to 0} \det(A_0 + UXX_1^{-1}V^T)$ also exists as XX_1^{-1} can be treated as a diagonal matrix with a different set of indeterminates. ◀

Now we prove for the case when $A_0 \neq 0$. Let $f = \det(A_0 + UXV^T)$ and $f' = \det(U'X'V'^T)$. From the Lemma 14, $\lim_{\varepsilon\to 0} f'$ exists as it is given that $\lim_{\varepsilon\to 0} f$ exists. Just like we discussed above for the case of $A_0 = 0$, we can get $\widehat{U}', \widehat{V}' \in \mathbb{F}^{(n+r)\times(2n+r)}$ such that $\lim_{\varepsilon\to 0} f' = \det(\widehat{U}'X'\widehat{V}'^T)$. For all $i \in [2n+r]$, let B_i be the $(n+r)\times(n+r)$ rank one matrix defined as $\widehat{U}'[i] \cdot \widehat{V}'[i]^T$, where $\widehat{U}'[i]$ and $\widehat{V}'[i]$ are the ith columns of $\widehat{U}'$ and $\widehat{V}'$ respectively. Hence, $\lim_{\varepsilon\to 0} f' = \det(\sum_{i=1}^{2n+r} B_ix_i)$. Let $\sum_{i=n+1}^{2n+r} B_i = B_0$. From the first part of Lemma 14, $\lim_{\varepsilon\to 0} f = \det(B_0 + \sum_{i=1}^{n} B_ix_i)$.

3.2 Proof of Corollary 3

We will show the following lemma which directly implies Corollary 3.

▶ **Lemma 15.** *Let $A \in \mathbb{C}(\varepsilon)^{n\times n}$ be a matrix of rank at most k and $A[S]$ denote the minor of A whose rows and columns are indexed by $S \subseteq [n]$. Let $\lim_{\varepsilon\to 0} A[S]$ exist for all subset $S \subset [n]$ of size k. Then, there exists $B \in \mathbb{C}^{n\times n}$ such that for all $S \subset [n]$ of size k,*

$$\lim_{\varepsilon\to 0} A[S] = B[S]$$

Proof. The claim is trivial when $\text{rank}(A) < k$ as all the minors are zero. Hence, we assume that $\text{rank}(A) = k$. Let $U, V \in \mathbb{C}(\varepsilon)^{k\times n}$ such that U^T, V is a rank-factorization of A. This implies that $A = U^T.V$ and for any subset $S \subset [n]$, $A[S] = \det(U_S^T.V_S)$. Since

$\lim_{\varepsilon\to 0} A[S] = \lim_{\varepsilon\to 0} \det(U_S)\det(V_S)$ exists for all $S \subset [n]$ of size k, from Lemma 13 there exists $\widetilde{U}, \widetilde{V}$ in $\mathbb{C}(\varepsilon)^{r\times n}$ such that for every $S \subseteq [n]$ of size k the following holds:

$$\lim_{\varepsilon\to 0} A[S] = \left(\lim_{\varepsilon\to 0} \det(\widetilde{U}_S)\right) \cdot \left(\lim_{\varepsilon\to 0} \det(\widetilde{V}_S)\right).$$

From Lemma 10, there exist two $k \times n$ matrices $\widehat{U}$ and $\widehat{V} \in \mathbb{C}^{k\times n}$ such that for all $S \subset [n]$, the following holds:

$$\lim_{\varepsilon\to 0} \det(\widetilde{U}_S) = \det(\widehat{U}_S) \text{ and } \lim_{\varepsilon\to 0} \det(\widetilde{V}_S) = \det(\widehat{V}_S)$$

Let $B = \widehat{U}^T.\widehat{V}$. Hence, for all $S \subset [n]$ of size k,

$$\lim_{\varepsilon\to 0} A[S] = \det(\widehat{U}_S^T).\det(\widehat{V}_S) = \det(\widehat{U}_S^T.\widehat{V}_S) = B[S]. \quad \blacktriangleleft$$

References

1. Eric Allender and Fengming Wang. On the power of algebraic branching programs of width two. *computational complexity*, 25:217–253, March 2016. doi:10.1007/s00037-015-0114-7.
2. Matthew Anderson, Amir Shpilka, and Ben Lee Volk. Personal communication, 2016.
3. N. R. Aravind and Pushkar S. Joglekar. On the expressive power of read-once determinants. In Adrian Kosowski and Igor Walukiewicz, editors, *Fundamentals of Computation Theory – 20th International Symposium, FCT 2015, Gdańsk, Poland, August 17-19, 2015, Proceedings*, volume 9210 of *Lecture Notes in Computer Science*, pages 95–105. Springer, 2015. doi:10.1007/978-3-319-22177-9_8.
4. Markus Bläser, Christian Ikenmeyer, Meena Mahajan, Anurag Pandey, and Nitin Saurabh. Algebraic branching programs, border complexity, and tangent spaces. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 21:1–21:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.21.
5. Cristiano Bocci and Enrico Carlini. Hadamard products of hypersurfaces. *Journal of Pure and Applied Algebra*, 226(11):107078, 2022. doi:10.1016/j.jpaa.2022.107078.
6. Karl Bringmann, Christian Ikenmeyer, and Jeroen Zuiddam. On algebraic branching programs of small width. *Journal of the ACM*, 65, February 2017. doi:10.1145/3209663.
7. Peter Bürgisser. The complexity of factors of multivariate polynomials. *Found. Comput. Math.*, 4(4):369–396, November 2004.
8. Andreas W.M. Dress and Walter Wenzel. Valuated matroids: a new look at the greedy algorithm. *Applied Mathematics Letters*, 3(2):33–35, 1990. doi:10.1016/0893-9659(90)90009-Z.
9. Andreas W.M Dress and Walter Wenzel. Valuated matroids. *Advances in Mathematics*, 93(2):214–250, 1992. doi:10.1016/0001-8708(92)90028-J.
10. Pranjal Dutta, Prateek Dwivedi, and Nitin Saxena. Demystifying the border of depth-3 algebraic circuits. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 92–103, 2022. doi:10.1109/FOCS52979.2021.00018.
11. Jack Edmonds. Systems of distinct representatives and linear algebra. *Journal of research of the National Bureau of Standards*, 71:241–245, 1967.
12. Jack Edmonds. Matroid partition. *Mathematics of the Decision Sciences*, 11:335–345, 1968.
13. Jack Edmonds. Matroid intersection. In P.L. Hammer, E.L. Johnson, and B.H. Korte, editors, *Discrete Optimization I*, volume 4 of *Annals of Discrete Mathematics*, pages 39–49. Elsevier, 1979. doi:10.1016/S0167-5060(08)70817-3.
14. András Frank. A weighted matroid intersection algorithm. *Journal of Algorithms*, 2(4):328–336, 1981. doi:10.1016/0196-6774(81)90032-8.

15 Rohit Gurjar and Thomas Thierauf. Linear matroid intersection is in quasi-nc. *Comput. Complex.*, 29(2):9, 2020. doi:10.1007/s00037-020-00200-z.

16 Shaowei Lin and Bernd Sturmfels. Polynomial relations among principal minors of a 4×4-matrix. *Journal of Algebra*, 322(11):4121–4131, 2009. Computational Algebra. doi:10.1016/j.jalgebra.2009.06.026.

17 László Lovász. Singular spaces of matrices and their application in combinatorics. *Boletim da Sociedade Brasileira de Matemática*, 20:87–99, October 1989. doi:10.1007/BF02585470.

18 Dori Medini and Amir Shpilka. Hitting sets and reconstruction for dense orbits in vp_{e} and ΣΠΣ circuits. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 19:1–19:27. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.19.

19 Ketan D. Mulmuley and Milind Sohoni. Geometric complexity theory i: An approach to the p vs. np and related problems. *SIAM Journal on Computing*, 31(2):496–526, 2001. doi:10.1137/S009753970038715X.

20 David Mumford. *Algebraic Geometry I.* Springer Berlin, Heidelberg, 1995.

21 Kazuo Murota. Mixed matrices: Irreducibility and decomposition. In Richard A. Brualdi, Shmuel Friedland, and Victor Klee, editors, *Combinatorial and Graph-Theoretical Problems in Linear Algebra*, pages 39–71, New York, NY, 1993. Springer New York.

22 Kazuo Murota. Valuated matroid intersection i: Optimality criteria. *SIAM Journal on Discrete Mathematics*, 9(4):545–561, 1996. doi:10.1137/S0895480195279994.

23 James G. Oxley. *Matroid Theory (Oxford Graduate Texts in Mathematics)*. Oxford University Press, Inc., New York, NY, USA, 2006.

24 Chandan Saha and Bhargav Thankey. Hitting sets for orbits of circuit classes and polynomial families. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2021, August 16-18, 2021, University of Washington, Seattle, Washington, USA (Virtual Conference)*, volume 207 of *LIPIcs*, pages 50:1–50:26. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.APPROX/RANDOM.2021.50.

25 Alexander Schrijver. *Combinatorial optimization : polyhedra and efficiency. Vol. B. , Matroids, trees, stable sets. chapters 39-69.* Algorithms and combinatorics. Springer-Verlag, Berlin, Heidelberg, New York, N.Y., et al., 2003. URL: http://opac.inria.fr/record=b1124843.

26 Jiang Zeng. A bijective proof of Muir's identity and the Cauchy-Binet formula. *Linear Algebra and its Applications*, 184:79–82, 1993.

On Correlation Bounds Against Polynomials

Peter Ivanov
Northeastern University, Boston, MA, USA

Liam Pavlovic
Northeastern University, Boston, MA, USA

Emanuele Viola
Northeastern University, Boston, MA, USA

Abstract

We study the fundamental challenge of exhibiting explicit functions that have small correlation with low-degree polynomials over $\mathbb{F}_2$. Our main contributions include:

1. In STOC 2020, CHHLZ introduced a new technique to prove correlation bounds. Using their technique they established new correlation bounds for low-degree polynomials. They conjectured that their technique generalizes to higher degree polynomials as well. We give a counterexample to their conjecture, in fact ruling out weaker parameters and showing what they prove is essentially the best possible.
2. We propose a new approach for proving correlation bounds with the central "mod functions," consisting of two steps: (I) the polynomials that maximize correlation are symmetric and (II) symmetric polynomials have small correlation. Contrary to related results in the literature, we conjecture that (I) is true. We argue this approach is not affected by existing "barrier results."
3. We prove our conjecture for quadratic polynomials. Specifically, we determine the maximum possible correlation between quadratic polynomials modulo 2 and the functions $(x_1, \ldots, x_n) \to z^{\sum x_i}$ for any z on the complex unit circle, and show that it is achieved by symmetric polynomials. To obtain our results we develop a new proof technique: we express correlation in terms of directional derivatives and analyze it by slowly restricting the direction.
4. We make partial progress on the conjecture for cubic polynomials, in particular proving tight correlation bounds for cubic polynomials whose degree-3 part is symmetric.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases Correlation bounds, Polynomials

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.3

Related Version *Previous Version*: https://eccc.weizmann.ac.il/report/2022/092/

Funding NSF grant CCF-2114116
Liam Pavlovic: Northeastern REU supplement

Acknowledgements We are grateful to Brenden Collins for collaborating during the initial stages of this project.

1 Introduction and our results

Exhibiting explicit functions that have small *correlation* with low-degree polynomials modulo 2 is a fundamental challenge in complexity theory, cf. the recent survey [33]. This challenge is generally referred to as "proving correlation bounds" and progress on it is a prerequisite for progress on a striking variety of other long-standing problems: circuit lower bounds [29, 30], Valiant's rigidity challenge [32], number-on-forehead communication complexity [32, 30], and even recently-made conjectures on the Fourier spectrum of low-degree polynomials [31].

After many years, the state-of-the-art on this challenge has not changed much since seminal works from at least thirty years ago. Two bounds are known for degree d polynomials. First, the results by Razborov and Smolensky from the 80's give correlation $O(d/\sqrt{n})$ [22, 24, 25]; second, the result by Babai, Nisan, and Szegedy [3] on number-on-forehead communication protocols yields correlation $\exp(-\Omega(n/d2^d))$. A slight improvement to $\exp(-\Omega(n/2^d))$ appears in [27]. Thus, the first bound applies to large degrees but yields weak correlation, while the second bound yields exponentially small correlation, but only applies to degrees less than $\log n$. Achieving correlation less than $1/\sqrt{n}$ for polynomials of degree $\log n$ remains open, for any explicit function. Remarkably, solving this specific setting of parameters is required for long-sought progress on any of the challenges mentioned in the previous paragraph.

1.1 The conjecture and our first result

In STOC 2020, Chattopadhyay, Hatami, Hosseini, Lovett, and Zuckerman [11]. introduced a novel technique which they established new correlation bounds for low-degree polynomials. The key ingredient in their approach is a structural result about the Fourier spectrum of low-degree polynomials over $\mathbb{F}_2$. They show that for any n-variate polynomial p over $\mathbb{F}_2$ of degree $\leq d$, there is a set S of variables such that almost all of the Fourier mass of p lies on Fourier coefficients that intersect with S, and the size of S is exponential in d. Further, they conjecture that the size of S needs to be just polynomial in d.

We give a counterexample to their conjecture. In fact, we shall rule out weaker parameters and show what they prove is essentially the best possible. This appears in Section 2.

1.2 Mod functions

A natural candidate for achieving small correlation are the Mod_ϕ functions which map inputs of Hamming weight w to the complex point on the unit circle with angle $w\phi$. These Mod_ϕ are closely related to the boolean mod m functions which indicate if the input Hamming weight is divisible by m. Specifically, one can bound the correlation with mod m for odd m by the correlations with the Mod_ϕ functions for $\phi = 2\pi k/m$ for $k = 1, 2, \ldots, (m-1)/2$ (see Lemma 36). In turn, as discussed below, an early motivation for studying the correlation with mod m was proving circuit lower bounds.

We now formally define these notions and then discuss previous results.

▶ **Definition 1.** *For any angle $\phi \in [0, 2\pi]$ the function $Mod_\phi \colon \{0,1\}^n \to \mathbb{C}$ is defined as*

$$Mod_\phi(x) := e^{\phi\sqrt{-1}\sum_i x_i}.$$

The correlation of a polynomial $p : \{0,1\}^n \to \{0,1\}$ with Mod_ϕ is

$$C_\phi(p) := \left|\mathbb{E}_{x\in\{0,1\}^n}(-1)^{p(x)} Mod_\phi(x)\right|.$$

For any integer m we define the boolean Mod m function $BMod_m : \{0,1\}^n \to \{0,1\}$ as

$$BMod_m(x) := \begin{cases} 1 & \text{if } \sum_{i=1}^n x_i \neq 0 \bmod m \\ 0 & \text{if } \sum_{i=1}^n x_i = 0 \bmod m. \end{cases}$$

The correlation between a polynomial $p : \{0,1\}^n \to \{0,1\}$ and $BMod_m$ is:

$$B_m(p) := \left|\mathbb{E}_{x:BMod_m(x)=0}(-1)^{p(x)} - \mathbb{E}_{x:BMod_m(x)=1}(-1)^{p(x)}\right|.$$

Most or all of the works in this area, including this paper, is concerned with the Mod_ϕ functions. And most of the works use correlation bounds with Mod_ϕ functions for various ϕ to obtain corresponding correlation bounds with the mod m functions. In particular, the two correlation bounds stated above hold for $Mod_{2\pi/3}$. The first bound essentially appears in Smolensky's paper. For the second bound, Bourgain first proved [9] correlation $\exp(-\Omega(n/c^d))$ with $Mod_{2\pi/m}$, with a correction in [16]. Nisan later pointed out that such bounds also follow from [3]. The constant c is optimized to 4 in [27]. For more discussion and background we refer to the survey [33], where the reader may find proofs of both bounds, including Nisan's derivation from [3].

1.2.1 Exact results

Unlike other models of computation such as circuits, polynomials seem simple enough that one may try to obtain *exact* results. That is, one may try to precisely characterize the polynomials that achieve the maximum correlation. Twenty years ago, a remarkable paper by Green [13], which is an inspiration for this work, took precisely such a step. Green, and the subsequent work [15], precisely characterized the quadratic polynomials modulo *three* that achieve the maximum correlation with the $Mod_{2\pi/2}$ function, i.e., parity. Compared to our discussion above, the moduli in [13] are swapped. Green considers polynomials modulo 3 instead of 2, and bounds the correlation with $Mod_{2\pi/2}$ instead of $Mod_{2\pi/3}$. Extending Green's result to other moduli has resisted attacks, see [13, 12]. While these works do not explicitly consider polynomials modulo 2, difficulties also arise trying to port Green's proof to our setting. In fact, jumping ahead, we will show that the answer is different, arguably explaining the difficulties.

1.2.2 Are symmetric polynomials optimal?

Aiming for exact results, a natural question to ask is whether, for some fixed degree, the polynomials modulo 2 that have maximum correlation with Mod_ϕ are *symmetric*. Indeed, this question has been asked by many authors; it appears explicitly for example in the 2001 paper by Alon and Beigel [2]. A positive answer would have dramatic consequences since symmetric polynomials modulo 2, even of large degree, have exponentially small correlation with, say, $Mod_{2\pi/3}$. Thus, if one could prove that symmetric polynomials correlate best, one would obtain long-sought correlation bounds.

However, until now the evidence for this has been negative. The maximizing polynomials in [13, 15] are *not* symmetric. Moreover, the work [14] has shown that for a large range of parameters, symmetric polynomials modulo 3 do *not* correlate best with parity (and are not even close). One of the families of polynomials that are shown to outperform symmetric in these works is that of *block-symmetric* polynomials, which are sums of symmetric polynomials on disjoint sets of variables. However, naive conjectures regarding the optimality of block-symmetric or other families of polynomials fail, and we are not aware of any natural family of polynomials modulo 3 that is a candidate to maximizing correlation with parity. The only available evidence that symmetric polynomials correlate best with mod functions are computer experiments up to 10 variables reported in [14].

1.3 A new approach

Departing from previous proofs, in this work we propose the following approach to proving correlation bounds with mod functions. It consists of two steps:

(I) Prove that symmetric polynomials correlate best with mod functions, and
(II) Prove that symmetric polynomials have exponentially small correlation with mod functions.

Regarding (I), we put forth the following conjecture:

▶ **Conjecture 2.** *For every d, n, ϕ degree-d symmetric polynomials correlate best with the Mod_ϕ function on n bits.*

We verify (II) in Section 7. The result is folklore. We remark that [10] proves a similar result, but in the case of symmetric polynomials mod m and the mod 2 function. However, changing moduli can yield different results, as shown by this paper.

1.3.1 Our approach vs. "barriers" to lower bounds

Over the years many "barriers" have been proposed for progress on lower bounds. Barriers based on oracles or relativization [4, 1] are not known to apply – they mostly concern uniform models of computation. The Natural Proofs barrier [23] (see also [20, 19]) is also not known to apply since we do not have candidate pseudorandom functions that correlate with low-degree polynomials.

More recently, Bhowmick and Lovett [7] proposed a new barrier specifically for proving correlation bounds. They consider an extension of polynomials called *non-classical polynomials,* an object first introduced in [26]. In short, in a non-classical polynomial of degree d monomials can have rational coefficients (with denominators depending on the degree) and the output of the polynomial is considered as an element in the torus $[0, 1]$. The work [7] shows that the proofs of most correlation bounds (such as those mentioned at the beginning of this introduction) also apply to non-classical polynomials. Moreover, for non-classical polynomials these bounds are actually tight! For example, there are non-classical polynomials of degree just $O(\log n)$ that correlate well with mod functions.

We argue that non-classical polynomials do not constitute an obstacle for our approach above. The main reason is that the non-classical polynomials in [7] – including those for mod functions – are actually symmetric. Hence, one could conceivably prove (I) above without distinguishing classical from non-classical polynomials. Moreover, the proof of (II) above already distinguishes classical from non-classical polynomials.

1.4 Our second result: Proof of Conjecture 2 for $d = 2$

A main technical contribution of this work is a proof of our Conjecture 2 in the case of degree two. That is, in contrast with the previous proofs discussed above, we show that, among quadratic polynomials modulo 2, those that correlate best with the Mod_ϕ functions are symmetric. Let us first define the elementary symmetric polynomials of degree 1 and 2.

▶ **Definition 3** (Elementary symmetric polynomials)**.** *Let*

$$e^1(x_1, \ldots, x_n) := \sum_{i=1}^{n} x_i,$$

$$e^2(x_1, \ldots, x_n) := \sum_{i<j}^{n} x_i x_j.$$

▶ **Example 4.** Let $\phi = 2\pi/3$ and $\omega = e^{\phi\sqrt{-1}}$. We have:

$$C_\phi(0) = \left|\mathop{\mathbb{E}}_{x\in\{0,1\}^n} \omega^{\sum_i x_i}\right| = \left|\mathop{\mathbb{E}}_{x_1\in\{0,1\}} \omega^{x_1}\right|^n = \left|\frac{1+\omega}{2}\right|^n = \left(\frac{1+\cos\phi}{2}\right)^{n/2} = \left(\frac{1}{2}\right)^n,$$

$$C_\phi(e^1) = \left|\mathop{\mathbb{E}}_{x\in\{0,1\}^n} (-1)^{\sum_i x_i}\omega^{\sum_i x_i}\right| = \left|\mathop{\mathbb{E}}_{x_1\in\{0,1\}} (-1)^{x_1}\omega^{x_1}\right|^n = \left|\frac{1-\omega}{2}\right|^n = \left(\frac{1-\cos\phi}{2}\right)^{n/2}$$
$$= \left(\frac{\sqrt{3}}{2}\right)^n,$$

$C_\phi(BMod_3) \geq 1/2,$

where the last inequality follows because the absolute value of the real component of $(-1)^{BMod_3(x)}\omega^{\sum_i x_i}$ is $\geq 1/2$ for every x.

We next state our result. Henceforth all polynomials in this paper have coefficients in $\{0,1\}$ and operate modulo two. We characterize the quadratic polynomials that maximize C_ϕ *for any angle* $\phi \in [0, 2\pi]$. Additionally, we show the correlation of other quadratic polynomials is a multiplicative factor smaller.

It is in fact sufficient to restrict our attention to angles $\phi \in [0, \pi/2]$ thanks to a simple symmetry argument presented in Section 3. When $\phi \in [0, \pi/4]$ then the constant zero polynomial maximizes correlation. Our main contribution is that when $\phi \in (\pi/4, \pi/2]$ the correlation is maximized by either e^2 or $e^2 + e^1$, depending on the value of n mod 4.

We define the quantity

$$v_\phi := 2^{-n-1} \cdot ((1+\sin\phi)^n + (1-\sin\phi)^n)$$

which plays a key role in this paper.

▶ **Theorem 5.** *Fix any angle $\phi \in [0, \pi/2]$. For all large enough n, the maximum $C_\phi(p)$ over quadratic polynomials p is attained by a symmetric polynomial. In more detail:*

1. *Suppose $\phi \in (\pi/4, \pi/2]$.*
   - **a.** *For n even we have $C_\phi(e^2) = C_\phi(e^2 + e^1) = \sqrt{v_\phi}$.*
   - **b.** *For $n \equiv 1 \bmod 4$ we have $C_\phi(e^2) = \sqrt{v_\phi + (\cos(\phi)/2)^n}$, $C_\phi(e^2 + e^1) = \sqrt{v_\phi - (\cos(\phi)/2)^n}$.*
   - **c.** *For $n \equiv 3 \bmod 4$ we have $C_\phi(e^2) = \sqrt{v_\phi - (\cos(\phi)/2)^n}$, $C_\phi(e^2 + e^1) = \sqrt{v_\phi + (\cos(\phi)/2)^n}$.*
   - **d.** *For any quadratic polynomial p besides e^2, $e^2 + e^1$ we have $C_\phi(p) \leq \sqrt{1 - \Omega(\sin\phi - \cos\phi)} \cdot \sqrt{v_\phi}$.*
2. *Suppose $\phi \in [0, \pi/4]$. Then $C_\phi(0) = \left(\frac{1+\cos\phi}{2}\right)^{n/2}$ and for any quadratic polynomial $p \neq 0$ we have $C_\phi(p) \leq (1 - \Omega(1)) \cdot C_\phi(0)$.*

Note that $\sqrt{v_\phi - (\cos(\phi)/2)^n} \geq (1 - o(1))\sqrt{v_\phi}$ and so the theorem shows that the correlation of non-symmetric polynomials is a constant-factor smaller than optimal.

An important message of this paper is that C_ϕ is maximized by *symmetric* polynomials. This contrasts with previous works, and gives hope that this may hold for larger degrees as well. If that is the case one would obtain long-sought correlation bounds, as discussed previously.

1.4.1 Results and directions for $d = 3$

We conjecture that Theorem 5 can be extended to show that for any cubic polynomial p and any ϕ, $C_\phi(p) \leq \max_{s\in\{0,e^1,e^2,e^2+e^1\}} C_\phi(s)$. In other words, the correlation over all cubic polynomials is still maximized by a quadratic symmetric. This would prove Conjecture 2 for $d = 3$ as well.

We make progress on this conjecture by proving this indeed holds when p is the sum of an arbitrary quadratic polynomial and a symmetric degree-3 polynomial. This is done in Section 8.

1.5 Boolean correlation

We now turn our attention to the boolean $BMod_m$ function. As mentioned earlier, most or all papers bounding the corresponding correlation B_m, including this one, proceed by first bounding C_ϕ for several corresponding values of ϕ and then using that information to bound B_m. Indeed, C_ϕ is a better-behaved quantity to work with. In turn, an early motivation for studying B_m is the so-called *discriminator lemma* [17]. The lemma implies that if there is a circuit consisting of a majority of s functions that computes $BMod_m$ then one of those functions p has $B_m(p) \geq 1/s$. Thus, one can use upper bounds on B_m to obtain lower bounds for such circuits.

In this paper we determine up to constant factors the maximum of B_m over quadratic polynomials. This is Item 1 in the next theorem. In fact, we obtain more precise information. Item 2 determines (exactly) the maximum value when n is congruent to $m, 3m \bmod 4m$: either e^2 or $e^2 + e^1$ maximizes B_m, and moreover it will achieve the upper bound on B_m from Item 1. Our inability to determine the maximum value of B_m for every n is reflected in Item 3, which shows when n is congruent to $0, 2m \bmod 4m$ this maximum is not achieved by symmetric polynomials.

▶ **Theorem 6.** *Fix any odd $m \geq 3$, let $\phi := 2\pi/m$, $\ell_1 \in \{\frac{m-1}{4}, \frac{m+1}{4}\}$ denote the integer closest to $\frac{m}{4}$, and set $\Psi := 2m/(m-1)\sqrt{v_{\ell_1\phi}}$. The following holds for large enough n. Let B_m^* denote the maximum $B_m(p)$ over all quadratic p.*

1. *For any n,*

$$\Psi(1/\sqrt{2} - o(1)) \leq B_m^* \leq \Psi(1 + o(1)).$$

2. *If $n \equiv m, 3m \bmod 4m$ then*

$$B_m^* = \max_{s \in \{e^2, e^2+e^1\}} B_m(s) = \Psi(1 - o(1)).$$

3. *If $n \equiv 0, 2m \bmod 4m$ then*

$$(1 + \Omega(1)) \max_{s \in \{0, e^1, e^2, e^2+e^1\}} B_m(s) < \max_{s' \in \{e^2, e^2+e^1\}} B_m(x_1 + s'(x_2, \ldots, x_n)).$$

Note that the polynomial in the right-hand side of Item 3 is not symmetric. We conjecture that this polynomial is in fact optimal (for the corresponding values of n). Our techniques yield slightly stronger results for specific m and n, but for simplicity we only state the above theorem that applies for any odd $m \geq 3$. In particular, when $m = 3$, it is possible to determine for every value of n whether symmetric polynomials maximize B_3.

Previous techniques could at best determine this maximum up to polynomial factors. Hence we also improve polynomially the corresponding circuit lower bounds obtained via the discriminator lemma – this is a straightforward application that we do not state formally.

Green's work [13] also determines exactly the maximum correlation between quadratic polynomials modulo 3 and the parity function. Our setting appears somewhat complicated by the fact that the $BMod_m$ functions are not balanced for odd m.

1.6 Proof sketch of Theorem 5

We begin by rewriting the correlation in a more convenient form, involving derivatives of the polynomial and of the mod function. Bounding the correlation in terms of derivatives is natural and done in several previous works, see e.g. discussion of the "squaring trick" in [28, Chapter 1]. However, these works take repeated derivatives until the polynomial becomes constant, use the Cauchy-Schwartz inequality, and are lossy.

By contrast, we take a single derivative, avoid Cauchy-Schwartz, and give an exact expression. In other words, previous works provide asymptotic correlation bounds for larger degree polynomials, while we provide an *exact bound* for quadratic polynomials.

For concreteness consider the complex mod 3 function $Mod_\phi := e^{\phi\sqrt{-1}\sum_i x_i} := \omega^{\sum_i x_i}$ where $\phi := 2\pi/3$, and fix some quadratic p. Let p_y denote the derivative $p(x+y)+p(x)$ of p in the direction $y \in \{0,1\}^n$. Analogously we let $Mod_{\phi,y}(x) := \omega^{\sum_i x_i - \sum_i (x_i \oplus y_i)}$. We can express the correlation squared as

$$C_\phi^2(p) = \mathbb{E}_y \mathbb{E}_x (-1)^{p_y(x)} Mod_{\phi,y}(x).$$

Writing $c_y(p)$ for the inner expectation – where c stands for *contribution* in direction y – we express the above as $\mathbb{E}_y c_y(p)$. In this language, our goal now is to prove the following for any quadratic p and $s = e^2, e^2 + e^1$:

$$\mathbb{E}_y |c_y(p)| \leq \mathbb{E}_y c_y(s). \tag{1}$$

1.6.1 Computing $\mathbb{E}_y c_y(s)$ and bounding $|c_y(p)|$

We begin by deriving a clean expression for $\mathbb{E}_y c_y(s)$. Let $w(y)$ denote the Hamming weight of y and let E, O denote the set of even, odd weight strings respectively. Supposing n is even for simplicity we have:

$$\mathbb{E}_y c_y(s) = 2^{-n} \sum_{y \in E} (\sin\phi)^{w(y)}. \tag{2}$$

To see this, observe that $s_y = \sum_{i:y_i=1} x_i$ if $y \in E$ and $s_y = \sum_{i:y_i=0} x_i$ if $y \in O$. On the other hand, $Mod_{\phi,y} = \omega^{\sum_{i:y_i=1}(2x_i-1)}$ which only depends on the variables indexed by the 1 bits of y *for every* y.

This means that for any $y \in O$, $c_y(s) = 0$ and for any $y \in E$, $c_y(s) = (\sin\phi)^{w(y)}$. Together this implies (2).

Moreover, by observing that $p_y(x)$ is linear one can show that $(\sin\phi)^{w(y)}$ is in fact an upper bound on $|c_y(p)|$. In other words, for any quadratic p and direction y we have

$$|c_y(p)| \leq (\sin\phi)^{w(y)}. \tag{3}$$

This is an important fact we will use throughout the proof.

1.6.2 Structure on p and slowly restricting y

To deal with $\sum_y |c_y(p)|$, we will first illustrate how we can bound $\sum_{y:y_1=0} |c_y(p)|$. Looking ahead, we are able to deal with any partial sum where at least one bit in y is restricted to 0, as long as p possesses certain structure. This idea, combined with one more ingredient we discuss in the next section, is the heart of the main proof.

For the sake of simplicity, suppose that $p = x_1x_2 + q(x_3, \ldots, x_n)$ for some quadratic q. With this structure on p, it turns out we gain something after conditioning on $y_1 = 0$:

$$\sum_{y:y_1=0} |c_y(p)| \leq \sum_{y:y_1=0,y_2=0} (\sin\phi)^{w(y)}. \tag{4}$$

We gain since this improves on the the bound which follows by only using (3):

$$\sum_{y:y_1=0} |c_y(p)| \leq \sum_{y:y_1=0} (\sin\phi)^{w(y)}. \tag{5}$$

To prove (4) we condition on y_2. If $y_2 = 1$ then we show $c_y(p) = 0$ by mimicking the proof that $c_y(s) = 0$ for any $y \in O$. By assumption on p we have $p_y(x) = x_1 + q_{y'}(x')$ for any $y = 01y'$. And recall $Mod_{\phi,y} = \omega^{\sum_{i:y_i=1}(2x_i-1)}$ does not depend on x_1 since $y_1 = 0$. If $y_2 = 0$ then we use the bound from (3). Combining the two cases implies (4).

In the next step, we would ideally like to bound $\sum_{y:y_1=1} |c_y(p)|$. However, it is not clear how to repeat the previous step, where the assumption on p and restricting $y_1 = 0$ crucially allowed us to observe that $c_y(p) = 0$ for half the directions.

To overcome this, we instead condition on $y_1 = 1, y_2 = 0$. Now $p_y(x) = x_2 + q_{y'}(x')$ for any $y = 10y'$, but $Mod_{\phi,y}$ does not depend on x_2 since $y_2 = 0$. Hence

$$\sum_{y:y_1=1,y_2=0} |c_y(p)| = 0.$$

To summarize, we can make progress on the partial sum $\sum_{y:y_1=1,\ldots,y_{j-1}=1} |c_y(p)|$ by conditioning on $y_j = 0$, as long x_j has certain structure in p. This argument gives a non-trivial bound on $\sum_y |c_y(p)|$, but is still not enough to prove (1). We strengthen it in the next section.

1.6.3 Bounding $t\mathbb{E}_y|c_y(p)|$

We are almost ready to prove our initial goal:

$$\mathbb{E}_y|c_y(p)| \leq \mathbb{E}_y c_y(s).$$

The last ingredient we need is that (3) can be improved to

$$|c_y(p)| \leq (\sin\phi)^{w(y)-1}(\cos\phi) \tag{6}$$

whenever $y \in O$, which we sketch in the next section.

Our proof strategy is similar to that of the previous section. We restrict the direction one bit at a time, but now, we will directly compare $\sum |c_y(p)|$ to $\sum c_y(s)$. In the first step we show that

$$\sum_{y:y_1=0} |c_y(p)| \leq \sum_{y:y_1=0} c_y(s). \tag{7}$$

We bound $\sum_{y:y_1=0} |c_y(p)|$ by applying (6) for the odd weight directions, which allows us to improve the bound on $\sum_{y:y_1=0} |c_y(p)|$ from (4) to the following:

$$\sum_{y:y_1=0} |c_y(p)| \leq \sum_{y:y_1=0,y_2=0,y'\in E} (\sin\phi)^{w(y)} + \sum_{y:y_1=0,y_2=0,y'\in O} (\sin\phi)^{w(y)-1}\cos\phi.$$

To compare this to $\sum_{y:y_1=0} c_y(s)$, we recall the expression from (2) which implies

$$\sum_{y:y_1=0} c_y(s) = \sum_{y:y_1=0,y_2=0,y'\in E} (\sin\phi)^{w(y)} + \sum_{y:y_1=0,y_2=1,y'\in O} (\sin\phi)^{w(y)}.$$

Now we can conclude the proof of (7) as

$$\sum_{y:y_1=0,y_2=0,y'\in O} (\sin\phi)^{w(y)-1}\cos\phi \le \sum_{y:y_1=0,y_2=1,y'\in O} (\sin\phi)^{w(y)}.$$

We remark the improvement from (6) is crucial since if we just used (4) then we would need

$$\sum_{y:y_1=0,y_2=0,y'\in O} (\sin\phi)^{w(y)} \le \sum_{y:y_1=0,y_2=1,y'\in O} (\sin\phi)^{w(y)}$$

which is clearly false as $\sin\phi < 1$.

For the next step, assuming that x_2 appears in at least a few quadratic terms (for the precise conditions see Lemmas 28, 29), we can similarly show that

$$\sum_{y:y_1=1,y_2=0} |c_y(p)| \le \sum_{y:y_1=1,y_2=0} c_y(s).$$

We continue this process until there are no more suitable direction bits to condition on. When this happens, we conclude by reasoning on the remaining structure of the polynomial (see Lemmas 31, 32).

1.6.4 The proof of (6) via handshaking

For any p and y, we can determine $p_y(x)$ by examining the graph $G_{p,y}$, which is defined with $w(y)$ nodes that correspond to the variables indexed by the 1 bits of y, and edges that represent the quadratic terms of p on those $w(y)$ variables. Observe that x_i appears in $p_y(x)$ iff x_i has odd degree in $G_{p,y}$.

Now fix some $y \in O$. The number of nodes in $G_{p,y}$ is odd, and the handshaking lemma implies the number of nodes in $G_{p,y}$ with odd degree must be even. Together this implies $p_y(x)$ contains at most $w(y)-1$ variables which in turn implies (6) after a calculation. For the formal proof see Claim 34.

1.6.5 Slackness

Although we get exact results in the end, we emphasize that some steps in the proof do not yield exact bounds, but are approximate. For example, after we open the first bit we in fact show a strict inequality between $\mathbb{E}_{y:y_1=0}|c_y(p)|$ and $\mathbb{E}_{y:y_1=0}c_y(s)$ when p is non-symmetric (Lemma 30). This gives us a "buffer" between $\mathbb{E}_y|c_y(p)|$ and $\mathbb{E}_y c_y(s)$, which is reflected in the statement of Item 1(d) in Theorem 5.

This extra factor is not just additional information, but is in fact critical for the proof since the final step might be lossy (this occurs when Lemma 31 is applied). The buffer gained will be much larger than the loss from Lemma 31 which allows us to conclude the proof.

2 The CHHLZ conjecture

In this section we present the new technique in [11], their conjecture, and our counterexample. The key ingredient in the approach in [11] is a structural result about the Fourier spectrum of low-degree polynomials over $\mathbb{F}_2$. They show that for any n-variate polynomial p over $\mathbb{F}_2$

of degree $\leq d$, there is a set S of variables such that almost all of the Fourier mass of p lies on Fourier coefficients that intersect with S, and the size of S is exponential in d. This remarkable result allows them to prove new correlation bounds. Further, they conjecture that the size of S needs to be just polynomial in d.

Next we present their conjecture in more detail, and then our results. The main quantity used in [11] is "local correlation" which they define as follows:

▶ **Definition 7** (Local correlation, [11]). *For any $F : \{0,1\}^n \to \{-1,1\}$,*

$$\Delta_S(F) := \mathbb{E}_{x^{\overline{S}}}\left[(\mathbb{E}_{x^S}[F(x)] - \mathbb{E}[F])^2\right].$$

For a polynomial $p : \mathbb{F}_2^n \to \mathbb{F}_2$ we write $e(p)$ for $(-1)^p$ which takes values in $\{-1,1\}$. Next we state their conjecture:

▶ **Conjecture 8** ([11, Conjecture 1.14]). *For every polynomial p of degree d there exists a set S of $\leq poly(d, \log(1/\epsilon))$ variables such that $\Delta_S(e(p)) \leq \epsilon$.*

In fact CHHLZ make a stronger conjecture (Conjecture 1.15 in [11]), where a single set S is found that works for an entire space of dimension k of polynomials. This generality is critical in proving their new correlation bounds. However, we shall give a counterexample even for $k = 1$. In fact, we shall rule out even much weaker parameters and show that what they prove is essentially the best possible. Specifically, we show that for $d = O(\log n)$ and constant ϵ, one needs $|S| \geq n/\log^{O(1)} n$.

▶ **Theorem 9.** *There exists a polynomial p of degree $d = O(\log n)$ such that $\Delta_S(e(p)) \geq \Omega(1)$ for any S of size $\leq c \cdot n/\log^2 n$, where $c > 0$ is an absolute constant.*

The rest of this section is devoted to the proof of this theorem. The idea behind it is quite natural in hindsight, and highlights the expressive power of polynomials of degree $O(\log n)$.

▶ **Definition 10** ([5]; cf. [21], Proposition 4.12). *We define* TRIBES $: \{0,1\}^n \to \{0,1\}$ *to be a read-once monotone DNF where every term has size w so that $|\mathbb{E}_x[\text{TRIBES}(x)] - 1/2| \leq O(\log n)/n$. This makes $w = \log n - \log\log n + O(1)$.*

The next result shows the probability TRIBES is fixed to 1 after a uniform assignment to $x^{\overline{S}}$ is approximately the same as after a uniform assignment to x, where $S \subset [n]$ is a subset of nearly linear size. This property was also used in [18] to show separations between DNFs composed with parity gates and parity decision trees.

▶ **Lemma 11.** *Fix any $S \subset [n]$ such that $|S| \leq O(n/\log^2 n)$. Then*

$$\mathbb{P}_{x^{\overline{S}}}[\text{TRIBES}(x) \text{ not fixed }] \leq 1/2 + o(1).$$

Proof. The set S can intersect at most $|S|$ AND terms. The probability over a uniform assignment to $x^{\overline{S}}$ that TRIBES(x) is fixed to 1 is at least the probability one of the untouched AND terms is set to 1. Hence,

$$\begin{aligned}
\mathbb{P}_{x^{\overline{S}}}[\text{TRIBES}(x) = 1] &\geq 1 - (1 - 2^{-w})^{n/w - |S|}. \\
&= 1 - \frac{\mathbb{P}_x[\text{TRIBES}(x) = 0]}{(1-2^{-w})^{|S|}} \\
&\geq 1 - (1/2 + O(\log n)/n)(1 + 1/\Omega(\log n)) \\
&\geq 1/2 - 1/\Omega(\log n).
\end{aligned}$$

where the second $\geq$ follows since $(1-2^{-w})^{|S|} \geq 1 - |S|/2^w \geq 1 - 1/\Omega(\log n)$ and the fact $1/(1-x) \geq 1 + x$. ◀

We next show that TRIBES can be approximated by a low-degree polynomial. This can be seen as a special case of Razborov's classical approximation [22].

▶ **Lemma 12.** *There exists a $O(\log n)$ degree polynomial p such that*

$$\mathbb{E}_x[e(\text{TRIBES}(x))e(p(x))] \geq 1/2 + \Omega(1).$$

Proof. We will construct a distribution D of $O(\log n)$ degree polynomials such that for any x, $\mathbb{P}_{q\sim D}[q(x) \neq \text{TRIBES}(x)] \leq 1/4$. This would allow us to conclude, since by averaging there must a polynomial $p \in D$ such that $\mathbb{P}_x[p(x) \neq \text{TRIBES}(x)] \leq 1/4$.

To construct D, first note the n/w AND terms can be computed by degree w monomials $m_1(x), \ldots, m_{n/w}(x)$. To sample $q \sim D$, we uniformly sample $T_1, T_2 \subseteq [n/w]$ and set

$$q(x) := 1 - (1 - \bigoplus_{i\in T_1} m_i(x)) \wedge (1 - \bigoplus_{i\in T_2} m_i(x)).$$

Since T_1, T_2 are chosen uniformly, for any x such that $(m_1(x), \ldots m_{n/w}(x)) \neq 0$ we have $\mathbb{P}_{q\sim D}[q(x) = 0] = 1/4$. And for any x such that $(m_1(x), \ldots m_{n/w}(x)) = 0$ we have $\mathbb{P}_{q\sim D}[q(x) = 1] = 0$. Together this implies for any x, $\mathbb{P}_{q\sim D}[q(x) \neq \text{TRIBES}(x)] \leq 1/4$. ◀

We are now ready to prove the main result.

Proof of Theorem 9. First we note that if $\Delta_S(e(p)) \leq \epsilon$ then by Markov's inequality

$$\mathbb{P}_{x^{\overline{S}}}\left[|\mathbb{E}_{x^S}[e(p(x))] - \mathbb{E}[e(p)]| > \epsilon^{1/4}\right] \leq \epsilon^{1/2}. \tag{8}$$

Then, using $T(x)$ to denote $\text{TRIBES}(x)$ for brevity, we can write

$$\begin{aligned}\mathbb{E}_x\left[e(T(x))e(p(x))\right] &= \mathbb{E}_x\left[e(T(x)) \cdot (e(p(x)) - \mathbb{E}[e(p)])\right] + \mathbb{E}[e(T)]\mathbb{E}[e(p)] \\ &\leq \mathbb{E}_x\left[e(T(x)) \cdot (e(p(x)) - \mathbb{E}[e(p)])\right] + O(\log n)/n\end{aligned}$$

where the $\leq$ follows since $|\mathbb{E}[e(T)]| \leq O(\log n)/n$ by the definition of TRIBES.

After a uniform assignment to $x^{\overline{S}}$, let E_1 denote the event $|\mathbb{E}_{x^S}[e(p(x))] - \mathbb{E}[e(p)]| \leq \epsilon^{1/4}$ and let E_2 denote the event that $\text{TRIBES}(x)$ is fixed. Then we have

$$\begin{aligned}&\mathbb{E}_x\left[e(T(x)) \cdot (e(p(x)) - \mathbb{E}[e(p)])\right] \\ &\quad\leq \mathbb{E}_{x^{\overline{S}}}\left[\left|\mathbb{E}_{x^S}\left[e(T(x)) \cdot (e(p(x)) - \mathbb{E}[e(p)])\right]\right|\right] \\ &\quad\leq \mathbb{E}_{x^{\overline{S}}}\left[\left|\mathbb{E}_{x^S}\left[e(T(x)) \cdot (e(p(x)) - \mathbb{E}[e(p)])\right]\right| | E_1E_2\right] + \mathbb{P}[\neg E_1] + \mathbb{P}[\neg E_2] \\ &\quad\leq \epsilon^{1/4} + \epsilon^{1/2} + 1/2 + o(1).\end{aligned}$$

For the last inequality, note that $\mathbb{E}_{x^S}[e(T(x)) \cdot (e(p(x)) - \mathbb{E}[e(p)])|E_1E_2] = \mathbb{E}_{x^S}[e(p(x)) - \mathbb{E}[e(p)]|E_1]$ since $\text{TRIBES}(x)$ is fixed conditioned on E_2. We bound $\mathbb{P}[\neg E_1]$ by (8) and $\mathbb{P}[\neg E_2]$ by Lemma 11. Setting ϵ to a small enough constant contradicts Lemma 12 and concludes the proof of Theorem 9. ◀

3 Derivatives

In this section we rewrite $C_\phi(p)^2$ in terms of the correlation of the derivatives of p with Mod_ϕ, and use this viewpoint to derive several basic facts which will be used later. Fix any $\phi \in [0, 2\pi]$, let $\omega = e^{\phi\sqrt{-1}}$, and from here on we let $\sigma := \sin\phi, \gamma := \cos\phi$.

We begin by using the fact that $|z|^2 = z\overline{z}$ for any complex number, where $\overline{z}$ is the complex conjugate, to rewrite the correlation square $C_\phi^2(p)$ as

$$\mathbb{E}_x(-1)^{p(x)}\omega^{\sum_i x_i} \cdot \overline{\mathbb{E}_y(-1)^{p(y)}\omega^{\sum_i y_i}}.$$

Replacing y with $x \oplus y$ and noting that $\overline{(-1)^{p(y)}\omega^{\sum_i y_i}} = (-1)^{p(y)}\omega^{-\sum_i y_i}$ we can rewrite the correlation square with the following expression:

$$\mathbb{E}_y\mathbb{E}_x(-1)^{p(x)+p(x\oplus y)}\omega^{\sum_i x_i - \sum_i (x_i \oplus y_i)}.$$

The inner expectation over x plays an important role and so we introduce a definition.

▶ **Definition 13.** *The* contribution *of polynomial p in the* direction *y, or the* y-contribution *of p, is $c_y(p) := \mathbb{E}_x(-1)^{p(x)+p(x\oplus y)}\omega^{\sum_i x_i - \sum_i (x_i \oplus y_i)}$.*

Note $c_y(p)$ is always defined with respect to an angle ϕ, which will always be clear from context. Repeating what was said above,

$$C_\phi(p)^2 = \mathbb{E}_y c_y(p).$$

The polynomial $p(x) + p(x \oplus y)$ that appears in $c_y(p)$ is the *derivative* of p in *direction* y, denoted p_y. When p is quadratic, this derivative is linear. Hence, $p_y(x) = \sum_{i \le n} p_{y,i}x_i + p_{y,0}$ where for every y, $p_{y,i} \in \{0,1\}$ are is the coefficient of x_i, and $p_{y,0}$ is the constant.

Because $p_y(x)$ is linear, for fixed y the expectation over x is actually the expectation of *independent* functions of the x_i and so the y-contribution can be written as

$$(-1)^{p_{y,0}} \prod_{i=1}^{n} \mathbb{E}_{x_i}(-1)^{p_{y,i}x_i}\omega^{x_i-(x_i\oplus y_i)}.$$

Each of the expectations $\mathbb{E}_{x_i}(-1)^{p_{y,i}x_i}\omega^{x_i-(x_i\oplus y_i)}$ above takes one of four different values, depending on the four possibilities for $p_{y,i}$ and y_i. These values play a crucial role in this paper and we present them next. Note that if $y_i = 0$ then $x_i - (x_i \oplus y_i) = 0$ and so the ω factor disappears.

▶ **Proposition 14.** *We have the following four possible values for $\mathbb{E}_{x_i}(-1)^{p_{y,i}x_i}\omega^{x_i-(x_i\oplus y_i)}$:*

$p_{y,i}$	y_i	$\mathbb{E}_{x_i}(-1)^{p_{y,i}x_i}\omega^{x_i-(x_i\oplus y_i)}$
0	*0*	$= 1$
1	*0*	$= \mathbb{E}_{x_i}(-1)^{x_i} = 0$
0	*1*	$= \mathbb{E}_{x_i}\omega^{x_i-(x_i\oplus 1)} = \frac{1}{2}\left(\omega^{-1}+\omega\right) = \gamma$
1	*1*	$= \mathbb{E}_{x_i}(-1)^{x_i}\omega^{x_i-(x_i\oplus 1)} = \frac{1}{2}\left(\omega^{-1}-\omega\right) = -\sqrt{-1}\cdot\sigma$

Restricting to $\phi \in [0, \pi/2]$

We now justify our previous assertion that we can restrict our attention to angles $\phi \in [0, \pi/2]$. First, if $\phi \in [\pi/2, 3\pi/2]$ then we can sum e^1 to p. Then $C_\phi(p + e^1) = C_{\pi+\phi}(p)$ and $\pi + \phi \in [-\pi/2, \pi/2]$. Next, if $\phi \in [-\pi/2, 0]$ then $C_\phi(p) = C_{-\phi}(p)$ and now $\phi \in [0, \pi/2]$.

▶ **Definition 15.** *We denote the Hamming weight of $x \in \{0,1\}^n$ by $w(x)$.*

Looking at the table above we can obtain the following bound on $c_y(p)$ in terms of the weight of the derivative.

▷ Claim 16 (Weight bound on contribution). For any $y \in \{0,1\}^n$ and any ϕ we have $|c_y(p)| \leq \max\{\sigma,\gamma\}^{w(y)}$.

We conclude this section by giving a quick illustration of how this framework can be used to compute the maximum correlation for $\phi \in [0,\pi/4]$. Note that Theorem 5 proves a stronger result, showing that non-symmetric polynomials have correlation a constant-factor smaller than optimal. For such ϕ we are going to show that the constant polynomial, which is symmetric, maximizes C_ϕ. By Example 4,

$$C_\phi^2(0) = 2^{-n}\left(1+\gamma\right)^n.$$

We show this is an upper bound for any quadratic polynomial p. We have

$$C_\phi^2(p) \leq \mathbb{E}_y |c_y(p)|,$$

where c_y is as in Definition 13. By Claim 16, since $\gamma > \sigma$, we have

$$|c_y(p)| \leq \gamma^{w(y)}.$$

Hence,

$$C_\phi^2(p) \leq 2^{-n}\sum_{i=0}^{n}\binom{n}{i}\gamma^i = 2^{-n}(1+\gamma)^n,$$

by the binomial theorem.

4 Correlation of symmetric polynomials

We use the information from Section 3 to compute the maximal correlation of symmetric quadratic polynomials, and note an important "no-cancellation" property which will guide the rest of the proof.

We first apply Proposition 14 to determine the *contributions of symmetric polynomials*. The derivatives of e^1 are simply the constant term $e^1_{y,0} = \sum_i y_i$. We now analyze the derivatives of e^2. The coefficient $e^2_{y,i}$ for $i \geq 1$ equals to $\sum_{j\neq i} y_j$ and the constant term $e^2_{y,0}$ equals $\sum_{i<j} y_i y_j$. Combining this information with the above we can characterize the y-contributions of symmetric polynomials.

▶ **Lemma 17** (Contributions of symmetric polynomials). *For any $\phi \in [0,\pi/2]$ and any $y \in \{0,1\}^n$ we have:*

1. *If $w(y)$ is even then $c_y(s) = \sigma^{w(y)}$ for either $s = e^2 + e^1$ or $s = e^2$.*
2. *If $w(y)$ is odd and $w(y) < n$ then $c_y(s) = 0$ for either $s = e^2 + e^1$ or $s = e^2$.*
3. *If $w(y) = n$ and $n \equiv 1 \bmod 4$ then $c_y(s) = +\gamma^n$ for $s = e^2$ and $c_y(s) = -\gamma^n$ for $s = e^1 + e^2$.*
4. *If $w(y) = n$ and $n \equiv 3 \bmod 4$ then $c_y(s) = -\gamma^n$ for $s = e^2$ and $c_y(s) = +\gamma$ for $s = e^1 + e^2$.*

Proof. Refer to Proposition 14.

If $w(y)$ is even, the expectations over x_i with $y_i = 0$ contribute 1 since the corresponding coefficient $s_{y,i}$ (the coefficient of x_i in the derivative polynomial s_y) is 0. This corresponds to the first row of Proposition 14. The other expectations contribute $(-\sqrt{-1})\sigma$. This corresponds to the last row of Proposition 14. In addition, we have the constant term. For

e^2 this term is $(-1)^{\binom{w(y)}{2}} = (-1)^{w(y)^2/2 - w(y)/2} = (-1)^{-w(y)/2}$ using that $w(y)$ is even. For $e^2 + e^1$ the constant term is $(-1)^{\binom{w(y)}{2}+w(y)}$ which again equals $(-1)^{-w(y)/2}$ because $w(y)$ is even. Hence the y-contribution equals

$$(-1)^{-w(y)/2} \cdot ((-\sqrt{-1})\sigma)^{w(y)} = \sigma^{w(y)}$$

where the last equality follows again because $w(y)$ is even.

If $w(y)$ is odd and less than n then some y_i is zero. The corresponding $s_{y,i}$ equals $w(y)$, which is odd. So the contribution is zero, by the second row of Proposition 14.

Finally, consider $w(y) = n$ when n is odd. Note that $s_{y,i} = n - 1$ which is even. By the third row of Proposition 14, the expectation of x is γ^n times the constant term. For $s = e^2$ the constant term is $(-1)^{\binom{n}{2}} = (-1)^{n(n-1)/2}$ which is 1 if $n \equiv 1 \mod 4$ and -1 otherwise. For $s = e^2 + e^1$ the constant term is $(-1)^{\binom{n}{2}+n} = (-1)^{n(n-1)/2+1}$ which is -1 if $n \equiv 1 \mod 4$ and 1 otherwise. ◀

Lemma 17 yields an expression for the maximum $C_\phi(s)$ attained by symmetric quadratic polynomials s. It is best to express this correlation using the quantity v_ϕ that we redefine in a way that is more convenient for the main proof.

▶ **Definition 18** (E, O, v)**.** *Let $E \subseteq \{0,1\}^n$ be the set of n-bit strings of even Hamming weight, and let O be the set of strings of odd weight. Define*

$$v_\phi := 2^{-n} \sum_{y \in E} \sigma^{w(y)}.$$

The equivalence between this definition and the one in the introduction is given by the following claim, which we will use often.

▷ Claim 19 (Odd-even sum). For any number d we have:
$\sum_{y:y\in E} d^{w(y)} = \sum_y d^{w(y)}(1 + (-1)^{w(y)})/2 = \frac{(1+d)^n + (1-d)^n}{2}$,
$\sum_{y:y\in O} d^{w(y)} = \sum_y d^{w(y)}(1 - (-1)^{w(y)})/2 = \frac{(1+d)^n - (1-d)^n}{2}$.

Proof. In each line, the second equality follows from the binomial theorem. ◁

For example, $v_{2\pi/3} = \Theta((1 + \sqrt{3}/2)/2)^n$, where $(1 + \sqrt{3}/2)/2 = 0.933\ldots$. We now give the maximal correlation of a symmetric quadratic polynomial.

▶ **Corollary 20.** *Fix $\phi \in [\pi/2, \pi/4)$ and let C^*_ϕ be the maximum C_ϕ attained by a symmetric quadratic polynomial on n bits for large enough n. We have:*
$C^*_\phi = \sqrt{v_\phi}$ if n is even. This is attained by both e^2 and $e^1 + e^2$.
$C^*_\phi = \sqrt{v_\phi + 1/4^n}$ if n is odd. This is attained by e^2 if $n \equiv 1 \mod 4$ and by $e^1 + e^2$ if $n \equiv 3 \mod 4$.

Proof. By Example 4, $C_\phi(e^1) < C_\phi(0) = \left(\frac{1+\gamma}{2}\right)^{n/2}$. By the definition of v_ϕ, $\sqrt{v_\phi} \geq \Omega\left(\left(\frac{1+\sigma}{2}\right)^{n/2}\right)$ which is greater for n large enough since $\sigma > \gamma$ when $\phi \in [\pi/2, \pi/4)$. The proof now follows from Lemma 17. ◀

(No) cancellations. Note an interesting fact holds for the symmetric polynomial that maximizes C_ϕ: *the y-contributions are always real and non-negative*, for any y. This is not true in general. For a simple example, take $p = e^2, n = 3 \bmod 4$, and $w(y) = n$. Then $c_y(p)$ is negative. This leads to cancellations in the correlation. However, for the symmetric polynomial that maximizes correlation, the inner expectation is always non-negative and there are *no cancellations*.

This fact shows that for the symmetric polynomials p that maximize correlation, the correlation square $C_\phi^2(p)$ can be equivalently written as

$$\mathbb{E}_y|c_y(p)|;$$

that is, we can take absolute values of the contributions "for free". Note that by the triangle inequality, for *any* polynomial p the above expression is an *upper bound* on the correlation. We used this when showing the constant polynomial maximizes C_ϕ for $\phi \in [0, \pi/4]$. For the symmetric polynomials that maximize correlation, it turns out that this bound can be attained.

In the proof of Theorem 5 we shall mostly be working with this quantity, which does not depend on the linear part of p. This is because the derivative of a linear polynomial is a constant depending only on y, which disappears when taking absolute values. Hence we can assume that p does not contain linear terms.

5 Proof of Theorem 5

The next two results are needed to prove the first, main item of Theorem 5. First we deal with polynomials that are missing at least one degree two monomial.

▶ **Theorem 21.** *Let $\phi \in (\pi/4, \pi/2]$ and p be a quadratic polynomial that is not equal to $e^2 + \ell$ for some linear polynomial ℓ. Then $\mathbb{E}_y |c_y(p)| \le (1 - \Omega(\sigma - \gamma))v_\phi$.*

Next we deal with non-symmetric polynomials that possess all degree two monomials. Note we use the quantity $\mathbb{E}_y c_y(p)$ instead.

▶ **Lemma 22.** *Let $\phi \in (\pi/4, \pi/2]$ and p be a polynomial that is equal to $e^2 + \ell$ where ℓ is a linear polynomial not equal to a constant or e^1. Then $\mathbb{E}_y c_y(p) \le (1 - \Omega(1))v_\phi$.*

Assuming these are true, we prove the first item of Theorem 5.

Proof of Theorem 5 Item 1. Follows from Corollary 20, Theorem 21, and Lemma 22. ◀

We next give similar results that are needed to prove the second item of Theorem 5.

▶ **Lemma 23.** *Let $\phi \in [0, \pi/4]$ and p be a quadratic polynomial that is not linear. Then $\mathbb{E}_y |c_y(p)| \le (1 - \Omega(1)) \left(\frac{1+\gamma}{2}\right)^n$.*

▶ **Lemma 24.** *Let $\phi \in [0, \pi/4]$ and p be a linear polynomial that is not equal to the constant polynomial. Then $\mathbb{E}_y c_y(p) \le (1 - \Omega(1)) \left(\frac{1+\gamma}{2}\right)^n$.*

Proof of Theorem 5 Item 2. Follows from Lemma 23, Lemma 24, and Example 4 which says $C_\phi^2(0) = \left(\frac{1+\gamma}{2}\right)^n$. ◀

5.1 Proof of Theorem 21

Our proof strategy is to *slowly restrict the direction* y, to try to connect the corresponding contributions with the target value v_ϕ.

▶ **Definition 25.** *A* restriction *r is an element of $\{0, 1, *\}^n$. The* weight *$w(r)$ of r is the number of ones, and $S(r)$ is the number of stars. We also view r as a function $r : \{0,1\}^{S(r)} \to \{0,1\}^n$ mapping assignments to stars to n-bit strings, and we write ry for $r(y)$. For a restriction r we call x_i a $b \in \{0, 1, *\}$ variable if the ith bit of r is b.*

We emphasize that r restricts the space of directions y, not x. So for example if x_i is a 0 variable then the corresponding directional bit y_i has been restricted to 0 – but x_i is never restricted. We next introduce restricted versions of the quantities in Theorem 21.

▶ **Definition 26** ($c(p,r)$ and $v_\phi(r)$)**.** *Let r be a restriction. For a polynomial p we define*

$$c(p,r) := \mathbb{E}_{y\in\{0,1\}^{S(r)}}|c_{ry}(p)|.$$

Note that $c(p,r)$ is defined with respect to the angle ϕ since $c_{ry}(p)$ is. We also define

$$v_\phi(r) := 2^{-S(r)} \sum_{y\in\{0,1\}^{S(r)}:ry\in E} \sigma^{w(ry)},$$

where we sum over all derivatives ry of even weight.

For any $r \in \{0,1\}^n$ we have $c(p,r) = |c_r(p)|$. Also,

$$\mathbb{E}_y|c_y(p)| = c(p,*^n),$$
$$v_\phi = v_\phi(*^n).$$

Using the above notation our goal is to show that

$$c(p,*^n) \le (1-\Omega(\sigma-\gamma))v_\phi.$$

Polynomials as graphs

We associate to a quadratic polynomial p the *graph* over the variables where x_i and x_j are connected iff monomial x_ix_j is present in p. Note this graph only depends on the monomials of degree 2 of p. The *degree* of a variable shall refer to the degree as a node in this graph. We shall also talk of variables being connected, etc.

▶ **Example 27.** Let $n = 3, r = (1*0) \in \{0,1,*\}^3, p = x_1x_2 + x_2x_3$. The $*$ variable x_2 is connected to the 1 variable x_1 and to the 0 variable x_3.

We now proceed with the proof of Theorem 21. In all upcoming statements, p is an arbitrary quadratic polynomial on n variables, $\phi \in (\pi/4, \pi/2]$, and we set n and a parameter t large enough so that both t and n/t are large enough depending on ϕ. The minimal n for which our proof of Theorem 5 holds increases as ϕ approaches $\pi/4$ (where σ approaches γ).

We next state several lemmas and prove Theorem 21 assuming them. The first two lemmas show that $c(p,r) \le v_\phi(r)$ under various conditions on p and r.

▶ **Lemma 28.** *Let $r \in \{0,1,*\}^n$ be a restriction. Suppose there exists a 0 variable that is connected to an odd number of 1 variables. Then $c(p,r) \le v_\phi(r)$.*

▶ **Lemma 29.** *Let $r \in \{0,1,*\}^n$ be a restriction. Suppose there exists a 0 variable that is connected to an even number of 1 variables and at least t $*$ variables. Then $c(p,r) \le v_\phi(r)$.*

The next lemma shows that if p is missing a degree two monomial then $v_\phi(0*^{n-1})$ gains an advantage over $c(p, 0*^{n-1})$. It can be considered a strengthening of Lemma 29 under an additional constraint.

▶ **Lemma 30** (Buffer)**.** *Let $r = 0*^{n-1}$. Suppose the 0 variable is connected to at least t $*$ variables and at most $n-2$ $*$ variables. Then $c(p,r) \le v_\phi(r) - \left(\frac{\sigma-\gamma}{16}\right)v_\phi$.*

We shall use the above lemmas to slowly restrict directions, beginning with Lemma 30 and then iteratively applying either Lemma 28 or Lemma 29. This process stops when we cannot find variables that satisfy the hypothesis of either Lemma 28 or Lemma 29.

When this happens, we consider two cases based on the number of variables restricted. In the first case, when the number is large, we give an upper bound on $c(p, r)$. This suffices because of the buffer afforded to us by Lemma 30.

▶ **Lemma 31** (Opened majority). *Let $r = 1^j *^{n-j}$ for some $j \geq n/2$. Then $c(p, r) < 2^j \left(\frac{\sigma-\gamma}{1000}\right) v_\phi$.*

In the second case, when the number of restricted variables is small, the polynomial has structure that we can utilize to again show $c(p, r) \leq v_\phi(r)$. Specifically, in the graph of the polynomial many variables have small degree.

▶ **Lemma 32** (Low degree loses). *Let $r = 1^j *^{n-j}$ for some $j < n/2$. Suppose every $*$ variable is connected to at most t other $*$ variables. Then $c(p, r) \leq v_\phi(r)$.*

We will need the following variant of Lemma 32 for an edge case in the main proof.

▶ **Lemma 33.** *Let $r = *^n$. Suppose there are at least $n - t$ variables connected to at most t other variables. Then $c(p, r) \leq (1 - (\sigma - \gamma))v_\phi$.*

Assuming these lemmas we can prove Theorem 21.

Proof of Theorem 21. We consider two cases based on the existence of a variable of certain degree in the graph of p. In the first case, when p is a "typical" polynomial, we suppose the existence of a variable with degree in $[t, n-2]$ (corresponding to the hypothesis of Lemma 30). Let us denote this variable x_1 for ease. We "open" the directional bit corresponding to x_1. That is, we condition $\mathbb{E}_y |c_y(p)|$ depending on the value of y_1:

$$c(p, *^n) = \frac{1}{2}\left(c(p, 0*^{n-1}) + c(p, 1*^{n-1})\right).$$

Correspondingly, it holds that

$$v_\phi(*^n) = \frac{1}{2}\left(v_\phi(0*^{n-1}) + v_\phi(1*^{n-1})\right).$$

Then we iteratively open up $*$ variables in the term where the restriction has no zeroes, as long as we can find a $*$ variable that is connected to an number of 1 variables or that is connected to an even number of 1 variables and at least t other $*$ variables. We can write the terms corresponding to the variables that were opened (up to permutation of variables):

$$c(p, *^n) = \frac{1}{2}c(p, 0*^{n-1}) + \frac{1}{4}c(p, 10*^{n-2}) + \cdots + \frac{1}{2^j}c(p, 1^j *^{n-j}),$$

for some $1 \leq j \leq n$ depending on p. We also write the corresponding terms for v_ϕ:

$$v_\phi(*^n) = \frac{1}{2}v_\phi(0*^{n-1}) + \frac{1}{4}v_\phi(10*^{n-2}) + \cdots + \frac{1}{2^j}v_\phi(1^j *^{n-j}).$$

We compare the terms in the right-hand sides in the two equations above. For the first term, we have $\frac{1}{2}c(p, 0*^{n-1}) \leq \frac{1}{2}v_\phi(0*^{n-1}) - (\frac{\sigma-\gamma}{32})v_\phi$ by Lemma 30. For all the other terms except the last one, we have that the $c(p, r)$ terms is at most the corresponding $v_\phi(r)$ term by either Lemma 28 or Lemma 29. Now we analyze the last terms depending on the value of j. Note that each $*$ variable is connected to at most t other $*$ variables.

If $1 \leq j < n/2$ we apply Lemma 32 which says $c(p, 1^j *^{n-j}) \leq v_\phi(p, 1^j *^{n-j})$ and conclude as $v_\phi(*^n) - c(p, *^n) \geq \frac{\sigma-\gamma}{32} v_\phi$.

If $j \geq n/2$ then $\frac{1}{2^j} c(p, 1^j *^{n-j}) \leq (\frac{\sigma-\gamma}{1000}) v_\phi$ by Lemma 31 and we conclude as $v_\phi(*^n) - c(p, *^n) \geq \left(\frac{\sigma-\gamma}{32} - \frac{\sigma-\gamma}{1000}\right) v_\phi$.

This finishes the proof of when p has a node with degree in $[t, n-2]$. For the second case, suppose that every node has degree at most $t-1$ or degree exactly $n-1$. We then claim there are $\leq t-1$ nodes with degree $n-1$. Supposing this is true we can immediately conclude by Lemma 33.

Now we verify the desired claim. Suppose there are z nodes of degree $n-1$ with $z \geq t$. Each of these nodes is connected to every other node, so every node in the graph has degree at least $z \geq t$. By the supposition, every node in the graph has degree $n-1$. This contradicts the hypothesis that $p \neq e^2 + \ell$. ◀

Next we give proofs of the technical lemmas.

5.1.1 Proof of Lemma 28

Fix a 0 variable x_i that is connected to an odd number of 1 variables. Let T denote the indices of the $*$ variables connected to x_i and let U denote the indices of the remaining $*$ variables. Write $y = (y^T, y^U)$ for the corresponding bits of y.

Note that by Proposition 14, $c_{ry}(p) = 0$ if $w(y^T)$ is even (because the coefficient of x_i would be odd). And if $w(y^T)$ is odd we apply the upper bound $|c_{ry}(p)| \leq \sigma^{w(ry)}$ from Claim 16. Combining these two things yields:

$$\begin{aligned} c(p, r) &= 2^{-S(r)} \sum_{y^T \in O, y^U} |c_{ry}(p)| \\ &\leq 2^{-S(r)} \sum_{y^T \in O, y^U} \sigma^{w(ry)}. \end{aligned}$$

Now we compare this value with the expression for v_ϕ. Let us assume that $w(r)$ is even. Then

$$v_\phi(r) = 2^{-S(r)} \sum_{y \in E} \sigma^{w(ry)}.$$

Hence to prove $c(p, r) \leq v_\phi(r)$ it suffices to show

$$\sum_{y^T \in O, y^U} \sigma^{w(y)} \leq \sum_{y \in E} \sigma^{w(y)}.$$

Note in the above two expressions we can assume $|T| > 0$ since otherwise the left hand-side will be 0 and we would be immediately done. Then by conditioning on the parity of y^U in each side it suffices to show

$$\sum_{y^T \in O, y^U \in E} \sigma^{w(y)} + \sum_{y^T \in O, y^U \in O} \sigma^{w(y)} \leq \sum_{y^T \in E, y^U \in E} \sigma^{w(y)} + \sum_{y^T \in O, y^U \in O} \sigma^{w(y)}.$$

The second sum in each side is the same, and the first sum in the right-hand side is bigger than the first sum in the left-hand side by Claim 19. This concludes the case of when $w(r)$ is even.

When $w(r)$ is odd

$$v_\phi(r) = 2^{-S(r)} \sum_{y \in O} \sigma^{w(ry)}.$$

Then it suffices to show

$$\sum_{y^T \in O, y^U \in E} \sigma^{w(y)} + \sum_{y^T \in O, y^U \in O} \sigma^{w(y)} \leq \sum_{y^T \in E, y^U \in O} \sigma^{w(y)} + \sum_{y^T \in O, y^U \in E} \sigma^{w(y)}.$$

The inequality holds again by Claim 19.

5.1.2 Proof of Lemma 29

The high-level approach is similar to the proof of Lemma 28, but we utilize the following improvement of Claim 16 when the weight of the derivative is odd. The improvement comes from the handshaking lemma.

▷ Claim 34. Let $y \in \{0,1\}^n$. Then $|c_y(p)|$ is either 0 or $\sigma^e \gamma^{w(y)-e}$, where e is an even integer and $0 \leq e \leq w(y)$.

Proof. Consider the graph G with $w(y)$ nodes which are the 1 variables and the edges represent monomials. Let S, T be the nodes in G that have odd, even degree respectively. Note that nodes in S contribute a σ factor, while the nodes in T contribute a γ factor. The remaining $n - w(y)$ 0 variables not in G contribute either 1 or 0.

So to finish the proof it suffices to show that $|S|$ must be even. The sum of all the degrees in G is $|S| \cdot odd + (|V| - |S|) \cdot even = |S| \cdot odd + even$. In any graph, the sum of degrees is even, hence $|S|$ is always even. ◁

To prove Lemma 29 we exploit that if $w(ry)$ is odd then the exponent of the σ factor is $< w(ry)$. Fix the 0 variable x_i that is connected to an even number of 1 variables and to at least t $*$ variables. Let T, U denote the same as in the previous proof. The ry contribution is zero if $w(y^T)$ is odd (because the coefficient of x_i in the ry derivative would be $even + odd = odd$). So then

$$\begin{aligned} c(p,r) &= 2^{-S(r)} \sum_{y^T \in E, y^U} |c_{ry}(p)| \\ &= 2^{-S(r)} \Big(\sum_{y^T \in E, y^U \in E} |c_{ry}(p)| + \sum_{y^T \in E, y^U \in O} |c_{ry}(p)| \Big). \end{aligned}$$

Suppose that $w(r)$ is even. For the first term, where $y^T \in E, y^U \in E$, we use Claim 16. For the second term, where $y^T \in E, y^U \in O$, $w(ry) = even + even + odd = odd$. By Claim 34, the max contribution of ry in the second term is $\leq \sigma^{w(ry)-1}\gamma$. So we can bound

$$c(p,r) \leq 2^{-S(r)} \Big(\sum_{y^T \in E, y^U \in E} \sigma^{w(ry)} + \frac{\gamma}{\sigma} \sum_{y^T \in E, y^U \in O} \sigma^{w(ry)} \Big).$$

We compare this to

$$\begin{aligned} v_\phi(r) &= 2^{-S(r)} \sum_{y \in E} \sigma^{w(ry)} \\ &= 2^{-S(r)} \Big(\sum_{y^T \in E, y^U \in E} \sigma^{w(ry)} + \sum_{y^T \in O, y^U \in O} \sigma^{w(ry)} \Big). \end{aligned}$$

The sums over $y^T \in E, y^U \in E$ are the same. Hence to show $c(p,r) \leq v_\phi(r)$ it suffices to show

$$\begin{aligned}
&\frac{\gamma}{\sigma} \sum_{y^T \in E, y^U \in O} \sigma^{w(y)} \leq \sum_{y^T \in O, y^U \in O} \sigma^{w(y)} \\
\iff &\frac{\gamma}{\sigma} \sum_{y^T \in E} \sigma^{w(y^T)} \leq \sum_{y^T \in O} \sigma^{w(y^T)} \\
\iff &(\sigma/\gamma + 1)(1-\sigma)^{|T|} \leq (\sigma/\gamma - 1)(1+\sigma)^{|T|} \\
\iff &\frac{\sigma+\gamma}{\sigma-\gamma} \leq \left(\frac{1+\sigma}{1-\sigma}\right)^{|T|}.
\end{aligned}$$

The second to last $\iff$ follows by applying Claim 19 and rearranging. The last inequality holds for t large enough, since $|T| \geq t$ and the left hand term will be some fixed positive number since $\phi \in (\pi/4, \pi/2]$. This concludes the $w(r)$ even case.

Now suppose $w(r)$ is odd. Proceeding similarly as before, we have

$$c(p,r) \leq 2^{-S(r)}\left(\frac{\gamma}{\sigma} \sum_{y^T \in E, y^U \in E} \sigma^{w(ry)} + \sum_{y^T \in E, y^U \in O} \sigma^{w(ry)}\right).$$

Which we need to compare with

$$\begin{aligned}
v_\phi(r) &= 2^{-S(r)} \sum_{y \in O} \sigma^{w(ry)} \\
&= 2^{-S(r)}\left(\sum_{y^T \in E, y^U \in O} \sigma^{w(ry)} + \sum_{y^T \in O, y^U \in E} \sigma^{w(ry)}\right).
\end{aligned}$$

Now the sums over $y^T \in E, y^U \in O$ are the same. So then it suffices to show

$$\frac{\gamma}{\sigma} \sum_{y^T \in E} \sigma^{w(y^T)} \leq \sum_{y^T \in O} \sigma^{w(y^T)}$$

which we have already verified.

5.1.3 Proof of Lemma 30

The proof starts identically as the proof of Lemma 29, but then we strengthen the analysis to give a strict inequality. Let T denote the set of $*$ variables connected to x_1, and let U denote the $*$ variables not connected to x_1. We have $|T| + |U| = n - 1$ and by hypothesis $t \leq |T| \leq n - 2$. We remark the strengthened analysis only works because of the condition $|T| \leq n - 2$.

We have the following derivation, where the first inequality follows from the same steps as in $w(r)$ even case of the previous proof. Let $a = 1 + \sigma, b = 1 - \sigma$, and $\delta = \gamma/\sigma$.

$$\begin{aligned}
2^{n-1}\left(v_\phi(0*^{n-1}) - c(p, 0*^{n-1})\right) &\geq \sum_{y^T \in O, y^U \in O} \sigma^{w(y)} - \frac{\gamma}{\sigma} \sum_{y^T \in E, y^U \in O} \sigma^{w(y)}. \\
&= \sum_{y^U \in O} \sigma^{w(y^U)} \cdot \left(\sum_{y^T \in O} \sigma^{w(y^T)} - \delta \sum_{y^T \in E} \sigma^{w(y^T)}\right) \\
&= \frac{a^{|U|} - b^{|U|}}{2} \cdot \frac{(1-\delta)a^{|T|} - (1+\delta)b^{|T|}}{2} \\
&\geq \frac{a^{|U|}}{4} \cdot \frac{(1-\delta)a^{|T|}}{4} \\
&= \frac{(1-\delta)a^{n-1}}{16}.
\end{aligned}$$

We elaborate on the last $\geq$. First, note that if $|U| = 0$ the inequality would not be valid since the entire expression would be equal to 0. Second, we verify that

$$\begin{aligned} &\frac{(1+\delta)b^{|T|}}{2} \leq \frac{(1-\delta)a^{|T|}}{4} \\ \iff &2 \cdot \frac{\sigma+\gamma}{\sigma-\gamma} \leq \left(\frac{1+\sigma}{1-\sigma}\right)^{|T|}. \end{aligned}$$

The last inequality holds for t large enough, since $|T| \geq t$. Note this is almost the same inequality that is in the proof of Lemma 29. Lastly, we verify that

$$\begin{aligned} &\frac{b^{|U|}}{2} \leq \frac{a^{|U|}}{4} \\ \Leftarrow\ &2 \leq \frac{1+\sigma}{1-\sigma}. \end{aligned}$$

The $\Leftarrow$ holds since $|U| > 0$ and the last inequality is equivalent to $\sigma \geq 1/3$ which holds since $\sigma = \sin(\phi) \geq \sin(\pi/4) = 1/\sqrt{2} \geq 1/3$.

We continue the derivation, applying similar logic:

$$\begin{aligned} \frac{(1-\delta)a^{n-1}}{16} &\geq \frac{(1-\delta)a^{n-1} + (1-\delta)b^{n-1}}{32} \\ &\geq \frac{(1-\delta)a^{n} + (1-\delta)b^{n}}{32a} \\ &= \frac{(1-\delta)}{16a} \cdot 2^n v_\phi. \end{aligned}$$

Dividing both sides by 2^{n-1} we obtain

$$\begin{aligned} v_\phi(0*^{n-1}) - c(p, 0*^{n-1}) &\geq \frac{(1-\delta)}{8a} \cdot v_\phi \\ &\geq \frac{\sigma-\gamma}{16} \cdot v_\phi. \end{aligned}$$

where the last $\geq$ follows since $a = 1+\sigma \leq 2$, $(1-\delta) = \frac{\sigma-\gamma}{\sigma} \geq \sigma - \gamma$ because $\sigma \leq 1$.

5.1.4 Proof of Lemma 31

Applying Claim 16 we can say

$$\begin{aligned} c(p, 1^j *^{n-j}) &\leq 2^{-(n-j)}\sigma^j \sum_y \sigma^{w(y)} \\ &= 2^{-(n-j)}\sigma^j (1+\sigma)^{n-j}. \end{aligned}$$

On the other hand,

$$2^j v_\phi(*^n) \geq 2^{-(n-j+1)}(1+\sigma)^n.$$

So it suffices to show that

$$\begin{aligned} &\frac{\sigma^j(1+\sigma)^{n-j}}{2^{n-j}} \leq \frac{\sigma-\gamma}{1000}\frac{(1+\sigma)^n}{2^{n-j+1}} \\ \iff &\frac{2000}{\sigma-\gamma} \leq \left(\frac{1+\sigma}{\sigma}\right)^j, \end{aligned}$$

where we divided by $\sigma - \gamma > 0$. The last inequality holds for n large enough since $j \geq n/2$ and $\sigma > 0$.

5.1.5 Proof of Lemma 32

Consider the subgraph induced by the $*$ variables. There are $n-j \geq n/2$ nodes in it of degree $\leq t$. By a greedy argument, this implies an independent set of size $\geq (n-j)/(t+1) \geq n/4t$. Let T denote the variables in the independent set and let S denote the remaining $*$ variables. Note $|S| + |T| = n - j$ and the remaining j variables are 1 variables.

For any fixing y^S of S, let $p^T(y^S) \in \{0,1\}^{|T|}$ denote the coefficients of the variables in T based on the partial restriction $1^j y^S *^{|T|}$. This is a valid definition because T is an independent set, and so $p^T(y^S)$ is unaffected by any fixing y^T of T. By Proposition 14, if for some fixing y^T there is a variable x_j in T such that $p^T_j(y^S) = 1$ but $y^T_j = 0$ then the contribution is 0. Using also the other values in the table in Proposition 14, for any fixed y^S we can let $\psi := w(p^T(y^S))$ and bound the contribution over y^T as follows:

$$\begin{aligned} 2^{|T|} c(p, 1^j y^S *^{|T|}) &\leq \sigma^{j+w(y^S)+\psi} \sum_{z \in \{0,1\}^{|T|-\psi}} \gamma^{w(z)} \\ &= \sigma^{j+w(y^S)+\psi}(1+\gamma)^{|T|-\psi} \\ &\leq \sigma^{j+w(y^S)}(1+\gamma)^{|T|}. \end{aligned}$$

The last $\leq$ follows since $\sigma < 1 \leq 1+\gamma$. By summing over all possible fixings y^S and applying the previous bound we can bound $c(p, 1^j *^{n-j})$ as follows:

$$\begin{aligned} 2^{n-j} c(p, 1^j *^{n-j}) &\leq \sigma^j (1+\gamma)^{|T|} \sum_{y^S} \sigma^{w(y^S)} \\ &= \sigma^j (1+\gamma)^{|T|} (1+\sigma)^{|S|} \\ &\leq \sigma^j (1+\gamma)^{n/4t} (1+\sigma)^{(n-j)-n/4t}. \end{aligned}$$

The last $\leq$ holds since $\sigma > \gamma$ and $|T| \geq n/4t$. On the other hand,

$$\begin{aligned} 2^{n-j} v_\phi(1^j *^{n-j}) &= \sum_{y : 1^j y \in E} \sigma^{j+w(y)} \\ &\geq \sigma^j \frac{(1+\sigma)^{n-j}}{4}. \end{aligned}$$

So then it suffices to show

$$\begin{aligned} \sigma^j (1+\gamma)^{n/4t} (1+\sigma)^{(n-j)-n/4t} &< \sigma^j \frac{(1+\sigma)^{n-j}}{4} \\ \iff (1+\gamma)^{n/4t} &< \frac{(1+\sigma)^{n/4t}}{4} \\ \iff 4 &< \left(\frac{1+\sigma}{1+\gamma}\right)^{n/4t}. \end{aligned}$$

Since $\sigma > \gamma$ when $\phi \in (\pi/4, \pi/2]$, the last inequality holds for n/t large enough.

5.1.6 Proof of Lemma 33

The proof is nearly identical to the proof of Lemma 32. The hypothesis implies the existence of an independent set of size $\geq (n-t)/(t+1) \geq (n-t)/2t$ in the graph consisting of all the variables. Following the same logic as before, we can upper bound $c(p, *^n)$ by

$$2^n c(p, *^n) \leq (1+\gamma)^{(n-t)/2t} (1+\sigma)^{n-(n-t)/2t}.$$

On the other hand,

$$2^n v_\phi \geq \frac{(1+\sigma)^n}{2}.$$

Then it suffices to show

$$\begin{aligned}(1+\gamma)^{(n-t)/2t}(1+\sigma)^{n-(n-t)/2t} &< (1-(\sigma-\gamma))\frac{(1+\sigma)^n}{2} \\ \iff \frac{2}{(1-(\sigma-\gamma))} &< \left(\frac{1+\sigma}{1+\gamma}\right)^{(n-t)/2t}.\end{aligned}$$

Recall that n/t is arbitrarily large, so $(n-t)/2t$ is also arbitrarily large and the inequality holds.

5.2 Proof of Lemma 22

We can perform a similar analysis as in the proof of Lemma 17. As before $c_y(p) = 0$ if $w(y)$ is odd. But now if $w(y)$ even, letting T denote the set of variables that appear in the linear polynomial ℓ, the contribution is

$$\begin{aligned}c_y(p) &= (-1)^{-w(y)/2+w(y^T)} \cdot ((-\sqrt{-1})\sigma)^{w(y)} \\ &= (-1)^{w(y^T)}\sigma^{w(y)}.\end{aligned}$$

So a derivative makes a positive contribution if $w(y)$ is even and $w(y^T)$ is even, and a negative one if $w(y)$ is even and $w(y^T)$ is odd. Let U be the complement of T. By hypothesis, $1 \leq |T|, |U| \leq n-1$. We can sum over the positive contributions and subtract the negative ones to get the expression

$$2^n \cdot \mathbb{E}_y c_y(p) = \sum_{y^T \in E, y^U \in E} \sigma^{w(y)} - \sum_{y^T \in O, y^U \in O} \sigma^{w(y)}.$$

On the other hand,

$$2^n \cdot v_\phi = \sum_{y^T \in E, y^U \in E} \sigma^{w(y)} + \sum_{y^T \in O, y^U \in O} \sigma^{w(y)}.$$

Combining the two expressions and letting $a = (1+\sigma), b = (1-\sigma)$, we get

$$\begin{aligned}2^n\left(v_\phi - \mathbb{E}_y c_y(p)\right) &= 2 \sum_{y^T \in O, y^U \in O} \sigma^{w(y)} \\ &= \frac{1}{2}\left(a^{|T|} - b^{|T|}\right)\left(a^{|U|} - b^{|U|}\right) \\ &\geq \frac{1}{2}\frac{a^{|T|}}{2}\frac{a^{|U|}}{2} \\ &= \frac{a^n}{8}.\end{aligned}$$

The second $=$ follows by Claim 19, and the $\geq$ after that follows since $1 \leq |T|, |U|$ by hypothesis and $2b < a$.

5.3 Proof of Lemma 23

Since p is not linear there is at least one node with degree ≥ 1 in the polynomial graph. Let us denote this node x_1 for ease, and let T, U denote the nodes connected, not connected to x_1 respectively. We write $y = (y^T, y^U)$ for the corresponding bits of y. Just like in the proof of Theorem 21 we condition on the value of y_1 to get

$$c(p, *^n) = \frac{1}{2}\left(c(p, 0*^{n-1}) + c(p, 1*^{n-1})\right).$$

We bound the second term by applying Claim 16 which says $c_{ry}(p) \leq \gamma^{w(ry)}$ using that $\phi \in [0, \pi/4]$:

$$\begin{aligned} 2^{n-1} c(p, 1*^{n-1}) &\leq \sum_{y \in \{0,1\}^{n-1}} \gamma^{1+w(y)} \\ &= \gamma\,(1+\gamma)^{n-1}. \end{aligned}$$

To deal with the first term, we proceed similarly as we did in the proof of Lemma 28. Note that $|T| \geq 1$, and if $w(y^T)$ is odd then $c_{1y}(p) = 0$. If $w(y^T)$ is even then as before we use the bound $c_{ry}(p) \leq \gamma^{w(ry)}$. These two things yield

$$\begin{aligned} 2^{n-1} c(p, 0*^{n-1}) &\leq \sum_{y^T \in E, y^U} \gamma^{w(y)} \\ &= \left(\frac{(1+\gamma)^{|T|} + (1-\gamma)^{|T|}}{2}\right)(1+\gamma)^{|U|} \\ &\leq 3/4(1+\gamma)^{n-1}. \end{aligned}$$

The last $\leq$ follows as $|T| \geq 1$ and $1 - \gamma < \frac{1+\gamma}{2}$ when $1/\sqrt{2} \leq \gamma$. Altogether this gives

$$2^n c(p, *^n) \leq (3/4 + \gamma)(1+\gamma)^{n-1}.$$

So it only remains to show $(3/4 + \gamma) \leq (1 - \Omega(1))(1+\gamma)$ which holds because $\gamma \leq 1$.

5.4 Proof of Lemma 24

Let T denote the set of variables that appear in the linear polynomial p and let U denote the remaining variables. Applying the same logic as in Example 4 we have

$$\begin{aligned} \mathbb{E}_y c_y(p) &= \left(\frac{1-\gamma}{2}\right)^{|T|}\left(\frac{1+\gamma}{2}\right)^{|U|} \\ &\leq \left(\frac{1-\gamma}{2}\right)\left(\frac{1+\gamma}{2}\right)^{n-1}. \end{aligned}$$

The $\leq$ follows since $|T| \geq 1$ and $1 + \gamma > 1 - \gamma$ when $\phi \in [0, \pi/4]$.

So it only remains to show $(1 - \gamma) \leq (1 - \Omega(1))(1+\gamma)$ which holds because $\gamma \geq 1/\sqrt{2}$.

6 Boolean correlation

In this section we prove Theorem 6. Recall that C_ϕ is defined as the absolute value of a sum. We need to analyze this sum more carefully, so we define it next.

▶ **Definition 35.** *$E_\phi(p) := \mathbb{E}_{x \in \{0,1\}^n}(-1)^{p(x)}\omega^{\sum_i x_i}$. Note that $|E_\phi(p)| = C_\phi(p)$.*

We now give an overview of the upcoming technical results. In the proof of Theorem 6, we will use Lemma 39, which relates $B_m(p)$ to the quantity $|Real(E_\phi(p))|$ for a specific angle ϕ, and Corollary 41, which allows us to compute $|Real(E_\phi(s))|$ for $s = e^2, e^2 + e^1$. Together these two results will enable us to compute $B_m(s)$ for $s = e^2, e^2 + e^1$.

On the other hand, combining Lemma 39 with Theorem 5 lets us bound $B_m(p)$ when p is not symmetric, since Theorem 5 bounds $C_\phi(p)$ and $|Real(E_\phi(p))| \le |E_\phi(p)| = C_\phi(p)$.

Proposition 36 and Claims 37, 38 are used to prove Lemma 39, and Lemma 40 is needed for Corollary 41.

For the rest of the section, fix any odd $m \ge 3$, set $\phi = 2\pi/m$, $\omega = e^{\phi\sqrt{-1}}$. We start with the following standard fact:

▶ **Proposition 36.** *Let b be the fraction of n-bit strings whose weight is divisible by m. For any p,*

$$B_m(p) = \frac{1}{b(1-b)} \left| \frac{2}{m} \cdot \sum_{k=1}^{(m-1)/2} Real(E_{k\phi}(p)) + \frac{1}{m} - b \right|$$

where $Real(z)$ denotes the real part of the complex number z.

Proof. Let $s(k) := \sum_{j=0}^{m} \omega^{jk} = 1 + \omega^k + \cdots + \omega^{(m-1)k}$ and note that $s(k) = m$ if $k \equiv 0 \mod m$ and $s(k) = 0$ otherwise. Using this notation we can write

$$B_m(p) = \left| \mathbb{E}_x (-1)^{p(x)} \frac{s(w(x))}{m} \cdot \frac{1}{b} - \mathbb{E}_x (-1)^{p(x)} \left(1 - \frac{s(w(x))}{m}\right) \cdot \frac{1}{1-b} \right|.$$

Collecting terms this is

$$\left| \mathbb{E}_x (-1)^{p(x)} \left(\frac{s(w(x))}{m} \cdot \frac{1}{b} - \left(1 - \frac{s(w(x))}{m}\right) \cdot \frac{1}{1-b} \right) \right|.$$

Using the definition of s this equals

$$\left| \mathbb{E}_x (-1)^{p(x)} \left[\left(\sum_{j=1}^{m} \omega^{jw(x)} \right) \left(\frac{1}{mb} + \frac{1}{m(1-b)} \right) + \frac{1}{mb} - \left(1 - \frac{1}{m}\right) \frac{1}{1-b} \right] \right|.$$

Also,

$$\frac{1}{mb} - \left(1 - \frac{1}{m}\right) \frac{1}{1-b} = \frac{1-mb}{mb(1-b)}.$$

Furthermore, $\omega^{jw(x)} + \omega^{(m-j)w(x)} = 2Real(\omega^{jw(x)})$ for each j. After factoring out $1/b(1-b)$ the result follows. ◀

Observe that in the statement of Lemma 36, if we replaced b with $1/m$ then the terms that don't multiply ω would be 0. However, $b \ne 1/m$ but it will be very close. We use the following bound [1] that's implicit in [8].

▷ Claim 37. $|b - 1/m| < \cos(\pi/m)^n$.

Now let $\ell_1 \in \{\frac{m-1}{4}, \frac{m+1}{4}\}$ denote the integer closest to $\frac{m}{4}$. The next result suggests we should focus on $Real(E_{\ell_1\phi}(p))$.

[1] When $m = 3$ the claim says $|b - 1/m| < 2^{-n}$ but we do not use this.

▷ Claim 38. Fix any odd $m \geq 3$ and $k \in \{1..., (m-1)/2\} : k \neq \ell_1$. Then for all large enough n and any quadratic p,

$$|Real(E_{k\phi}(p))| = o(\sqrt{v_{\ell_1\phi}}).$$

Proof. By Theorem 5, for any k it holds that

$$C_{k\phi}(p) \leq \max_{s\in\{0,e^1,e^2,e^2+e^1\}} C_{k\phi}(s) \leq \max\left\{O\left(\left(\frac{1+|\sin(k\phi)|}{2}\right)^{n/2}\right), \left(\frac{1+|\cos(k\phi)|}{2}\right)^{n/2}\right\}.$$

Next we claim that if $k \in \{1..., (m-1)/2\}, k \neq \ell_1$ then $\max\{|\sin(k\phi)|, |cos(k\phi)|\} < \sin(\ell_1\phi)$. If this holds we can conclude since $\sqrt{v_{\ell_1\phi}} = \Omega((\frac{1+\sin(\ell_1\phi)}{2})^{n/2})$ and $|Real(E_{k\phi}(p))| \leq C_{k\phi}(p)$.

To verify the claim, note for $k \neq \ell_1$, $|\sin(k\phi)|$ is maximized when $k = \ell_2$, where ℓ_2 denotes the second closest integer to $m/4$. Since m is odd, $\ell_2 \in \{\frac{m-3}{4}, \frac{m+3}{4}\}$ which implies $\sin(\ell_2\phi) < \sin(\ell_1\phi)$.

And $|\cos(k\phi)|$ is maximized for $k = (m-1)/2$ and $|\cos(k\phi)| = |\cos(\pi - \pi/m)| = \cos(\pi/m)$. We can now conclude as $\cos(\pi/m) < \sin(\ell_1\phi) = \sin(\pi/2 \pm \pi/2m) = \cos(\pi/2m)$. ◁

The next result, which combines Claim 37, 38 with Proposition 36, says we can approximate $B_m(p)$ using just $|Real(E_{\ell_1\phi}(p))|$.

▶ **Lemma 39.** *For all large enough n and any quadratic p,*

$$\left|B_m(p) - \frac{2m}{m-1}|Real(E_{\ell_1\phi}(p))|\right| \leq o(\sqrt{v_{\ell_1\phi}}).$$

For $m = 3$ this can be improved to

$$\left|B_3(p) - 3\left|Real(E_{2\pi/3}(p))\right|\right| \leq O(2^{-n}).$$

Proof. By Claim 37 and noting that $\cos(\pi/m)^n = o(\sqrt{v_{\ell_1\phi}})$ we have

$$\left|\frac{1}{b(1-b)} - \frac{m^2}{m-1}\right| = o(\sqrt{v_{\ell_1\phi}}).$$

Applying the triangle inequality and Claim 38 we also have

$$\left|\left|\sum_{k=1}^{(m-1)/2} Real(E_{k\phi}(p))\right| - \left|Real(E_{\ell_1\phi}(p))\right|\right| \leq \sum_{k\neq\ell_1}\left|Real(E_{k\phi}(p))\right| \leq m \cdot o(\sqrt{v_{\ell_1\phi}}).$$

Inserting the previous two inequalities into Lemma 36 implies

$$|B_m(p) - 2m/(m-1)|Real(E_{\ell_1\phi}(p))|| \leq O(m)o(\sqrt{v_{\ell_1\phi}}).$$

We can now conclude since we consider m fixed. ◀

We are naturally interested in computing $B_m(s)$ for $s = e^2, e^2 + e^1$ and the next lemma allows us to do so by giving an expression for $E_{\ell_1\phi}(s)$. In Section 4 we determined $C_{\ell_1\phi}(s) = |E_{\ell_1\phi}(s)|$, but this no longer suffices as we need to understand the angle of $E_{\ell_1\phi}(s)$ in order to compute $|Real(E_{\ell_1\phi}(s))|$.

▶ **Lemma 40.** *For any $k \in \{1, 2, \ldots, m-1\}$ we have:*

$$E_{k\phi}(e^2) = 2^{-(n+1)}\left[(1+i)(1-i\omega^k)^n + (1-i)(1+i\omega^k)^n\right],$$
$$E_{k\phi}(e^2+e^1) = 2^{-(n+1)}\left[(1-i)(1-i\omega^k)^n + (1+i)(1+i\omega^k)^n\right].$$

Proof. We prove Item 1. Since $(-1)^{e^2(x)} = (-1)^{\binom{w(x)}{2}}$ we can write

$$E_{k\phi}(e^2) = \sum_{j=0}^{n} \binom{n}{j} (-1)^{\binom{j}{2}} \omega^{kj}.$$

We also have

$$\sum_{j=0 \bmod 4} \binom{n}{j} \omega^{kj} = \sum_{j=0}^{n} \binom{n}{j} \omega^{kj} \left(\frac{1+i^j}{2}\right)\left(\frac{1+(-1)^j}{2}\right)$$
$$\sum_{j=2 \bmod 4} \binom{n}{j} \omega^{kj} = \sum_{j=0}^{n} \binom{n}{j} \omega^{kj} \left(\frac{1-i^j}{2}\right)\left(\frac{1+(-1)^j}{2}\right).$$

So this implies

$$\sum_{j=0 \bmod 4} \binom{n}{j} \omega^{kj} - \sum_{j=2 \bmod 4} \binom{n}{j} \omega^{kj} = \frac{1}{2}\left[(1+\omega^k i)^n + (1+\omega^k(-i))^n\right].$$

Doing the analogous for $j = 1, 3 \mod 4$ gives

$$\sum_{j=1 \bmod 4} \binom{n}{j} \omega^{kj} - \sum_{j=3 \bmod 4} \binom{n}{j} \omega^{kj} = \frac{1}{2}\left[-i(1+\omega^k i)^n + i(1+\omega^k(-i))^n\right].$$

The proof of Item 2 is similar. ◀

The next result reduces the problem of computing $|Real(E_{\ell_1\phi}(s)|$ to the problem of computing $|\cos(\chi \pm \pi/4)|$ for a certain angle χ. The angle $\chi \pm \pi/4$ arises because it is the angle of the vector $(1 \pm i)(1 - i\omega^{\ell_1})^n$, which is the dominant term in the previous expressions for $E_{\ell_1\phi}(s)$. The last equality below then allows us to relate $|Real(E_{\ell_1\phi}(s)|$ to $\sqrt{v_{\ell_1\phi}}$.

▶ **Corollary 41.** *Let $\chi = \frac{n\pi}{4m}, -\frac{n\pi}{4m}$ when $\ell_1 = \frac{m+1}{4}, \frac{m-1}{4}$ respectively. Let $\gamma = \sqrt{2}|1 - i\omega^{\ell_1}|^n$. For all large enough n, the following holds:*

1. $\left|2^{n+1}|Real(E_{\ell_1\phi}(e^2))| - |\cos(\chi + \pi/4)|\gamma\right| = o(1)$
2. $\left|2^{n+1}|Real(E_{\ell_1\phi}(e^2 + e^1))| - |\cos(\chi - \pi/4)|\gamma\right| = o(1)$,
3. $\left|2^{n+1}\sqrt{v_{\ell_1\phi}} - \gamma\right| = o(1)$.

Proof. We show the first equality when $\ell_1 = \frac{m+1}{4}$. The $\ell_1 = \frac{m-1}{4}$ case is symmetrical.

By definition $\omega^{\ell_1} = e^{\sqrt{-1}(2\pi/m)(m+1)/4} = e^{\sqrt{-1}(\pi/2+\pi/2m)}$, hence $-i\omega^{\ell_1} = e^{\sqrt{-1}(\pi/2m)}$. This implies $(1 - i\omega^{\ell_1}) = |1 - i\omega^{\ell_1}|e^{\sqrt{-1}(\pi/4m)}$. Additionally, $1 + i = \sqrt{2}e^{\sqrt{-1}(\pi/4)}$. So then

$$\begin{aligned}(1+i)(1-i\omega^{\ell_1})^n &= \sqrt{2}e^{\sqrt{-1}(\pi/4)} \cdot |1 - i\omega^{\ell_1}|^n e^{\sqrt{-1}(\pi/4m)n} \\ &= \gamma e^{\sqrt{-1}(n\pi/4m+\pi/4)}.\end{aligned}$$

We can now conclude by Lemma 40, the fact $|Real(e^{\sqrt{-1}\phi})| = |\cos\phi|$ for any ϕ, and noting $|1 + i\omega^{\ell_1}|^n = o(1)$ since $|1 + i\omega^{\ell_1}| < 1$ when m is odd. The second inequality is done similarly.

The third inequality follows by Lemma 40, the facts $|E_{\ell_1\phi}(p)| = C_{\ell_1\phi}(p)$, $|1+i\omega^{\ell_1}|^n = o(1)$, and since when $s = e^2, e^2 + e^1$, $\left|C_{\ell_1\phi}(s) - \sqrt{v_{\ell_1\phi}}\right| \le o(1)$ by Lemma 17. ◀

6.1 Proof of Theorem 6

6.1.1 Proof of Item 1

First we prove the upper bound. Lemma 39 implies that

$$B_m(p) \le 2m/(m-1)\,|Real(E_{\ell_1\phi}(p))| + o(\sqrt{v_{\ell_1\phi}}).$$

The upper bound now follows since $|Real(E_{\ell_1\phi}(p))| \le |E_{\ell_1\phi}(p)| = C_{\ell_1\phi}(p) \le (1+o(1))\sqrt{v_{\ell_1\phi}}$. The last inequality holds by Theorem 5.

Next we prove the lower bound by showing

$$\max_{s\in\{e^2,e^2+e^1\}} B_m(s) \ge (2m/(m-1) - o(1))\sqrt{\frac{v_{\ell_1\phi}}{2}}. \tag{9}$$

Lemma 39 implies that

$$B_m(s) \ge 2m/(m-1)\,|Real(E_{\ell_1\phi}(s))| - o(\sqrt{v_{\ell_1\phi}}).$$

Then we claim that for either $s = e^2$ or $s = e^2 + e^1$,

$$|Real(E_{\ell_1\phi}(s))| \ge (1-o(1))\sqrt{\frac{v_{\ell_1\phi}}{2}}.$$

The previous two inequalities imply Equation 9.

To verify the claim, note that since $\cos(\pi/4) = 1/\sqrt{2}$, at least one of the next two inequalities hold for any angle χ:

$$\cos(\chi + \pi/4) \ge 1/\sqrt{2},$$
$$\cos(\chi - \pi/4) \ge 1/\sqrt{2}.$$

We then conclude by Corollary 41.

6.1.2 Proof of Item 2

We present the $n \equiv 3m \bmod 4m, \ell_1 = \frac{m+1}{4}$ case. In the proof we show that $E_{\ell_1}(e^2)$ is essentially real, which means $|Real(E_{\ell_1}(e^2))|$ equals $\sqrt{v_{\ell_1\phi}}$ by Corollary 41. On the other hand, for any non-symmetric p, $C_{\ell_1\phi}(p)$ is a constant factor smaller than $\sqrt{v_{\ell_1\phi}}$ by Theorem 5. This suffices as $|E_{\ell_1\phi}(p)| = C_{\ell_1\phi}(p)$, and note the angle of $E_{\ell_1\phi}(p)$ does not even matter.

So first we show

$$B_m(e^2) \ge (2m/(m-1) - o(1))\sqrt{v_{\ell_1\phi}}.$$

This follows by Lemma 39 and the claim that

$$\left|Real(E_{\ell_1\phi}(e^2))\right| \ge (1-o(1))\sqrt{v_{\ell_1\phi}}.$$

To verify the claim, note when $n \equiv 3m \bmod 4m$, $n\pi/4m = (3m + k4m)\pi/4m \equiv 3\pi/4 + k\pi \bmod 2\pi$ for some integer k. Hence $\cos(n\pi/4m + \pi/4) = \cos((k+1)\pi) = \pm 1$. We then conclude by Corollary 41. Note $\cos(n\pi/4m - \pi/4) = 0$, so $B_m(e^2 + e^1) < B_m(e^2)$.

On the other hand, for any $p \ne e^2, e^2 + e^1$ we show

$$B_m(p) \le 2m/(m-1)\sqrt{1 - \Omega(\sin(\ell_1\phi) - \cos(\ell_1\phi))}\cdot\sqrt{v_{\ell_1\phi}}.$$

This follows by Lemma 39 and Theorem 5 which states

$$C_{\ell_1\phi}(p) \leq \sqrt{1 - \Omega(\sin(\ell_1\phi) - \cos(\ell_1\phi))} \cdot \sqrt{v_{\ell_1\phi}}.$$

This yields the desired inequality since $|Real(E_{\ell_1\phi}(p))| \leq C_{\ell_1\phi}(p)$.

If $p = e^1, 0$ we show

$$\max_{s\in\{0,e^1\}} B_m(s) \leq (2m/(m-1)) \cdot o(\sqrt{v_{\ell_1}}). \tag{10}$$

This follows by Lemma 39 and noting for $s = e^1, 0$, $C_{\ell_1\phi}(s) = (\frac{1+\cos(\ell_1\phi)}{2})^{n/2} = o(\sqrt{v_{\ell_1\phi}})$ since $\cos(\ell_1\phi) < \sin(\ell_1\phi)$.

The $n \equiv 3m, \ell_1 = \frac{m-1}{4}$ case is similar except we use $e^2 + e^1$ instead of e^2. The $n \equiv m$ cases are analogous.

6.1.3 Proof of Item 3

We present the $n \equiv 0 \bmod 4m, \ell_1 = \frac{m+1}{4}$ case. First note that Equations 9 and 10 imply it suffices to prove $\max_{s\in\{e^2,e^2+e^1\}} B_m(s) < B_m(q)$ for some non-symmetric q. We will show that $E_{\ell_1\phi}(e^2), E_{\ell_1\phi}(e^2 + e^1)$ are both maximally imaginary as allowed by Equation 9. Next, consider $q := x_1 + e^2(x_2, \ldots, x_n)$. $C_{\ell_1\phi}(q)$ is close to, but less than $C_{\ell_1\phi}(s)$ for $s = e^2, e^2 + e^1$. However, $E_{\ell_1\phi}(q)$ will be more real which is enough to compensate for this difference and show that $|Real(E_{\ell_1\phi}(s))| < |Real(E_{\ell_1\phi}(q))|$.

So first we show that for either $s = e^2, e^2 + e^1$,

$$B_m(s) \leq (2m/(m-1) + o(1)) \cdot \sqrt{\frac{v_{\ell_1\phi}}{2}}.$$

This follows by Lemma 39 and the claim that for either $s = e^2, e^2 + e^1$,

$$|Real(E_{\ell_1\phi}(s))| \leq (1 + o(1))\sqrt{\frac{v_{\ell_1\phi}}{2}}.$$

To verify the claim, since $n \equiv 0 \bmod 4m$, then $n\pi/4m \equiv k\pi \bmod 2\pi$. Hence $\cos(n\pi/4m \pm \pi/4m) = \pm 1/\sqrt{2}$. We then conclude by Corollary 41.

On the other hand, we show that

$$B_m(q) > (2m/(m-1) - o(1)) \cdot \frac{(1 + \tan(\pi/4m))\sqrt{v_{\ell_1\phi}}}{\sqrt{2}}.$$

Note $1 + \tan(\pi/4m) > 1$ for $m \geq 3$. The inequality holds by Lemma 39 and the claim

$$|Real(E_{\ell_1\phi}(q))| \geq (1 - o(1)) \cdot \frac{(1 + \tan(\pi/4m))\sqrt{v_{\ell_1\phi}}}{\sqrt{2}}.$$

To show the claim, we start by rewriting $E_{\ell_1\phi}(q)$ by conditioning on x_1 (below e^2 is on $n-1$ variables):

$$E_{\ell_1\phi}(q) = \frac{(1 - \omega^{\ell_1})}{2} E_{\ell_1\phi}(e^2).$$

An analogous version of Corollary 41 Item 1 holds for e^2 on $n-1$ variables:

$$\left|2^n |Real(E_{\ell_1\phi}(e^2))| - \left|\cos\left(\frac{(n-1)\pi}{4m} + \frac{\pi}{4}\right)\right| \frac{\gamma}{|1 - i\omega^{\ell_1}|}\right| = o(1).$$

Since $-\omega^{\ell_1} = e^{\sqrt{-1}(-\pi/2+\pi/2m)}$ we have $(1-\omega^{\ell_1}) = |1-\omega^{\ell_1}|e^{\sqrt{-1}(-\pi/4+\pi/4m)}$. Combining this with the previous equality implies that

$$\left|2^{n+1}|Real(E_{\ell_1\phi}(q))| - \left|\cos\left(\frac{(n-1)\pi}{4m}+\frac{\pi}{4m}\right)\right|\frac{|1-\omega^{\ell_1}|}{|1-i\omega^{\ell_1}|}\gamma\right| = o(1)$$

$$\iff \left|2^{n+1}|Real(E_{\ell_1\phi}(q))| - \frac{|1-\omega^{\ell_1}|}{|1-i\omega^{\ell_1}|}\gamma\right| = o(1).$$

The $\iff$ follows as $\cos(n\pi/4m) = \pm 1$ when $n \equiv 0 \bmod 4m$.

To conclude, by Corollary 41 it suffices to show

$$\frac{1+\tan(\pi/4m)}{\sqrt{2}} = \frac{|1-\omega^{\ell_1}|}{|1-i\omega^{\ell_1}|}.$$

Using the identity $|1+e^{\sqrt{-1}\phi}| = 2|cos(\phi/2)|$, we have $|1-i\omega^{\ell_1}| = 2\cos(\pi/4m)$ and $|1-\omega^{\ell_1}| = 2|\cos(-\pi/4+\pi/4m)| = 2\cos(\pi/4-\pi/4m) = \sqrt{2}(\cos(\pi/4m)+\sin(\pi/4m))$ where the last step holds as $cos(a-b) = \cos a\cos b + \sin a \sin b$. Hence the equality holds.

The $n \equiv 0$, $\ell_1 = \frac{m-1}{4}$ case is similar except q will be $e^2(x_2,\ldots,x_n)$ instead. The $n \equiv 2m$ cases are analogous.

7 Symmetric correlates poorly with mod m

For completeness, we show that symmetric polynomials mod 2 correlate poorly with the complex mod m function. To get a sense of the parameters below, fix $m = 3$ and apply the identities $\cos x \le 1 - x^2/6$ and $(1-x)^n \le e^{-xn}$. This yields $C_\phi(s) \le O(d)2^{-\Omega(n/d^2)}$, so if Conjecture 2 were true this would imply exponentially small correlation bounds for any $O(\log n)$ degree polynomial - a long-standing open problem.

▶ **Theorem 42.** *Let $\phi = 2\pi k/m$ for some odd m and $k \in \{1,\ldots m-1\}$. Then for any degree d symmetric polynomial s,*

$$C_\phi(s) \le 2md \cdot \cos\left(\frac{\pi}{2md}\right)^n.$$

Proof. Let δ be an integer such that $2^{\delta-1} \le d < 2^\delta$. It is shown in [6] that $s(x)$ is determined by the weight of x mod 2^δ. Hence we can write

$$(-1)^{s(x)} = \sum_{i=0}^{2^\delta-1} c_i \mathbf{1}_{w(x)\equiv i \bmod 2^\delta}$$

where $c_i \in \{-1,1\}$ for each i. Then we can write the correlation as

$$C_\phi(s) = \left|\mathbb{E}_x[Mod_\phi(x)\cdot\sum_{i=0}^{2^\delta-1} c_i\mathbf{1}_{w(x)\equiv i \bmod 2^\delta}]\right|$$

$$= \left|\sum_{i=0}^{2^\delta-1}\mathbb{E}_x\left[Mod_\phi(x)\cdot c_i\mathbf{1}_{w(x)\equiv i \bmod 2^\delta}\right]\right|.$$

Letting $\omega = e^{\sqrt{-1}\cdot 2\pi/m}$, for any i we have

$$\mathbb{E}_x\left[Mod_\phi(x)\cdot\mathbf{1}_{w(x)\equiv i \bmod 2^\delta}\right] = \sum_{j=0}^{m-1}\omega^{(i+j2^\delta)k}\mathbb{P}_x[w(x)\equiv i+j2^\delta \bmod m2^\delta].$$

We next use a slightly generalized version of Claim 37: ◀

▷ Claim 43. For any k, m, $|\mathbb{P}_x[w(x) \equiv k \bmod m] - 1/m| \leq \cos(\pi/m)^n$.

Proof. Combining this with the fact $\sum_{j=0}^{m-1} \omega^{(i+j2^\delta)k} = 0$ implies that

$$\left|\mathbb{E}_x\left[Mod_\phi(x) \cdot \mathbf{1}_{w(x)\equiv i \bmod 2^\delta}\right]\right| \leq m(\cos(\pi/m2^\delta))^n.$$

Hence

$$C_\phi(s) \leq \sum_{i=0}^{2^\delta-1} \left|\mathbb{E}_x\left[Mod_\phi(x) \cdot c_i \mathbf{1}_{w(x)\equiv i \bmod 2^\delta}\right]\right| \leq m2^\delta \cos(\pi/m2^\delta))^n.$$

We can now conclude the proof since $2^\delta \leq 2d$. ◁

8 Structured cubic loses to quadratic

In this section we show that any cubic polynomial with a symmetric degree 3 part has correlation that is a constant factor worse than the optimal achieved by quadratic polynomials.

▶ **Theorem 44.** *Suppose $t = e^3 + q$ for some arbitrary quadratic q. Then for any ϕ,*

$$C_\phi(t) \leq (1 - \Omega(1)) \max_{s\in\{0,e^1,e^2,e^2+e^1\}} C_\phi(s).$$

We first show that cubic symmetric polynomial e^3 has worse correlation than the optimal quadratic symmetric. We prove this by applying the derivative framework from Section 3. We analyze for every direction y what the derivative e^3_y will be and use this to bound the contribution $|c_y(e^3)|$ in Lemma 45.

Next we show that $t = e^3 + q$ can only have worse correlation than e^3 for any quadratic q. We do this in Lemma 46 by showing that for any direction y, adding the derivative q_y (which will be linear) to e^3_y can only decrease the contribution. In other words, we show $|c_y(t)| \leq |c_y(e^3)|$ for every y.

▶ **Lemma 45.** *For any y,*

1. *If $w(y) \in E$ then*

$$|c_y(e^3)| \leq \frac{|\sigma|^{w(y)} + |\gamma|^{w(y)}}{2}.$$

2. *If $w(y) \in O$ then*

$$|c_y(e^3)| \leq \frac{2^{w(y)}}{2^{n-1}}.$$

▶ **Lemma 46.** *Suppose $t = e^3 + q$ for some arbitrary quadratic q. Then for any y,*

$$|c_y(t)| \leq |c_y(e^3)|.$$

The previous two lemmas imply Theorem 44.

Proof of Theorem 44 assuming Lemmas 45, 46. By Lemmas 45, 46 we have

$$\begin{aligned} C^2_\phi(t) &\leq \sum_{y:w(y)\in E} \frac{|\sigma|^{w(y)} + |\gamma|^{w(y)}}{2} + \sum_{y:w(y)\in O} 2^{-(n-w(y)-1)} \\ &= \frac{(1+|\sigma|)^n + (1-|\sigma|)^n}{4} + \frac{(1+|\gamma|)^n + (1-|\gamma|)^n}{4} + \frac{3^n-1}{2^n}. \end{aligned}$$

The = follows by Claim 19. Next note that for any ϕ, $\max\{1+|\sigma|, 1+|\gamma|\} \geq 1+1/\sqrt{2} > 3/2$. Suppose ϕ is such that $|\sigma| > |\gamma|$. Then

$$C^2_\phi(t) \leq 2^{-n}\frac{(1+o(1))(1+|\sigma|)^n}{4}.$$

On the other hand by Theorem 5 we know that

$$\max_{s\in\{e^2, e^2+e^1\}} C^2_\phi(s) \geq 2^{-n}\frac{(1+|\sigma|)^n}{2}.$$

Now suppose ϕ is such that $|\sigma| \leq |\gamma|$. Then

$$C^2_\phi(t) \leq 2^{-n}\frac{(2+o(1))(1+|\gamma|)^n}{4}.$$

However by Theorem 5,

$$\max_{s\in\{0,e^1\}} C^2_\phi(s) = 2^{-n}(1+|\gamma|)^n.$$ ◀

8.1 Proof of Lemma 45

We first list some preliminary results we will need. The following is a standard fact we state without proof.

▷ Claim 47. Let s denote either e^2, e^2+e^1 on n variables, and let ℓ denote an arbitrary linear polynomial. Then $|bias((-1)^{s+\ell})| \leq 2^{-(n-1)/2}$.

Below and for the remainder of the section, we let $V_1, V_0 \subseteq [n]$ denote the indices of the 1, 0-variables respectively with respect to a fixed direction y.

The next result says that if the bias of p_y is small after an arbitrary restriction to the 1-variables, then $|c_y(p)|$ must be small.

▶ **Proposition 48.** *Fix some polynomial p and direction $y \in \{0,1\}^n$. Suppose for any restriction $r \in \{0,1\}^{|V_1|}$ of the 1-variables,*

$$\left|\mathbb{E}_{x:x^{V_1}=r}(-1)^{p_y(x)}\right| \leq \delta.$$

Then

$$|c_y(p)| \leq \delta.$$

Proof. We have

$$\begin{aligned} c_y(p) &= \mathbb{E}_x[(-1)^{p_y(x)}Mod_{\phi,y}(x)] \\ &= \mathbb{E}_{x^{V_1}}[Mod_{\phi,y}(x)\cdot\mathbb{E}_{x^{V_0}}[(-1)^{p_y(x)}]] \\ &\leq \delta. \end{aligned}$$

The second = follows since $Mod_{\phi,y}(x)$ only depends on the 1-variables. The $\leq$ follows since $|Mod_{\phi,y}(x)| = 1$ and by the hypothesis on p_y. ◀

Next we characterize the derivatives of e^3 which depend on the weight of y mod4. We abuse notation and let $e^i(V_j)$ denote the polynomial e^i defined on the variables indexed by V_j.

▶ **Proposition 49.** *Fix any direction $y \in \{0,1\}^n$ and consider the derivative e^3_y.*

1. *If $w(y) \equiv 0 \bmod 4$ then*

$$e^3_y = e^1(V_1) + e^1(V_1)e^1(V_0).$$

2. *If $w(y) \equiv 2 \bmod 4$ then*

$$e^3_y = e^1(V_1)e^1(V_0) + e^1(V_0).$$

3. *If $w(y) \equiv 1 \bmod 4$ then*

$$e^3_y = e^2(V_1) + e^2(V_0).$$

4. *If $w(y) \equiv 3 \bmod 4$ then*

$$e^3_y = (e^2 + e^1)(V_1) + (e^2 + e^1)(V_0) + 1.$$

Proof. We can write $e^3 = e^3(V_1) + e^2(V_1)e^1(V_0) + e^1(V_1)e^2(V_0) + e^3(V_0)$. Firstly note the term $e^3(V_0)$ does not affect e^3_y. Secondly, the term $e^1(V_1)e^2(V_0)$ only contributes $e^2(V_0)$ to e^3_y when $|V_1| = w(y)$ is odd.

Thirdly, we deal with $e^2(V_1)e^1(V_0)$. Note that $e^1(V_0)$ has a coefficient of $\binom{w(y)}{2}$ in e^3_y, which is odd when $w(y) \equiv 2, 3 \bmod 4$. Now let x_i denote a 1-variable. Then $x_i e^1(V_0)$ has a coefficient of $\binom{w(y)-1}{1}$, hence $e^1(V_1)e^1(V_0)$ appears when $w(y)$ is even.

Lastly, we deal with $e^3(V_1)$. Note x_i has a coefficient of $\binom{w(y)-1}{2}$, hence $e^1(V_1)$ appears if $w(y) \equiv 0, 3 \bmod 4$. Let x_j denote a second 1-variable. Then $x_i x_j$ has a coefficient of $\binom{w(y)-2}{1}$ hence $e^2(V_1)$ appears if $w(y)$ is odd. The constant 1 has a coefficient of $\binom{w(y)}{3}$ which is odd when $w(y) \equiv 3 \bmod 4$. ◀

Proof of Lemma 45

Proof. Suppose $w(y) \equiv 0 \bmod 4$. By Proposition 49, if $x^{V_0} \in E$ then $e^3_y = e^1(V_1)$. If $x^{V_0} \in O$ then $e^3_y = 0$. Hence

$$\begin{aligned} c_y(e^3) &= 2^{-n}\Big(\sum_{x : x^{V_0} \in E} \sigma^{w(y)} + \sum_{x : x^{V_0} \in O} \gamma^{w(y)}\Big) \\ &= \frac{\sigma^{w(y)} + \gamma^{w(y)}}{2}. \end{aligned}$$

The $w(y) \equiv 2 \bmod 4$ case is similar. If $x^{V_0} \in E$ then $e^3_y = 0$. Otherwise, $e^3_y = e^1(V_0) + 1$. Hence $c_y(e^3) = \frac{-\sigma^{w(y)} + \gamma^{w(y)}}{2}$. This concludes the $w(y) \in E$ case.

Now suppose $w(y) \in O$. Fact 47 implies that for $s = e^2(V_0), (e^2 + e^1)(V_0)$, $|bias((-1)^s)| \leq 2^{-(n-w(y)-1)}$. Since e^3_y is disjoint on V_0, V_1, Proposition 48 implies that $|c_y(e^3)| \leq 2^{-(n-w(y)-1)}$. ◀

8.2 Proof of Lemma 46

Suppose that $t = e^3 + q$ for some quadratic q. Note that for any direction y, t_y has the same quadratic terms as e^3_y and q_y only affects the linear terms in p_y. Let us write $q_y = u(V_1) + v(V_0)$, where $u(V_1), v(V_0)$ are linear polynomials over the $1, 0$-variables respectively.

First suppose $y \equiv 0 \bmod 4$. We now consider restricting the 1-variables. If $x^{V_1} \in E$ then $t^3_y = c + v(V_0)$ where c is some constant. If $x^{V_1} \in O$ then $t^3_y = c + (e^1 + v)(V_0)$. Note that if $0 \neq v(V_0) \neq e^1(V_0)$, then the bias of the restricted function will be 0 for both cases. Hence by Proposition 48, $c_y(t) = 0$ and we are done. If $v(V_0) = e^1(V_0)$ then this is symmetrical to when $v(V_0) = 0$. Hence we can assume that $v(V_0) = 0$.

From here, we switch back to restricting the 0-variables. If $x^{V_0} \in E$ then $e_y^3 = (e^1+u)(V_1)$, and if $x^{V_0} \in O$ then $e_y^3 = u(V_1)$. Suppose $u(V_1)$ contains k variables. Then $|c_y(t)| \leq |\sigma|^{w(y)-k}|\gamma|^k$ whenever $x^{V_0} \in E$ and $|c_y(t)| \leq |\sigma|^k|\gamma|^{w(y)-k}$ otherwise. Hence

$$|c_y(t)| \leq \frac{|\sigma|^{w(y)-k}|\gamma^k| + |\sigma|^k|\gamma|^{w(y)-k}}{2}.$$

Assume that $|\sigma| > |\gamma|$ (the other case is similar). We can now conclude as

$$\begin{aligned} &\frac{|\sigma|^{w(y)-k}|\gamma|^k + |\sigma|^k|\gamma|^{w(y)-k}}{2} \leq \frac{|\sigma|^{w(y)} + |\gamma|^{w(y)}}{2} \\ \iff &\frac{|\gamma|^{w(y)-k}(|\sigma|^k - |\gamma|^k)}{2} \leq \frac{|\sigma|^{w(y)-k}(|\sigma|^k - |\gamma|^k)}{2} \\ \iff &|\gamma| \leq |\sigma|. \end{aligned}$$

The $w(y) \equiv 2 \bmod 4$ case is analogous.

Now suppose $w(y) \equiv 1 \bmod 4$. After an arbitrary restriction to x^{V_1}, we have $e_y^3 = e^2(V_0)+v(V_0)+c$ for some constant c. Fact 47 implies that $|bias((-1)^{e_y^3})| \leq 2^{-(n-w(y)-1)}$ after any restriction to x^{V_1}. We can now conclude by applying Proposition 48. The $w(y) \equiv 3 \bmod 4$ case is analogous.

References

1. Scott Aaronson and Avi Wigderson. Algebrization: a new barrier in complexity theory. In *40th ACM Symp. on the Theory of Computing (STOC)*, pages 731–740, 2008.
2. Noga Alon and Richard Beigel. Lower bounds for approximations by low degree polynomials over Z_m. In *IEEE Conf. on Computational Complexity (CCC)*, pages 184–187, 2001.
3. László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *J. of Computer and System Sciences*, 45(2):204–232, 1992.
4. Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P=?NP question. *SIAM J. on Computing*, 4(4):431–442, 1975.
5. Michael Ben-Or and Nathan Linial. Collective coin flipping, robust voting schemes and minima of Banzhaf values. In *26th Symposium on Foundations of Computer Science*, pages 408–416, Portland, Oregon, 21–23 October 1985. IEEE.
6. Nayantara Bhatnagar, Parikshit Gopalan, and Richard J. Lipton. Symmetric polynomials over Z_m and simultaneous communication protocols. *J. of Computer and System Sciences*, 72(2):252–285, 2006.
7. Abhishek Bhowmick and Shachar Lovett. Nonclassical polynomials as a barrier to polynomial lower bounds. In *IEEE Conf. on Computational Complexity (CCC)*, pages 72–87, 2015. doi:10.4230/LIPIcs.CCC.2015.72.
8. Ravi Boppana, Johan Håstad, Chin Ho Lee, and Emanuele Viola. Bounded independence versus symmetric tests. *ACM Trans. Computation Theory*, 11(4):21:1–21:27, 2019.
9. Jean Bourgain. Estimation of certain exponential sums arising in complexity theory. *Comptes Rendus Mathématique. Académie des Sciences. Paris*, 340(9):627–631, 2005.
10. Jin-Yi Cai, Frederic Green, and Thomas Thierauf. On the correlation of symmetric functions. *Mathematical Systems Theory*, 29(3):245–258, 1996.
11. Eshan Chattopadhyay, Pooya Hatami, Kaave Hosseini, Shachar Lovett, and David Zuckerman. XOR lemmas for resilient functions against polynomials. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *ACM Symp. on the Theory of Computing (STOC)*, pages 234–246. ACM, 2020. doi:10.1145/3357713.3384242.

12 Eduardo Dueñez, Steven J. Miller, Amitabha Roy, and Howard Straubing. Incomplete quadratic exponential sums in several variables. *Journal of Number Theory*, 116(1):168–199, 2006.

13 Frederic Green. The correlation between parity and quadratic polynomials mod 3. *J. of Computer and System Sciences*, 69(1):28–44, 2004.

14 Frederic Green, Daniel Kreymer, and Emanuele Viola. Block-symmetric polynomials correlate with parity better than symmetric. *Computational Complexity*, 26(2):323–364, 2017. Available at `https://www.ccs.neu.edu/home/viola/papers/blocksym.pdf`.

15 Frederic Green and Amitabha Roy. Uniqueness of optimal mod 3 circuits for parity. *Journal of Number Theory*, 130:961–975, 2010.

16 Frederic Green, Amitabha Roy, and Howard Straubing. Bounds on an exponential sum arising in Boolean circuit complexity. *Comptes Rendus Mathématique. Académie des Sciences. Paris*, 341(5):279–282, 2005.

17 András Hajnal, Wolfgang Maass, Pavel Pudlák, Mário Szegedy, and György Turán. Threshold circuits of bounded depth. *J. of Computer and System Sciences*, 46(2):129–154, 1993.

18 Xuangui Huang, Peter Ivanov, and Emanuele Viola. Affine extractors and ac0-parity, 2021.

19 Eric Miles and Emanuele Viola. Substitution-permutation networks, pseudorandom functions, and natural proofs. *J. of the ACM*, 62(6), 2015.

20 Moni Naor, Omer Reingold, and Alon Rosen. Pseudorandom functions and factoring. *SIAM J. Comput.*, 31(5):1383–1404, 2002.

21 Ryan O'Donnell. Analysis of boolean functions, 2007. Lecture notes. Available at `http://www.cs.cmu.edu/~odonnell/boolean-analysis/`.

22 Alexander Razborov. Lower bounds on the dimension of schemes of bounded depth in a complete basis containing the logical addition function. *Akademiya Nauk SSSR. Matematicheskie Zametki*, 41(4):598–607, 1987. English translation in Mathematical Notes of the Academy of Sci. of the USSR, 41(4):333-338, 1987.

23 Alexander Razborov and Steven Rudich. Natural proofs. *J. of Computer and System Sciences*, 55(1):24–35, August 1997.

24 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *19th ACM Symp. on the Theory of Computing (STOC)*, pages 77–82. ACM, 1987.

25 Roman Smolensky. On representations by low-degree polynomials. In *34th IEEE IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 130–138, 1993.

26 Terence Tao and Tamar Ziegler. The inverse conjecture for the gowers norm over finite fields in low characteristic. In *Annals of Combinatorics*, 2012.

27 Emanuele Viola. New correlation bounds for GF(2) polynomials using Gowers uniformity. *Electronic Colloquium on Computational Complexity*, Technical Report TR06-097, 2006. URL: `https://eccc.weizmann.ac.il/report/2006/097/`.

28 Emanuele Viola. Correlation bounds for polynomials over $\{0,1\}$. *SIGACT News, Complexity Theory Column*, 40(1), 2009.

29 Emanuele Viola. On the power of small-depth computation. *Foundations and Trends in Theoretical Computer Science*, 5(1):1–72, 2009.

30 Emanuele Viola. Challenges in computational lower bounds. *SIGACT News, Open Problems Column*, 48(1), 2017.

31 Emanuele Viola. Fourier conjectures, correlation bounds, and majority. In *Coll. on Automata, Languages and Programming (ICALP)*, 2021. Available at `https://www.ccs.neu.edu/home/viola/papers/L12requiresCor.pdf`.

32 Emanuele Viola. New lower bounds for probabilistic degree and AC0 with parity gates. *Theory of Computing*, 2021. Available at `https://www.ccs.neu.edu/home/viola/papers/diago.pdf`.

33 Emanuele Viola. Correlation bounds against polynomials, a survey, 2022.

On the Algebraic Proof Complexity of Tensor Isomorphism

Nicola Galesi
Dipartimento Ingegneria Informatica Automatica e Gestionale "A. Ruberti",
Sapienza University of Rome, Italy

Joshua A. Grochow
Departments of Computer Science and Mathematics, University of Colorado Boulder, CO, USA

Toniann Pitassi
Department of Computer Science, Columbia University, New York, NY, USA

Adrian She
Department of Mathematics and Computer Science, University of Toronto, Canada

Abstract

The Tensor Isomorphism problem (TI) has recently emerged as having connections to multiple areas of research within complexity and beyond, but the current best upper bound is essentially the brute force algorithm. Being an algebraic problem, TI (or rather, proving that two tensors are *non*-isomorphic) lends itself very naturally to algebraic and semi-algebraic proof systems, such as the Polynomial Calculus (PC) and Sum of Squares (SoS). For its combinatorial cousin Graph Isomorphism, essentially optimal lower bounds are known for approaches based on PC and SoS (Berkholz & Grohe, SODA '17). Our main results are an $\Omega(n)$ lower bound on PC degree or SoS degree for Tensor Isomorphism, and a nontrivial upper bound for testing isomorphism of tensors of bounded rank.

We also show that PC cannot perform basic linear algebra in sub-linear degree, such as comparing the rank of two matrices (which is essentially the same as 2-TI), or deriving $BA = I$ from $AB = I$. As linear algebra is a key tool for understanding tensors, we introduce a strictly stronger proof system, PC+Inv, which allows as derivation rules all substitution instances of the implication $AB = I \to BA = I$. We conjecture that even PC+Inv cannot solve TI in polynomial time either, but leave open getting lower bounds on PC+Inv for any system of equations, let alone those for TI. We also highlight many other open questions about proof complexity approaches to TI.

2012 ACM Subject Classification Theory of computation → Proof complexity; Theory of computation → Problems, reductions and completeness

Keywords and phrases Algebraic proof complexity, Tensor Isomorphism, Graph Isomorphism, Polynomial Calculus, Sum-of-Squares, reductions, lower bounds, proof complexity of linear algebra

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.4

Funding *Joshua A. Grochow*: Supported by NSF CAREER award CCF-2047756.
Toniann Pitassi: Supported by NSF grant CCF-1900460, and by the IAS School of Mathematics.
Adrian She: Supported by NSERC Canada Graduate Scholarship.

Acknowledgements NG and JAG would like to thank Michael Forbes for early conversation about the PC degree of matrix rank, which occurred at Dagstuhl Seminar 18051: Proof Complexity in early 2018. We would also like to thank the organizers A. Atserias, J. Nordstrom, P. Pudlák, and R. Santhanam for their invitation and support.

1 Introduction

Tensors have rapidly emerged as a fundamental data structure and key mathematical object of the 21st century. They play key roles in many different areas of science, engineering, and mathematics, from quantum mechanics and general relativity to neural networks [39] and mechanical engineering. They arise in theoretical computer science in many ways, including from (post-quantum) cryptography [41, 30], derandomization, Matrix Multiplication, Graph Isomorphism [26], and several different parts of Geometric Complexity Theory.

The fundamental notion of equivalence between tensors is that of isomorphism: two tensors are isomorphic if one can be transformed into the other by an invertible linear change of basis in each of the corresponding vector spaces. For example, two 2-tensors (=matrices) M, M' are equivalent under this notion if there are invertible matrices X, Y such that $XMY = M'$; similarly, two 3-tensors, represented by 3-way arrays T_{ijk}, T'_{ijk} are isomorphic if there are three invertible matrices X, Y, Z such that

$$\sum_{ijk} X_{ii'} Y_{jj'} Z_{kk'} T_{ijk} = T'_{i'j'k'} \tag{1}$$

for all i', j', k'. The problem of (3-)Tensor Isomorphism (TI) is: given two such 3-way arrays, to decide if they are isomorphic.

Over finite fields, two different versions of TI sandwich the complexity of its more famous cousin, Graph Isomorphism. Namely, as presented above, GI reduces to TI. In the other direction, over a finite field $=_{p^a}$, one can take an $n \times n \times n$ tensor and list it out "verbosely", as a set of p^{an} many $n \times n$ matrices over ; the isomorphism problem for such verbosely given tensors is equivalent to Group Isomorphism for a certain class of p-groups, widely believed to be the hardest cases of Group Isomorphism in general. As such, this verbose version of TI reduces to GI. Furthermore, with Babai's quasi-polynomial-time algorithm [4], the running times are quite close: $N^{O(\log N)}$ for VerboseTI and $N^{O(\log^2 N)}$ for GI (the exponent of the exponent was worked out by Helfgott [28]). Thus TI stands as a key obstacle to putting GI into P.

In this paper, we initiate the study of (algebraic) proof complexity approaches to proving that two tensors are non-isomorphic. Lower bounds on the Polynomial Calculus proof system imply lower bounds on Gröbner basis techniques, and the latter are some of the leading methods for solving TI-complete problems in cryptanalysis, e.g., [49, 18]. In the context of GI, proof complexity plays an important role, through its connection with the Weisfeiler–Leman (WL) algorithm. Although this algorithm does not, on its own, solve GI in polynomial time [15], it is a key subroutine in many of the best algorithms for GI, both in theory [4] and in practice (see [36, 37]). And the picture that has emerged is that some proof systems for GI are known to be equivalent in power to WL [3], and some lower bounds on proof systems are closely related to lower bounds for WL [45, 40]. Versions of WL for groups, and in particular finite p-groups – and hence, by the connection above, tensors over finite fields – have only recently begun to be explored [12, 9, 10, 17].

1.1 Main results

We focus on the Polynomial Calculus (PC, or Gröbner) proof system [16], though our results will also hold for semi-algebraic proof systems such as Sum-of-Squares [32] as well. PC is used to show that a system of polynomial equations over a field is unsatisfiable over the algebraic closure , by deriving from the system of equations, in a line-by-line fashion, the contradiction $1 = 0$. The degree of a PC proof is the maximum degree of any line appearing

in the proof, and it is a fundamental result that PC proofs of constant degree can be found in polynomial time [16]. Much as WL informally "captures all combinatorial approaches" to GI, PC informally "captures all approaches based on Gröbner bases" to showing that a system of polynomial equations is unsatisfiable.

The systems of equations we study are, for two non-isomorphic tensors T, T', the equations (1) along with new matrices X', Y', Z', and equations saying that these are the inverses of X, Y, Z, resp., viz.: $XX' = X'X = \mathrm{Id}$, and similarly for the others. The reason for introducing these new matrices, despite their not appearing in (1), is that these invertibility equations are only degree 2. In contrast, if we instead used the determinant to indicate that X was invertible, then our starting equations would have degree $n+1$, rather than constant degree ≤ 3. Since the main complexity measure we study on PC is degree, having starting equations of degree n would make it difficult to make meaningful lower bound statements.

Our first main result is (two proofs of) a lower bound on such techniques.

▶ **Theorem 1.1.** *Over any field, there are instances of $n \times n \times n$ Tensor Isomorphism that require PC degree $\Omega(n)$ to refute. Over $\mathbb{R}$, they also require Sum-of-Squares degree $\Omega(n)$ to refute.*

The preceding goes by reduction from known lower bounds on PC for Graph Isomorphism [7, 8], but has the disadvantage (from the tensor point of view) that the resulting tensors are quite sparse: in one direction, one of the slices is supported on an $\Omega(n) \times n$ matrix and all the others slices have support size 1. In a second proof (Section 6), we get a polynomially worse lower bound $\Omega(\sqrt[4]{n})$, but with a reduction from Random 3XOR that is more direct. Indeed, we show that 3XOR itself can be viewed as a particular instance of a tensor problem *without* gadgets; gadgets are only then needed to reduce from that tensor problem to Tensor Isomorphism itself. In contrast, the lower bounds on PC for GI (*ibid.*) already use the Cai–Fürer–Immerman gadgets [15] to reduce from XOR-SAT, and then even further gadgets are needed to reduce from GI to TI.

Our technical contributions in the above theorem are thus three-fold:

1. We show that the known reductions from GI to TI can be carried out in low-degree PC;
2. We realize 3XOR very naturally as a tensor problem; and
3. We give new reductions from 3XOR, through a series of tensor-related problems, to TI, that work as many-one reductions of the decision problems that can be carried out in low-degree PC.

Complementing our lower bound, we also show that tensors of low rank are comparatively easy to test for (non)-isomorphism. Here, one of our upper bounds is in the weaker Nullstellensatz proof system (giving a stronger upper bound than only a PC upper bound). In the Nullstellensatz proof system, a proof that a system of equations $f_1 = \cdots = f_m = 0$ is unsatisfiable consists of polynomials g_i such that $\sum g_i f_i = 1$, and the Nullstellensatz degree is the maximum degree of any $g_i f_i$. The PC degree is always at most the Nullstellensatz degree, and the gap between the two can be nearly maximal for Boolean equations: $O(1)$ versus $\Omega(n/\log n)$ [13]. (For Boolean equations, there is always an $O(n)$ upper bound, though this does not apply to TI, see Remark 1.3 below).

▶ **Theorem 1.2.** *Over any field, the Nullstellensatz degree of refuting isomorphism of two $n \times n \times n$ tensors of tensor rank $\leq r$ is at most $2^{O(r^2)}$. If working over a finite field $_q$ and including the equations $x^q - x$, the PC degree is at most $O(qr^2)$.*

In particular, isomorphism of constant-rank tensors can be decided in polynomial time.

▶ Remark 1.3. In many settings in proof complexity, Boolean axioms such as $x_i^2 = x_i$ or $x_i^2 = 1$ are included among the system of equations, and all such unsatisfiable systems of equations can be refuted in degree $O(n)$ ($n =$# variables). If this were the case here, the above would only be interesting for very small values of r. In contrast, the equations for TI do not include any such Boolean axioms, and as such the naive degree upper bound is exponential in the number of variables. For $n \times n \times n$ tensors, this gives an upper bound of $2^{O(n^2)}$ [48], and thus, Theorem 1.2 gives nontrivial upper bounds all the way up to $r \leq n$. (We note that $n \times n \times n$ tensors can have rank up to $\Theta(n^2)$ [34].) The proof of Theorem 1.2 shows that for rank-r tensors, TI can essentially be reduced to a system of equations in only $O(r^2)$ variables.

▶ Remark 1.4. For fixed r, testing if an $n \times n \times n$ tensor has rank $\leq r$ can be done in polynomial time, as follows. This will show that the algorithm of Theorem 1.2 genuinely solves the decision problem, and not just a promise problem. Given an $n \times n \times n$ tensor T, consider its three $n \times n^2$ flattenings. Use Gaussian elimination to put each such flattening, separately, into reduced row echelon form. If any of these flattenings has rank $> r$, reject. Otherwise, we get from this a list of $3r$ vectors $u_1, \ldots, u_r, v_1, \ldots, v_r, w_1, \ldots, w_r$, such that T lives in the $r \times r \times r$-dimensional space $\mathrm{Span}\{u_1, \ldots, u_r\} \otimes \mathrm{Span}\{v_1, \ldots, v_r\} \otimes \mathrm{Span}\{w_1, \ldots, w_r\}$. Now in this space we can write down the Brent equations [11] for T to have rank $\leq r$, which will be r^3 cubic equations in $3r^2$ variables (Brent's equations [11, (5.06)] were specifically for the matrix multiplication tensor, but analogous equations are easily constructed for arbitrary tensors using the same idea). Since r is constant, these equations may be solved in polynomial time (here we assume that we are either working over a finite field, a finite-degree extension of the rationals – see, for example, Grigoriev [22] – or in the BSS model over an arbitrary field).

Lastly, one may wonder why we focus on 3-Tensor Isomorphism, and not some of its many related variants. Indeed, just as there are other equivalence notions for matrices – such as conjugacy XMX^{-1} and congruence XMX^T – there are many different kinds of multilinear objects that can be represented by multi-way arrays, including tensors, homogeneous polynomials (commutative or noncommutative), alternating matrix spaces, multilinear maps, and so on, each with their own corresponding notion of isomorphism. While these problems are indeed distinct, they are all equivalent under polynomial-time isomorphisms [19, 26]; such problems are called TI-complete. Even isomorphism of k-way tensors (for any fixed $k \geq 3$) is equivalent to isomorphism of 3-tensors [26]. This partially justifies our focus on 3-Tensor Isomorphism. In the course of proving our reductions for the results stated above, we use many of the gadgets from [19, 26], and show that such uses also often yield proof complexity reductions as well. Because of the variety of gadgets used in our reductions, we believe that many, if not all, of the gadgets from those results would also yield proof complexity reductions, so the proof complexity of all the known TI-complete problems should be polynomially related.

1.2 Comparison with linear algebra, a new proof system, and a conjecture

As linear algebra is part of the core toolkit for understanding tensors, it is natural to wonder how linear algebra can help in algebraic proof complexity approaches to TI. We believe that even if it had the "full power" of linear algebra at its disposal "for free," PC could still not solve TI efficiently. We begin to make this precise in this section.

Some basic derivations in linear algebra are to relate the ranks of two matrices and to derive $BA = I$ from $AB = I$ (the Inversion Principle, one of the so-called "hard matrix identities" [47], only recently shown to have short NC^2-Frege proofs [29]). Soltys [46] and Soltys & Cook [47] discuss the relationship between these and other standard implications in linear algebra. We show that PC is not strong enough to prove these in low-degree:

▶ **Theorem 1.5.** *The unsatisfiable system of equations $XY = \mathrm{Id}_n$ where X is $n \times r$ and Y is $r \times n$ with $1 \leq r < n$, requires degree $\geq r/2 + 1$ to refute in PC, over any field.*

We refer to this system of equations as the Rank Principle, as refuting them amounts to showing that $\mathrm{rk}\,\mathrm{Id}_n > r$.

▶ **Theorem 1.6.** *Any PC derivation of $BA = I$ from $AB = I$, where A, B are $n \times n$ matrices with $\{0, 1\}$ entries, requires degree $\geq n/2 + 1$, over any field.*

We also observe that the Rank Principle can be derived in low degree from the Inversion Principle.

Although it remains open whether the Inversion Principle is "complete" for linear-algebraic reasoning (see [46, 47]), we introduce the proof system PC+Inv in an attempt to capture some linear-algebraic reasoning that seems potentially useful for TI. PC+Inv has all the same derivation rules as PC, but in addition, for any square matrices A, B (whose entries may themselves be polynomials – that is, we allow substitution instances), we have the rule

$$\frac{AB = I}{BA = I}.$$

where the antecedent represents the set of n^2 equations corresponding to $AB = I$, and similarly the consequent denotes the set of n^2 equations $BA = I$ (see 2.3 for more details). Degree is still measured in the usual way, but this rule lets us "cut out" the high-degree proof that would usually be required to derive $BA = I$ from $AB = I$. We now formalize our intuition that linear algebra should not suffice to solve TI efficiently in the following:

▶ **Conjecture 1.7.** *Tensor Isomorphism for $n \times n \times n$ tensors requires degree $\Omega(n)$ in PC+Inv, over any field.*

Despite the conjecture, we do not yet know how to prove lower bounds on PC+Inv for any unsatisfiable system of equations, let alone those coming from TI. Mod p counting principles (for p different from the characteristic of the field) strike us as potentially interesting instances to examine for PC+Inv lower bounds, before tackling a harder problem like TI. In the final section, we highlight many other open questions around the proof complexity of TI.

1.3 Organization

In Section 2 we cover preliminaries. In Section 3 we prove the lower bounds on linear algebraic principles just discussed. In Section 4 we prove the upper bound for isomorphism of bounded rank tensors (Theorem 1.2). In Section 5 we prove Theorem 1.1 by reduction from GI. In Section 6 we prove the polynomially related lower bound by direct reduction from Random 3XOR.

2 Preliminaries

2.1 Proof systems

All our rings are commutative and unital. *Polynomial calculus* (PC) is a proof system to prove that a given system of (multivariate) polynomial equations $\mathcal{P}$ over a field $\mathbb{F}$ of the form $p = 0$, has no solution over the algebraic closure (i.e. the system is unsolvable). We usually shorten the polynomial equation $p = 0$ to just p. The derivation rules of the system are the following one:

$$\frac{p}{xp} \text{ (multiplication)}, \quad \frac{p \quad q}{ap + bq} \text{ (linear combination)}$$

where x is any formal variable, $a, b \in \mathbb{F}$ and p, q are polynomials over $\mathbb{F}$.

When refuting Boolean systems of equations it is common to include the Boolean axioms $x_i^2 - x_i$. Because we do *not* always include these (esp. for TI) we are explicit about our use of these, but do not assume they are built into the proof system – that is, if we are assuming them as axioms, we say so.

A *PC derivation (or proof)* of a polynomial q from a set of polynomials $\mathcal{P}$ is a sequence of polynomial equations $p_1, \ldots, p_m$ ending with the polynomial q (so p_m is q) and where each p_i, $i \in [m]$, is either an *axiom* p for $p \in \mathcal{P}$, or is obtained from previous equations in the refutation by multiplication or linear combination. We denote this by writing $\mathcal{P} \vdash q$. Observe that if p is derivable in PC and q is a polynomial then, by repeated applications of multiplication and linear combination rules, we can derive pq. We often use this generalization of the multiplication in our proofs without mention.

A *PC refutation* is just a PC proof of the polynomial 1. The *degree* of a PC derivation is the maximal degree of a polynomial used in the proof. The *size* of a polynomial p is the number of terms in p. The *size* a PC derivation $p_1, \ldots, p_m$ is the sum of the sizes of the polynomials $p_1, \ldots, p_m$.

For our upper bound in Theorem 1.2, we also consider another algebraic proof system, known as *Nullstellensatz* (NS), to certify unsolvability of sets of polynomial equations. Nullstellensatz is defined in a static form as follows: a refutation of a list $\mathcal{P} = (p_1, \ldots, p_m)$ of polynomial equations over variables $x_1 \ldots, x_n$ is given by the list of polynomials $\mathcal{Q} = (q_1, \ldots q_m)$ such that

$$\sum_{i \in [m]} p_i q_i = 1$$

The *degree* of a NS refutation is the maximal degree of a polynomial in $\mathcal{P} \cup \mathcal{Q}$. The size of NS proof is the sum of the number of monomials appearing in the polynomials $q_1, \ldots, q_m$.

Sum-of-Squares (SOS) is a static proof system for certifying the unsolvability of systems of polynomial equations and polynomial inequalities, where polynomials are usually over the ring $\mathbb{R}[x_1 \ldots, x_n]$.

A polynomial p is a *sum-of-squares* polynomial if it is in the form $p = \sum_i r_i^2$ and the r_i's are polynomials as well. Given a system made by a set of polynomial equations $\mathcal{P} = \{p_1 = 0, \ldots p_m = 0\}$ and a set $\mathcal{Q} = \{q_1 \geq 0, \ldots q_k \geq 0\}$ of polynomial inequalities, a *sum-of-squares* proof of the polynomial inequality $p \geq 0$ from $\mathcal{P} \cup \mathcal{Q}$ is given by the formal identity

$$p = s_0 + \sum_{i \in [k]} s_i q_i + \sum_{j \in [m]} t_j p_j$$

where $s_0, s_1, \ldots, s_k$ are sum-of-squares polynomials, while $t_1, \ldots, t_m$ are arbitrary polynomials. When the system $\mathcal{P} \cup \mathcal{Q}$ is unsatisfiable, a *refutation* of $\mathcal{P} \cup \mathcal{Q}$ is a proof of the inequality $-1 \geq 0$, that is for p the constant polynomial -1. The *degree* of the proof is the $\max\{\deg(p), \deg(s_0), \deg(s_i) + \deg(q_i), \deg(t_j) + \deg(p_j) | i \in [k], j \in [m]\}$.

▶ **Definition 2.1** (PC reduction between systems of polynomials, cf. [14, Sec. 3]). *Let $P(x_1, \ldots, x_n)$ and $Q(y_1, \ldots, y_m)$ be two sets of polynomials over a field $\mathbb{F}$. P is (d_1, d_2)-*reducible *to Q if:*

1. *For each $i \in [m]$ there is a polynomial $r_i(\mathbf{x})$ of degree at most d_1 (which we think of as defining y_i in terms of the $\mathbf{x}$ variables);*
2. *There exists a degree d_2 PC derivation of $Q(r_1(\mathbf{x}), \ldots, r_m(\mathbf{x}))$ from polynomials $P(\mathbf{x})$.*

▶ **Lemma 2.2** ([14, Lem. 1]). *If $P(\mathbf{x})$ is (d_1, d_2)-reducible to $Q(\mathbf{y})$ and there is a degree d PC refutation of $Q(\mathbf{y})$, then there is a degree $\max(d_2, d_1 d)$ refutation of $P(\mathbf{x})$.*

In their paper, they typically only applied this to systems of equations which were known to be unsatisfiable (such as PHP and Tseitin tautologies), whereas in our paper we have several situations we want to combine the above notion together with the usual notion of many-one reduction. We encapsulate this in the following definition. We say a decision problem Π is a *polynomial solvability problem* over a field if all valid instances of the problem are systems of polynomial equations over , and the problem is to decide whether such a system of equations has solutions over the algebraic closure . Thus, the difference between multiple polynomial solvability problems is just *which* systems of equations are valid inputs.

▶ **Definition 2.3** (PC many-one reduction). *Let Π_1, Π_2 be two polynomial solvability problems over a field . We say that Π_1* (d_1, d_2)-many-one reduces *to Π_2 if there is a polynomial-time many-one reduction ρ from Π_1 to Π_2, such that for all unsatisfiable instances $\mathcal{F}$ of Π_1, $\mathcal{F}$ (d_1, d_2)-reduces to $\rho(\mathcal{F})$. When this occurs with $d_1, d_2 = O(1)$, we write*

$$\Pi_1 \leq_m^{PC} \Pi_2.$$

2.2 Linear algebra and tensors

Given three vector spaces U, V, W over a field , a 3-tensor is an element of the vector space $U \otimes V \otimes W$, whose dimension is $(\dim U)(\dim V)(\dim W)$. If e_i is the i-th standard basis vector, then a basis for $U \otimes V \otimes W$ is given by the vectors $\{e_i \otimes e_j \otimes e_k\}$. One may also interpret the symbol $\otimes$ more concretely as the Kronecker product, in which $e_i \otimes e_j \otimes e_k$ represents a 3-way array whose only nonzero entry is in the (i, j, k) position. The vector space of such 3-way arrays (with coordinate-wise addition) is isomorphic to $U \otimes V \otimes W$.

The rank of a tensor $T \in U \otimes V \otimes W$ is the minimum r such that $T = \sum_{i=1}^{r} u_i \otimes v_i \otimes w_i$ for some vectors u_i, v_i, w_i.

Two $n \times m \times p$ 3-tensors $T, T' \in U \otimes V \otimes W$ are isomorphic if there exist matrices $X \in \mathrm{GL}(U), Y \in \mathrm{GL}(V), Z \in \mathrm{GL}(W)$ such that $(X, Y, Z) \cdot T = T'$, where the latter is shorthand for (1). If we treat T, T' as given non-isomorphic tensors, then we may treat (1) as a system of equations in the $n^2 + m^2 + p^2$ variables $X_{ii'}, Y_{jj'}, Z_{kk'}$. To enforce that these variable matrices are invertible, we furthermore introduce three additional sets of variables X', Y', Z' meant to be the inverse matrices, and include also the equations

$$XX' = X'X = I_n \qquad YY' = Y'Y = I_m \qquad ZZ' = Z'Z = I_p,$$

where I_n denotes the $n \times n$ identity matrix, which is Id_U in any basis. (We could have instead introduced new variables such as δ and the equation $\det(X)\delta = 1$, however, the latter equation is degree n, whereas the above equations all have degree $O(1)$, which is more desirable from the point of view of algebraic proof complexity.)

2.3 Polynomial encodings and the inversion principle

Some principles of linear algebra can be formulated as tautologies in propositional logic and therefore also as a set of polynomial equations. In this paper we preliminarily consider two such principles.

Rank Principle. As a first example we consider a set of unsatisfiable polynomials encoding the principle that the product of a $n \times r$ matrix X by a $r \times n$ matrix Y cannot be the identity matrix whenever $r < n$. We consider variables $x_{i,k}, y_{j,k}$ for $i, j \in [n]$ and $k \in [r]$, where $r < n$ to encode X and Y. Then the polynomial encoding is:

$$\mathbb{I}(r, n) := \sum_{k \in [r]} x_{i,k} y_{j,k} - \delta_{i,j} \qquad i, j \in [n]$$

where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. This set of polynomials is clearly unsatisfiable as long as $r < n$.

Inversion Principle. The second principle encodes the invertibility of a square $n \times n$ matrix A, expressing the tautology that $AB = I \to BA = I$ where A, B are $n \times n$ matrices and I is the identity matrix. Stephen A. Cook suggested this principle as a tautology that may be hard to prove in several proof systems.

Let $a_{i,j}, b_{i,j}$ be formal variables encoding respectively the (i, j)-th entries of A and B. We represent the fact that $AB = I$ as the set of degree 2 polynomials

$$\sum_{k \in [n]} a_{i,k} b_{k,j} - \delta_{i,j} \qquad i, j \in [n],$$

where $\delta_{i,j} = 1$ if $i = j$ and 0 otherwise. We denote this set of polynomials by $AB = I$. In Section 3, we study the degree complexity of $AB = I \vdash BA = I$, that is of PC derivations of the polynomials $BA = I$ from the polynomials $AB = I$.

In view of the results we obtain in Section 3, in Section 1.2 we considered a *polynomial rule schema* of the form

$$\frac{AB = I}{BA = I}$$

which we call the *Inversion Rule* (INV) meant to be added to PC as an extra rule. We make this slightly more precise here.

A polynomial instantiation τ of the polynomials $AB = I$ is a substitution of polynomials $p_{i,j}, q_{i,j}$ to variables $a_{i,j}$ and $b_{i,j}$. In PC+INV a polynomial p is derivable from a set of polynomials $\mathcal{P}$ if

1. p is an axiom, or $p \in \mathcal{P}$;
2. p is obtained by multiplication or linear combination from previous polynomials in the proof;
3. p is a polynomial among a polynomial instantiation τ of $BA = I$, given that among the polynomials previously derived in the proof there are all the polynomials forming the instantiation τ of $AB = I$.

Pigeonhole Principle. An important role in proving the results in Section 3 is played by the well-known *Pigeonhole principle* stating that any function f from $[n]$ to $[r]$ with $r < n$ has a collision, that is there are $i \neq i' \in [n]$ and a $j \in [r]$ such that $f(i) = f(i') = j$. PHP_r^n is the set of polynomials:

$$\sum_{k\in[r]} p_{i,k} - 1, \text{ for } i \in [n], \qquad p_{i,k}p_{j,k}, \text{ for } i \neq j \in [n], k \in [r]$$

$$p_{ij}^2 - p_{ij}, \text{ for } i \in [n], j \in [r]$$

Razborov [43] additionally included the "functional equations" (encoding that each pigeon cannot be matched to more than one hole):

$$p_{i,k}p_{i,k'}, \text{ for } i \in [n], k \neq k' \in [r].$$

3 Linear algebra warm-up: PC for matrices

Two matrices $M, M' \in U \otimes V$ are isomorphic as tensors if they are equivalent as matrices, meaning under left- and right-multiplication by invertible matrices $X \in \mathrm{GL}(U), Y \in GL(V)$, that is,

$$XMY = M'.$$

Since we want X, Y to be invertible, we also introduce variable matrices X', Y' as before, together with the equations

$$XX' = X'X = \mathrm{Id}_U \qquad YY' = Y'Y = \mathrm{Id}_V .$$

Then by left multiplying our initial matrix equation by Y', we may replace it with the new matrix equation

$$XM = M'Y'.$$

The latter has the advantage of being linear in X and Y', but the quadratic equations $XX' = \mathrm{Id}_U, YY' = \mathrm{Id}_V$ still make even this case not totally obvious.

3.1 A trick for PC degree

If our focus is on PC *degree*, we note that the degree of the equations is unchanged if we first left- or right-multiply M, M' by invertible scalar matrices. For example, if we replace M by $\overline{M} = AMB$ with $A, B \in \mathrm{GL}(U)$, then we may replace X by $\overline{X} := XA^{-1}$, Y by $\overline{Y} := B^{-1}Y$. Then we have $\overline{M} \cong M$, so $\overline{M} \cong M'$ iff $M \cong M'$. Furthermore, since the transformation $X \mapsto XA^{-1}$, $Y \mapsto B^{-1}Y$ is linear and invertible, any PC proof that $\overline{M} \ncong M'$ can be transformed by the inverse linear transformation into a PC proof that $M \ncong M'$ of the same degree.

Now, for matrices under this equivalence relation, we have a normal form, namely every matrix M is equivalent to a diagonal matrix with $\mathrm{rk}(M)$ 1s on the diagonal and all the remaining entries 0, that is, $\sum_{i=1}^{\mathrm{rk}(M)} e_i \otimes e_i = I_r \oplus 0$, where the latter 0 denotes a 0 matrix of appropriate size $(n-r) \times (m-r)$. So by using the preceding trick, we may put both M and M' in this form. The two are isomorphic iff $\mathrm{rk}(M) = \mathrm{rk}(M')$, so for PC degree we have now reduced to the case of showing that $I_r \oplus 0$ and $I_{r'} \oplus 0$ are not isomorphic when $r \neq r'$.

Note that, aside from the equations saying X and Y are invertible, this is almost identical to the Rank Principle (see Section 2.3). In the rest of this section we will prove PC lower bounds on both the Rank Principle and the Inversion Principle.

3.2 Inversion Principle implies the Rank Principle

▶ **Lemma 3.1.** *If the $r \times r$ Inversion Principle has a degree d PC derivation, then there is a degree $\max\{d, 3\}$ PC refutation of the Rank Principle stating that a rank r matrix is not equivalent (isomorphic) to a rank n matrix, for any $n > r$.*

If the Inversion Principle has a degree d NS derivation, then the Rank Principle has a degree $d + 2$ NS refutation.

Proof. Suppose the $r \times r$ Inversion Principle has a degree-d derivation. Consider the Rank Principle $XY = I_n$ where X is $n \times r$ and Y is $r \times n$, with $n > r$. Write

$$X = \begin{bmatrix} X_0 \\ X_1 \end{bmatrix} \text{ and } Y = \begin{bmatrix} Y_0 & Y_1 \end{bmatrix},$$

where X_0, Y_0 are $r \times r$. Then, examining the upper-left $r \times r$ corner of the original equations, we find $X_0 Y_0 = I_r$. As these are square matrices, by assumption in degree d we may then derive that $Y_0 X_0 = I_r$ as well.

Now, multiply both sides of $XY = I_n$ on the left by the matrix $\begin{bmatrix} Y_0 & 0 \\ 0 & I_{n-r} \end{bmatrix}$. The result is then the set of degree-3 equations

$$\begin{bmatrix} Y_0 X_0 \\ X_1 \end{bmatrix} \begin{bmatrix} Y_0 & Y_1 \end{bmatrix} = \begin{bmatrix} Y_0 & 0 \\ 0 & I_{n-r} \end{bmatrix}.$$

Considering the upper-right $r \times (n - r)$ block of these equations, we find the equations $Y_0 X_0 Y_1 = 0$.

But now, from the equation $Y_0 X_0 = I_r$, we may right-multiply by Y_1 to get $Y_0 X_0 Y_1 = Y_1$. Combining with the equation at the end of the last paragraph, we then conclude $Y_1 = 0$.

Finally, consider the lower-right $(n - r) \times (n - r)$ part of the original equation $XY = I_n$, namely, $X_1 Y_1 = I_{n-r}$. We had already derived $Y_1 = 0$, which we can then left-multiply by X_1 to get $X_1 Y_1 = 0$. Considering any diagonal entry of these two equations, we then derive the contradiction $1 = 0$.

To see the NS certificate, we unwrap the above proof. First write $Y_0 X_0 - I_r$ as a linear combination of the equations $X_0 Y_0 - I_r$ with polynomial coefficients, in total degree d. Among our starting equations in the Rank Principle, we have $X_0 Y_1$ and $X_1 Y_1 - I_{n-r}$. Then the following linear combination has degree 2 more than $Y_0 X_0 - I_r$, and derives 1 in any of its diagonal entries:

$$-X_1 Y_0 \underline{X_0 Y_1} + X_1 (Y_0 X_0 - I_r) Y_1 + \underline{(X_1 Y_1 - I_{n-r})}. \qquad \blacktriangleleft$$

▶ **Observation 3.2.** *The $n \times n$ Inversion Principle has a proof of degree $2n + 2$.*

Proof. The idea is to use Laplace expansion. We spell out the details.

We start with $XY = I_n$, where X and Y are $n \times n$ matrices of variables. Left-multiply by Y to get $YXY = Y$, and then right multiply by $Adj(Y)$ (whose entries are the $(n-1) \times (n-1)$ cofactors of Y, hence have degree $n - 1$) to get $YXYAdj(Y) = YAdj(Y)$. Now, by Laplace expansion, we have $YAdj(Y) \equiv \det(Y) I_n$, so we get $YX \det(Y) = \det(Y) I_n$.

Next, starting from $XY = I_n$ and expanding out the determinant term-by-term, we derive $\det(XY) = 1$. (Note that here, we are not simply applying the determinant to the matrix $XY - I$, as that would give us the value of the characteristic polynomial evaluated at 1. Instead, we repeatedly use that from $a - b = 0$ and $c - d = 0$ we can derive $ac - bd = 0$ as $\underline{(a - b)}c + b\underline{(c - d)}$. Similarly, we can derive $(a+c)-(b+d) = 0$ as $(a-b)+(c-d)$.) Now, since $\det(XY) \equiv \det(X) \det(Y)$ identically as polynomials, we have derived $\det(X) \det(Y) = 1$ in degree n.

Now, from $YX\det(Y) - \det(Y)I_n$ in the first paragraph, we multiply by $\det(X)$ to get $(YX - I_n)(\det(X)\det(Y))$. From $\det(X)\det(Y) - 1$ in the second paragraph, we multiply by $-(YX - I_n)$ and add to the preceding to get $YX - I_n$, all in degree at most $2n + 2$. ◀

3.3 Lower bound on the Rank Principle (and Inversion Principle) via reduction from PHP

Here we show that the Rank Principle (see Section 2.3) requires large PC degree, via a reduction to the Pigeonhole Principle. For the Pigeonhole principle, a tight PC degree lower bound is known:

▶ **Theorem 3.3** (Razborov [43]). *Any PC refutation of the Functional PHP_r^n requires degree $r/2+1$ over any field.*

We use this to show:

▶ **Theorem 3.4.** *Let $n \in \mathbb{N}$, $n \geq 2$ and $1 \leq r < n$. $\mathbb{I}(r, n)$ (with or without the Boolean axioms) requires degree $r/2 + 1$ in PC over any field.*

Proof. We prove that PHP_r^n is $(1, 2)$-reducible to $\mathbb{I}(r, n)$. First we consider the following degree 1 polynomials defining x and y variables of $\mathbb{I}(r, n)$ in terms of the p variables of PHP_r^n. variables

$$x_{i,k} = y_{i,k} = p_{i,k} \qquad \text{for } i \in [n], k \in [n-1].$$

Second we show a degree 2 PC proof of $\mathbb{I}(r, n)$ from the polynomials defining the PHP_r^n. From PHP axioms $p_{i,k}p_{k,j}$ for $i, j \in [n], i \neq j$, and summing over all $k \in [r]$, we get

$$\sum_{k\in[r]} p_{i,k}p_{k,j},$$

which are exactly the axioms of $\mathbb{I}(r, n)$ for $i \neq j, i, j \in [n]$, after the substitution of variables.

For a $i \in [n]$, take the boolean axioms written in the form $p_{i,k}p_{i,k} - p_{i,k}$ and sum them over $k \in [r]$:

$$\sum_{k\in[r]} p_{i,k}p_{i,k} - \sum_{k\in[r]} p_{i,k}$$

Summing this last polynomial with the PHP axiom $\sum_{k\in[r]} p_{i,k} - 1$ we get the polynomial

$$\sum_{k\in[r]} p_{i,k}p_{i,k} - 1,$$

which is the axiom of $\mathbb{I}(r, n)$ for $i = j$ after the substitution of the variables. The proof has degree 2. The result follows immediately from Lemma 2.2 and Theorem 3.3. ◀

▶ **Corollary 3.5.** *Any PC proof of $AB = I \vdash BA = I$, where A, B are square $n \times n$ $\{0, 1\}$ matrices requires degree $n/2 + 1$.*

Proof. Follows immediately from Theorem 3.4 and Lemma 3.1. ◀

4 Upper bound for non-isomorphism of bounded-rank tensors

▶ **Theorem 4.1.** *Over any algebraically closed field, there is a function $f(r) \leq 2^{O(r^2)}$, depending only on r, such that, given two non-isomorphic tensors M, M' of tensor rank $\leq r$, the Nullstellensatz degree of refuting isomorphism is at most $f(r)$.*

If working over a finite field $GF(q)$ and including the equations $x^q - x = 0$ for all variables x, then the PC degree is at most $12qr^2$.

Proof. The proof is based mainly on the so-called inheritance property of tensor rank.

Let $M = \sum_{i=1}^{r} u_i \otimes v_i \otimes w_i$ and let $M' = \sum_{i=1}^{r} u'_i \otimes v'_i \otimes w'_i$ be our two tensors of format $n_1 \times n_2 \times n_3$. Let $d_1 = \dim \mathrm{Span}\{u_1, u_2, \ldots, u_r, u'_1, u'_2, \ldots, u'_r\}$, d_2 similarly for the v's and d_3 for the w's. Choose a basis $e_1, e_2, \ldots, e_{n_1}$ for ^{n_1} such that $\mathrm{Span}\{e_1, \ldots, e_{d_1}\} = \mathrm{Span}\{u_1, \ldots, u_r, u'_1, \ldots, u'_r\}$. Let $f_1, \ldots, f_{n_2}$ be a similar basis for ^{n_2} (with the first d_2 vectors a basis for $\mathrm{Span}\{v_1, \ldots, v_r, v'_1, \ldots, v'_r\}$), and similarly $g_1, \ldots, g_{n_3}$. Changing everything in sight into the $e_\bullet \otimes f_\bullet \otimes g_\bullet$ basis, we find that M, M' are both supported in the upper-left $d_1 \times d_2 \times d_3$ sub-tensors, with all zeros outside of this. Call the corresponding $d_1 \times d_2 \times d_3$ tensors $\overline{M}, \overline{M}'$. Because all the entries outside this box are zero, it is not difficult to show that $M \cong M'$ iff $\overline{M} \cong \overline{M}'$ (the so-called "Inheritance Theorem,", see, e. g., [31, §3.7.1]); note that isomorphism of $\overline{M}$ with $\overline{M}'$ is via the much smaller group $\mathrm{GL}_{d_1} \times \mathrm{GL}_{d_2} \times \mathrm{GL}_{d_3}$, rather than $\mathrm{GL}_n \times \mathrm{GL}_n \times \mathrm{GL}_n$ (the latter of which is used to determine isomorphism of M with M').

In this basis, isomorphism of $\overline{M}, \overline{M}'$ is solely determined by the upper-left $d_1 \times d_1$ sub-matrix of X, X', the upper-left $d_2 \times d_2$ submatrix of Y, Y', and the upper-left $d_3 \times d_3$ sub-matrix of Z, Z'. So we now only need to deal with equations in $d_1^2 + d_2^2 + d_3^2$ variables. Since each $d_i \leq 2r$, this is at most $12r^2$ variables.

Since we have $\leq 12r^2$ variables, $d_1 d_2 d_3$ cubic equations, and $6n^2$ quadratic equations ($XX' = I = X'X = YY' = \cdots$), over an algebraically closed field Sombra's Effective Nullstellensatz [48] implies that the Nullstellensatz degree of refuting our equations is then at most $4 \cdot 3^{\Theta(r^2)}$.

Over a finite field with the extra equations $x^q = x$, we may reduce degrees so that the degree of each variable is never more than q, the size of the field. In this case, the PC degree is at most q times the number of variables, i. e., at most $12qr^2$. ◀

5 Lower bound on PC degree for Tensor Isomorphism from Graph Isomorphism

▶ **Definition 5.1.** *Given two graphs G, H with adjacency matrices A, B (resp.), the equations for Graph Isomorphism (the same as those used by Berkholz & Grohe [7, 8]) are as follows. Let Z be an $n \times n$ matrix of variables z_{ij} (where the intended interpretation is that $z_{ij} = 1$ iff an isomorphism maps vertex $i \in V(G)$ to vertex $j \in V(H)$). We say that a partial map, which sends $(i, i') \mapsto (j, j')$ is a local isomorphism if (1) $i = i'$ iff $j = j'$ (it's a well-defined map) and (2) $(i, i') \in E(G) \Leftrightarrow (j, j') \in E(H)$. (One may also do Colored Graph Isomorphism and require that the colors match, $c(i) = c(j), c(i') = c(j')$.) Then the equations are:*

$z_{ij}^2 - z_{ij}$	$\forall i, j$	*All variables $\{0, 1\}$-valued*
$1 - \sum_i z_{ij}$	$\forall j$	*each $j \in V(H)$ is mapped to from exactly one vertex*
$1 - \sum_j z_{ij}$	$\forall i$	*each $i \in V(G)$ maps to exactly one vertex*
$z_{ij} z_{i'j'}$		*Whenever $(i, i') \mapsto (j, j')$ is not a local isomorphism.*

In this section, we prove a lower bound on PC (and SoS) for TI, by reducing from GI and using the known lower bounds on GI [7, 8]. Specifically, we show

▶ **Theorem 5.2.** *Over any field, there are instances of* Tensor Isomorphism *of size* $O(n) \times O(n) \times O(n)$ *that require PC degree* $\Omega(n)$ *to refute. The same holds over the reals for SoS degree.*

Proof. Berkholz and Grohe [7, 8] show the same statement for n-vertex graphs of bounded vertex degrees, with the same PC/SoS degree bound. In Proposition 5.4 we show that GI reduces to Monomial Code Equivalence by a (2,4)-many-one reduction that turns n-vertex, m-edge graphs into $m \times (3m + n)$ matrices. in Proposition 5.5 we show that Monomial Code Equivalence reduces to TI by a (2,4)-many-one reduction that turns $k \times N$ matrices into $(k + 2N) \times N \times (1 + 2N)$ tensors. By Lemma 2.2, this completes the proof. ◀

To reduce from GI to TI we use the following intermediate problem. A matrix is *monomial* if it has exactly one nonzero entry in each row and column; equivalently, a monomial matrix is the product of a permutation matrix and an invertible diagonal matrix.

▶ **Definition 5.3.** *Monomial Code Equivalence is the problem: given two* $k \times n$ *matrices* C, C'*, do there exist matrices* X, Y *such that* $XCY^T = C'$ *where* X *is invertible and* Y *is invertible and monomial? Given two such matrices* C, C'*, the equations for Monomial Code Equivalence are as follows. There are* $2(k^2 + n^2)$ *variables arranged into matrices* X, X' *(of size* $k \times k$*) and* Y, Y' *(of size* $n \times n$*). The equations are*

$$XCY^T = C' \qquad XX' = X'X = \mathrm{Id} \qquad YY' = Y'Y = \mathrm{Id}$$

and

$$y_{ij}y_{ij'}(\forall i \forall j \neq j') \qquad y_{ij}y_{i'j}(\forall i \neq i', \forall j)$$
$$y'_{ij}y'_{ij'}(\forall i \forall j \neq j') \qquad y'_{ij}y'_{i'j}(\forall i \neq i', \forall j)$$

(Note: there are no equations forcing the variables to take on values in $\{0, 1\}$*.)*

▶ **Proposition 5.4.** *The reduction of Petrank & Roth [42] from Graph Isomorphism to Linear Code Equivalence over* $_2$ *in fact gives a (2,4)-many-one reduction from Graph Isomorphism to Monomial Code Equivalence (sic!) over any field.*

Proof. The reduction of Petrank & Roth is as follows: given a simple undirected graph G with n vertices and m edges, let $D(G)$ be its $m \times n$ incidence matrix: $D_{e,v} = 1$ iff $v \in e$ and is 0 otherwise, and let $M(G)$ be the $m \times (3m + n)$ matrix

$$M(G) = \left[\ I_m \ \middle|\ I_m \ \middle|\ I_m \ \middle|\ D(G)\ \right].$$

Many-one reduction. It was previously shown (over $_2$ in [42] and over arbitrary fields in [23, Lem. II.4]) that this gives a many-one reduction to Permutational Code Equivalence. Here we observe that the same reduction also gives a reduction to Monomial Code Equivalence. Thus, all that remains to show is that if $M(G)$ and $M(H)$ are monomially equivalent, then G must be isomorphic to H.

In fact, what was shown in [42] (over arbitrary fields in [23]) is that, up to permutation and scaling of the rows, $M(G)$ is the unique generator matrix of its code satisfying the following properties: (1) $M(G)$ is $m \times (3m + n)$, (2) each row has Hamming weight ≤ 5, (3) any linear combination that includes two or more rows with nonzero coefficients has Hamming weight ≥ 6.

Now, suppose (X, Y) is a monomial equivalence of the codes $M(G), M(H)$. Then the rowspans of $M(G)Y^T$ and $M(H)$ are the same. Since Y is monomial, if we consider just the supports of the rows of $M(G)Y^T$, up to re-ordering the rows, by the preceding paragraph, those supports must be the same as the supports of the rows of $M(H)$. Thus X must also be monomial. Say $X = DP$ and $Y = EQ$ where D, E are diagonal and P, Q are permutation matrices. Then $PM(G)Q^T$ has the same support as $XM(G)Y^T = M(H)$, and since P and Q are permutation matrices and $M(G)$ and $M(H)$ have all entries in $\{0, 1\}$, we must have $PM(G)Q^T = M(H)$. Thus $M(G)$ and $M(H)$ are in fact equivalent by a *permutation* matrix (in place of the monomial matrix Y). Thus, by the fact that $(G, H) \mapsto (M(G), M(H))$ was a reduction to Permutational Code Equivalence, we conclude that $G \cong H$.

Low-degree PC reduction. Let X, X', Y, Y' be the variable matrices in the equations for Monomial Code Equivalence of $M(G), M(H)$, and let Z be the variable matrix in the equations for Graph Isomorphism of G, H. Let $n = |V(G)|, m = |E(G)|$; so, X, X' are of size m, Y, Y' are of size $3m + n$, and Z is of size n.

Let $Z^{(2)}$ denote the $\binom{n}{2} \times \binom{n}{2}$ matrix whose $(\{i, i'\}, \{j, j'\})$ entry is $z_{ij}z_{i'j'} + z_{ij'}z_{i'j}$. The idea is that if Z is a map on the vertices, then $Z^{(2)}$ is the corresponding map on the edges; the two terms come from the fact that the edge $\{i, i'\}$ can be mapped to the edge $\{j, j'\}$ either by $(i, i') \mapsto (j, j')$ or by $(i, i') \mapsto (j', j)$. Note that, since Z is a permutation matrix, at most one of these terms is nonzero, and thus $Z^{(2)}$ is also a $\{0, 1\}$-matrix (in fact, a permutation matrix). Let $Z^{(2)}_E$ denote the $|E| \times |E|$ submatrix of $Z^{(2)}$ all of whose row indices are $\{i, i'\} \in E(G)$ and all of whose column indices are $\{j, j'\} \in E(H)$. Note also that $(Z^{(2)}_E)^T = (Z^T)^{(2)}_E$, so we use these notations interchangeably for convenience.

Now consider the following substitution:

$$\begin{array}{ll} X \mapsto (Z^{(2)}_E)^T & Y \mapsto (Z^T)^{(2)}_E \oplus (Z^T)^{(2)}_E \oplus (Z^T)^{(2)}_E \oplus (Z^T) \\ X' \mapsto Z^{(2)}_E & Y' \mapsto Z^{(2)}_E \oplus Z^{(2)}_E \oplus Z^{(2)}_E \oplus Z \end{array}$$

After making these substitutions in the equations for Monomial Code Equivalence of $M(G), M(H)$, we get the equations

$$(Z^{(2)}_E)^T Z^{(2)}_E = Z^{(2)}_E (Z^{(2)}_E)^T = \mathrm{Id}_m \qquad (Z^{(2)}_E)^T D(G) Z = D(H) \qquad ZZ^T = Z^T Z = \mathrm{Id}_n \quad (2)$$

along with equations saying that Z and $Z^{(2)}_E$ are monomial.

We now show how to derive these equations in low-degree PC from the GI equations.

The monomial equations for Z are part of the GI equations, so there is nothing to do for those.

The monomial equations for $Z^{(2)}_E$ are of the form $(z_{ij}z_{i'j'} + z_{ij'}z_{i'j})(z_{k\ell}z_{k'\ell'} + z_{k\ell'}z_{k'\ell})$ where either (1) $\{i, i'\} = \{k, k'\}$ and $\{j, j'\} \neq \{\ell, \ell'\}$ or (2) vice versa. We expand out to get

$$z_{ij}z_{i'j'}z_{k\ell}z_{k'\ell'} + z_{ij}z_{i'j'}z_{k\ell'}z_{k'\ell} + z_{ij'}z_{i'j}z_{k\ell}z_{k'\ell'} + z_{ij'}z_{i'j}z_{k\ell'}z_{k'\ell}$$

We show how to get this equation in case (1); case (2) follows similarly, *mutatis mutandis*. In case (1), without loss of generality suppose that $i = k$, $i' = k'$, and $j \notin \{\ell, \ell'\}$. The first two terms are divisible by the GI equations $z_{ij}z_{i\ell}$ (since $i = k$ and $j \neq \ell$), the third term is divisible by $z_{i'j}z_{i'\ell'}$ (since $i' = k'$ and $j \neq \ell'$), and the last term is divisible by $z_{i'j}z_{i'\ell}$ similarly.

Next, the equations $ZZ^T = \mathrm{Id}_n$ are, expanded out,

$$\sum_j z_{ij}z_{ij} - 1 (\forall i) \qquad \sum_j z_{ij}z_{kj} (\forall i \neq k).$$

The first is gotten by linear combination from $1 - \sum_j z_{ij}$ and the Boolean axioms $z_{ij}^2 - z_{ij}$. The second is a linear combination of the monomial axioms $z_{ij}z_{kj}$ (part of the local non-isomorphism axioms). Similarly for $Z^TZ = \mathrm{Id}$, using $1 - \sum_i z_{ij}$ instead.

Next, we expand out the equations $Z_E^{(2)}(Z^T)_E^{(2)} = \mathrm{Id}_m$, to get[1]

$$\sum_{\{j,j'\}\in E(H)} (z_{ij}z_{i'j'} + z_{ij'}z_{i'j})(z_{kj}z_{k'j'} + z_{k'j}z_{kj'}) - \delta_{\{i,i'\},\{k,k'\}}(\forall\{i,i'\},\{k,k'\} \in E(G))$$

Thus, for $\{i,i'\} \neq \{k,k'\}$, we need to derive

$$\sum_{\{j,j'\}\in E(H)} \left(z_{ij}z_{i'j'}z_{kj}z_{k'j'} + z_{ij'}z_{i'j}z_{kj}z_{k'j'} + z_{ij}z_{i'j'}z_{k'j}z_{kj'} + z_{ij'}z_{i'j}z_{k'j}z_{kj'}\right).$$

Without loss of generality, suppose that $i \notin \{k,k'\}$. Then the first two terms of each summand are divisible by the GI equation $z_{ij}z_{kj}$, the third term is divisible by $z_{ij}z_{k'j}$, and the last term is divisible by $z_{ij'}z_{kj'}$. On the other hand, when $\{i,i'\} = \{k,k'\}$, we need to derive

$$-1 + \sum_{\{j,j'\}\in E(H)} \left(z_{ij}^2z_{i'j'}^2 + 2z_{ij'}z_{i'j}z_{ij}z_{i'j'} + z_{ij'}^2z_{i'j}^2\right).$$

The middle terms of each summand are divisible by the GI equations $z_{ij'}z_{ij}$. For the first and third terms, we can use the Boolean axioms to remove the squares, and thus we are left to derive

$$-1 + \sum_{\{j,j'\}\in E(H)} (z_{ij}z_{i'j'} + z_{ij'}z_{i'j}) \tag{3}$$

We derive this from the GI equations as follows. Consider $(\sum_j z_{ij} - 1)(\sum_{j'} z_{i'j'} - 1) + (\sum_j z_{ij} - 1) + (\sum_{j'} z_{i'j'} - 1)$ and break up the resulting sum according to whether $j = j'$, $\{j,j'\} \in E(H)$ or $\{j,j'\} \notin E(H)$. Then we get

$$\sum_j z_{ij}z_{i'j} + \sum_{j,j':\{j,j'\}\in E(H)} z_{ij}z_{i'j'} + \sum_{j\neq j'\{j,j'\}\notin E(H)} z_{ij}z_{i'j'} - 1$$

Every summand in the first sum is a monomial axiom since $i \neq i'$. Every summand in the third sum is a local non-isomorphism axiom, since $\{i,i'\} \in E(G)$ but $\{j,j'\} \notin E(H)$. Note that every edge $\{j,j'\}$ of $E(H)$ is represented twice in the middle sum: once as (j,j') and once as (j',j). Thus, the above simplifies to

$$\sum_{\{j,j'\}\in E(H)} (z_{ij}z_{i'j'} + z_{ij'}z_{i'j}) - 1,$$

which is what we sought to derive. The derivation of $(Z_E^{(2)})^TZ_E^{(2)} = \mathrm{Id}$ is similar.

Finally, we show how to derive the equation $(Z_E^{(2)})^TD(G)Z = D(H)$ from the equations $ZA(G) = A(H)Z$, where $A(G)$ denotes the adjacency matrix of G, with $A(G)_{ii'} = 1$ iff $\{i,i'\} \in E(G)$. Writing out the equations in indices, we need to derive, for all $\ell \in V(H)$ and all $\{j,j'\} \in E(H)$,

$$\sum_{\{i,i'\}\in E(G),k\in V(G)} \left(Z_E^{(2)}\right)_{\{i,i'\},\{j,j'\}} D(G)_{\{i,i'\},k}z_{k\ell} = D(H)_{\{j,j'\},\ell}$$

[1] We use the notation $\sum_{\{j,j'\}\in E(H)}$ to denote a sum in the index of summation takes on the value $e \in E(H)$ for each edge of H exactly once. Because our edges are undirected, we only use such sums when the summand expression is itself invariant under swapping the roles of j,j'. If so desired, one could equivalently say $\sum_{j<j',\{j,j'\}\in E(H)}$.

Using the fact that $D(G)_{\{i,i'\},k} = \delta_{ik} + \delta_{i'k}$ and the definition of $Z^{(2)}$, this is the same as

$$\sum_{\{i,i'\}\in E(G),k\in V(G)} (z_{ij}z_{i'j'} + z_{ij'}z_{i'j})\,(\delta_{ik}+\delta_{i'k})z_{k\ell} = \delta_{j\ell}+\delta_{j'\ell}(\forall \ell \in V(H), \forall\{j,j'\} \in E(H))$$

Thus we need to derive:

$$\sum_{\{i,i'\}\in E(G)} (z_{ij}z_{i'j'} + z_{ij'}z_{i'j})\,(z_{i\ell} + z_{i'\ell}) = \begin{cases} 1 & \ell \in \{j,j'\} \\ 0 & \text{otherwise.} \end{cases}$$

Expanding out the summand, we find the four terms

$$z_{ij}z_{i'j'}z_{i\ell} + z_{ij}z_{i'j'}z_{i'\ell} + z_{ij'}z_{i'j}z_{i\ell} + z_{ij'}z_{i'j}z_{i'\ell}.$$

When $\ell \notin \{j,j'\}$, each of these terms is divisible by one of the monomial (local non-isomorphism) axioms, respectively: $z_{ij}z_{i\ell}$, $z_{i'j'}z_{i'\ell}$, $z_{ij'}z_{i\ell}$, and $z_{i'j}z_{i'\ell}$.

Finally, when $\ell \in \{j,j'\}$, without loss of generality suppose that $\ell = j$. Then the only terms that are not divisible by the monomial axioms as above are $z_{ij}^2 z_{i'j'} + z_{ij'}z_{i'j}^2$. Using the Boolean axioms we can easily convert each such summand to $z_{ij}z_{i'j'} + z_{ij'}z_{i'j}$. The derivation of the sum of these over all $\{i,i'\} \in E(G)$ is analogous, *mutatis mutandis*, to the derivation of (3) above. This completes the proof. ◀

▶ **Proposition 5.5.** *The many-one reduction from* Monomial Code Equivalence *to* Tensor Isomorphism *from Grochow & Qiao [25] is in fact a (2, 4)-many-one reduction.*

Proof. We recall the reduction, then prove that it is a low-degree PC reduction. Let M be a $k \times n$ matrix. We build a 3-tensor of size $(k+2n) \times n \times (1+2n)$ as follows. The first frontal slice is $\begin{bmatrix} M \\ 0_{2n\times n} \end{bmatrix}$. The remaining $2n$ slices all have just a single nonzero entry, which serve to place a 2×2 identity matrix "behind and perpendicular" to M, one 2×2 matrix in each column. Let us index these slices by $[n] \times 2$. Then the (i,b) slice has a 1 in entry $(2(i-1)+b, i)$, for all $i \in [n], b \in [2]$. Let us call this tensor $r(M)$. Then the reduction maps M, M' to $r(M), r(M')$.

Let X, X', Y, Y', Z, Z' be the variable matrices for the TI equations for $r(M), r(M')$, and let A, B, A', B' be the variable matrices for Monomial Code Equivalence of M, M' (that is, $AMB^T = M'$, A is invertible, B is monomial and invertible). Consider the substitution:

$$X \mapsto A \oplus (B \otimes I_2) \qquad Y \mapsto B \qquad Z \mapsto 1 \oplus (B' \circ B') \otimes I_2$$

$$X' \mapsto A' \oplus (B' \otimes I_2) \qquad Y' \mapsto B' \qquad Z' \mapsto 1 \oplus (B \circ B) \otimes I_2.$$

As before, $B \circ B$ denotes the Hadamard or entry-wise product. Let us see what the TI equations become under this substitution. We get

$$AMB^T = M' \qquad AA' = A'A = \mathrm{Id}$$

$$BB' = B'B = \mathrm{Id} \qquad (B' \circ B')(B \circ B) = (B \circ B)(B' \circ B') = \mathrm{Id}$$

Indeed, notice that the effect of the $B \otimes I_2$ in X and the B in Y is that the row and column locations of the 2×2 matrix gadgets get permuted in the same way, and the gadget get multiplied by the *square* of the nonzero entries of B. These are then multiplied by the $B' \circ B'$ in Z.

Now, we derive these equations from the equations for Monomial Code Equivalence. The first three are already present in the equations for Monomial Code Equivalence. The last one we expand out, to see that we need to derive:

$$\sum_j b_{ij}^2 (b'_{jk})^2 = \delta_{ik} (\forall i, k)$$

Now, for $i \neq k$, we may take the equation $\sum_j b_{ij} b'_{jk}$ and square it, to derive

$$\sum_{j \neq j'} b_{ij} b'_{jk} + b_{ij'} b'_{j'k} + \sum_j b_{ij}^2 b_{j'k}^2.$$

Each term in the first sum is divisible by one of the monomial axioms $b_{ij} b_{ij'}$ since $j \neq j'$, and the second sum is what we wanted to derive.

Finally, for $i = k$, we square the equation $\sum_j b_{ij} b'_{ji} - 1$ and add to it $2\left(\sum_j b_{ij} b'_{ji} - 1\right)$. We then proceed to cancel terms with the monomial axioms as above, and end up with $\sum_j b_{ij}^2 (b'_{ji})^2 - 1$, as desired. ◀

6 Lower bound on PC degree for Tensor Isomorphism from Random 3XOR

We get a lower bound on PC refutations for Tensor Isomorphism through the following series of low-degree PC many-one reductions (Definition 2.3):

$$\begin{aligned}
\text{Random 3-XOR} &\leq_m^{PC} \{\pm 1\}\text{-Monomial Equivalence of} && (4)\\
&\quad \{\pm 1\}\text{-Multilinear Noncommutative Cubic Forms} && (5)\\
&\leq_m^{PC} \text{Monomial Equivalence of } \{\pm 1\} \text{ Noncommutative} && (6)\\
&\quad \text{Cubic Forms} && (7)\\
&\leq_m^{PC} \text{Equivalence of } \{\pm 1\} \text{ Noncommutative Cubic Forms} && (8)\\
&\leq_m^{PC} \text{Tensor Isomorphism} && (9)
\end{aligned}$$

We then appeal to the following PC lower bound on Random 3-XOR:

▶ **Theorem 6.1** (Ben-Sasson & Impagliazzo [5, Thm. 3.3 & Lem. 4.7]). *Let be any field of characteristic $\neq 2$. A random 3-XOR instance with clause density $\Delta = m/n$ requires PC degree $\Omega(n/\Delta^2)$ to refute, with probability $1 - o(1)$.*

This allows us to prove:

▶ **Theorem 6.2.** *Over any field of characteristic $\neq 2$, there is a random distribution of instances of $n \times n \times n$ Tensor Isomorphism – which assigns nonzero probability to at least $2^{\Omega(\sqrt[4]{n}) \log n}$ different instances – whose associated equations require PC degree $\Omega(\sqrt[4]{n})$ to refute, with probability $1 - o(1)$.*

Note that such instances have $N = 6n^2$ variables, so this is really only an $\Omega(\sqrt[8]{N})$ lower bound relative to the number of variables.

In the following subsections we recall the definitions of the above problems and their associated systems of polynomial equations, and we give the reductions in the order listed above.

The first two reductions are gadget constructions of linear size; the proof of correctness for the first uses the fact that random hypergraphs have no automorphisms, while the second is fairly straightforward. Reduction (8) uses a gadget from Grochow & Qiao [26], albeit for a

new application, and shows that the reduction using this gadget also yields a low-degree PC reduction. Reduction (9) is based on two lemmas, which show that the many-one reduction for this problem in fact also gives a low-degree PC reduction.

▶ Remark 6.3. Both of the latter two reductions have a quadratic size increase, so while we get a nearly-linear lower bound on PC degree for refutations of Monomial Equivalence of Noncommutative Cubic Forms, we only get a $\Omega(\sqrt{n})$ degree lower bound Equivalence of Noncommutative Cubic Forms and a $\Omega(\sqrt[4]{n})$ degree lower bound on Tensor Isomorphism. If the gadget sizes of these latter two reductions could be improved to linear, we would get a similarly near-linear lower bound (linear in the side length, still $\sqrt{N}$ relative to the number of variables) on PC refutations for Tensor Isomorphism as well. As many of the reductions in [19, 26] are of a similar flavor to the ones we consider here, we believe that they can all be proven in low-degree PC, so we expect the main obstacle to such an improvement is the size of the constructions themselves.

6.1 From Random 3-XOR to $\{\pm 1\}$-multilinear noncommutative cubic forms

▶ **Definition 6.4.** *A random 3-XOR instance with n variables and m clauses is obtained by sampling m clauses independently and uniformly from the set of all $2\binom{n}{3}$ parity constraints on 3 variables. Each parity constraint is encoded by an equation of the form $x_i x_j x_k = \pm 1$, and the Boolean constraints are encoded by $x_i^2 = 1$.*

By a $\{\pm 1\}$-monomial matrix, we mean a monomial matrix in which all nonzero entries are one of ± 1. $\{\pm 1\}$-Monomial Equivalence of Noncommutative Cubic Forms is the problem of deciding, given two noncommutative cubic forms f, f' in n variables $x_1, \ldots, x_n$ with all nonzero coefficients ± 1, whether there is a permutation $\pi \in S_n$ and signs $e_i \in \{\pm 1\}$ such that $f(e_1 x_{\pi(1)}, \ldots, e_2 x_{\pi(2)}, \ldots, e_n x_{\pi(n)}) = f'(\vec{x})$. Equivalently, if we represent a noncommutative cubic form f by the 3-way array T_{ijk} such that $f(\vec{y}) = \sum_{i,j,k \in [n]} T_{ijk} y_i y_j y_k$, the problem here asks whether there is a $\{\pm 1\}$-monomial matrix A such that $(A, A, A) \cdot T = T'$, that is, whether $T'_{i'j'k'} = \sum_{ijk} a_{ii'} a_{jj'} a_{kk'} T_{ijk}$ for all $i', j', k' \in [n]$.

▶ **Definition 6.5.** *We define the systems of equations associated to several variations of Equivalence of Noncommutative Cubic Forms.*

1. *Given two $n \times n \times n$ 3-way arrays T, T', the system of equations for Equivalence of Noncommutative Cubic Forms is the following system of equations in $2n^2$ variables. Let A, A' be $n \times n$ matrices of independent variables a_{ij}, a'_{ij}, respectively.*

$$\begin{aligned} &(A, A, A) \cdot T = T' && \text{(A is an equivalence)} \\ &AA' = A'A = \mathrm{Id} && \text{(A is invertible with } A^{-1} = A') \end{aligned}$$

2. *The system of equations for Monomial Equivalence of Noncommutative Cubic Forms includes the preceding equations, as well as:*

$$\begin{aligned} &a_{ij} a_{ij'} = 0 \quad \forall i \forall j \neq j' && \text{(at most one nonzero per row)} \\ &a_{ij} a_{i'j} = 0 \quad \forall j \forall i \neq i' && \text{(at most one nonzero per column)} \end{aligned}$$

3. *The system of equations for $\{\pm 1\}$-Monomial Equivalence of Noncommutative Cubic Forms includes all the preceding equations, as well as*

$$a_{ij}(a_{ij} + 1)(a_{ij} - 1) = 0 \quad \forall i, j \in [n] \quad \text{(all entries in } \{0, \pm 1\}\text{)}$$

4. *A noncommutative cubic form* $\sum_{ijk} T_{ijk} x_i x_j x_k$ *is* multilinear *if all nonzero terms* T_{ijk} *have* i, j, k *distinct (that is,* $|\{i, j, k\}| = 3$*). The system of equations for* $\{\pm 1\}$*-*Monomial Equivalence of Adjective Noncommutative Cubic Forms *is the same as the above, with the restriction that* T *and* T' *both satisfy* Adjective *(e. g., multilinear, nonzero entries in* $\{\pm 1\}$*, etc.).*

▶ **Theorem 6.6.** *There is a linear-size (1,3)-reduction from* Random 3-XOR *instances on* n *variables with* m *clauses, where* $10^4 n \leq m \leq \binom{n}{3}/10^{12}$*, to* $\{\pm 1\}$*-*Monomial Equivalence of $\{\pm 1\}$ Multilinear Noncommutative Cubic Forms*, over any ring* R *of characteristic* $\neq 2$*.*

The reduction is always a (1,3)-reduction, but we only show the resulting system of equations for $\{\pm 1\}$-Monomial Equivalence of Noncommutative Cubic Forms is unsatisfiable with probability $1 - o(1)$ when the 3-XOR instance is chosen randomly with the parameters specified in the theorem. (It is possible that it is always unsatisfiable when the input 3-XOR instance is, but our proof does not answer this question.)

Proof idea. We build multilinear noncommutative cubic forms from the 3-XOR instance such that they are equivalent by a $\{\pm 1\}$ diagonal matrix iff the 3-XOR instance is satisfiable: an equation $x_i x_j x_k = \pm 1$ corresponds to setting $T_{ijk} = 1, T'_{ijk} = \pm 1$ in this construction. The noncommutative cubic forms are multilinear because the construction of the random 3XOR instance ensures that each XOR clause contains 3 distinct variables. In fact, the equations for $\{\pm 1\}$-diagonal equivalence of the correspondence noncommutative cubic forms will turn out to be identically the same as the equations for the 3-XOR instance.

Next, for random instances chosen with the stated parameters, the 3-way arrays T, T' are the adjacency hyper-matrices of a 3-uniform hypergraph that has no nontrivial automorphisms by [40, Lemma 6.9]; this is why we needed to restrict the parameter range for m as we did. Because the hypergraphs have no nontrivial automorphisms, any monomial equivalence of the corresponding cubic forms must in fact be diagonal, thus letting us further reduce to $\{\pm 1\}$-monomial equivalence. ◀

Proof. We are given a system of 3-XOR equations, which we'll denote $x_{i_\ell} x_{j_\ell} x_{k_\ell} = s_\ell$ for $\ell = 1, \ldots, m$, where $i_\ell \leq j_\ell \leq k_\ell \in [n]$ are indices of variables and $s_\ell \in \{\pm 1\}$ for all ℓ. It also includes the equations $x_i^2 = 1$ for all $i = 1, \ldots, n$.

Step 1: Reduce from random 3-XOR to $\{\pm 1\}$-diagonal equivalence of noncommutative cubic forms. From the above system of equations, we now construct two $n \times n \times n$ 3-way arrays T, T'. For the original equations $x_{i_\ell} x_{j_\ell} x_{k_\ell} = s_\ell$ $(\ell = 1, \ldots, m)$, and for any $a_\ell \in \{\pm 1\}$ of our choice (we may set all $a_\ell = 1$ if we wish, but this additional flexibility may be useful in other settings) we set

$$T_{i_\ell, j_\ell, k_\ell} = a_\ell \text{ and } T'_{i_\ell, j_\ell, k_\ell} = s_\ell a_\ell.$$

All other entries of T and T' are set to zero.

We start with a warmup lemma, to see that this part of the construction already has a desirable property. By a "$\{\pm 1\}$ diagonal isomorphism" of two non-commutative cubic forms, we mean a diagonal matrix X whose diagonal entries are all one of ± 1 such that X gives an equivalence between T, T'.

▶ **Lemma 6.7.** *Notation as in the paragraph above. There is a bijection between the solutions to the 3-XOR instance and the* $\{\pm 1\}$ *diagonal isomorphisms of the noncommutative cubic forms defined by* T, T'*.*

Proof. Suppose $\mathbf{x}$ is a solution to the 3-XOR instance. Let $X = \mathrm{diag}(x_1, \ldots, x_n)$ be the diagonal matrix with $\mathbf{x}$ on the diagonal. We claim that X is an equivalence between the noncommutative cubic forms represented by T, T', or the same, that (X, X, X) is an isomorphism of the tensors T, T'. Note that for any diagonal matrices X, Y, Z, we have $((X, Y, Z) \cdot T)_{ijk} = x_i y_j z_k T_{ijk}$. In particular, the action of diagonal matrices does not change which entries of T are zero or nonzero, it merely scales the nonzero entries. Since T, T' have the same support by construction, it is necessary and sufficient to handle the nonzero entries. By the construction above, there are precisely m such nonzero entries, one for each cubic equation in the 3-XOR instance. For each $\ell = 1, \ldots, m$, we have

$$\begin{aligned}((X, X, X) \cdot T)_{i_\ell j_\ell k_\ell} &= x_{i_\ell} x_{j_\ell} x_{k_\ell} T_{i_\ell j_\ell k_\ell} \\ &= s_\ell T_{i_\ell j_\ell k_\ell} \\ &= T'_{i_\ell j_\ell k_\ell}.\end{aligned}$$

In the other direction, if $X = \mathrm{diag}(\mathbf{x})$ is a diagonal matrix whose diagonal entries are in $\{\pm 1\}$ giving an isomorphism of the noncommutative cubic forms, then we have

$$x_{i_\ell} x_{j_\ell} x_{k_\ell} = T_{i_\ell j_\ell k_\ell} T'_{i_\ell j_\ell k_\ell} = s_\ell$$

for $\ell = 1, \ldots, m$. (Here we have pulled $T_{i_\ell, j_\ell, k_\ell}$ across the equals sign because every term in the above equation is ± 1.) This concludes the proof of the lemma. ◀

We thus consider the equations corresponding to $\{\pm 1\}$-diagonal equivalence of T, T': there are n variables x_i $(i = 1, \ldots, n)$. Let X denote the diagonal matrix with $\mathbf{x}$ on the diagonal. Then the equations are

$$X^2 = \mathrm{Id} \qquad (X, X, X) \cdot T = T'. \tag{10}$$

By Lemma 6.7, we have that the original 3XOR instance is satisfiable iff (10) is satisfiable. We claim furthermore that there is (1,3)-reduction from the 3XOR equations to this system of equations. In fact, as the proof of the preceding lemma shows, they are actually *the same set of equations*! So there is nothing more to show.

Step 2: Reduce from $\{\pm 1\}$-diagonal equivalence to $\{\pm 1\}$-monomial equivalence. We claim that there is a $(1, 3)$-reduction from (10) to the the equations for $\{\pm 1\}$-monomial equivalence, see (6.5). The variable substitution is given by

$$a_{ij} = a'_{ij} \mapsto \begin{cases} 0 & i \neq j \\ x_i & i = j. \end{cases}$$

Under this substitution:

- The equivalence condition $(A, A, A) \cdot T = T'$ becomes exactly the original equivalence condition $(X, X, X) \cdot T = T'$.
- The invertibility equations $AA' = A'A = \mathrm{Id}$ become $XX = \mathrm{Id}$
- The row and column equations both become $0 = 0$, since at least one of the two a_{ij} variables occurring will not be on the diagonal, hence will become 0 after substitution.
- The equation $a_{ij}(a_{ij} + 1)(a_{ij} - 1) = 0$ becomes $x(x^2 - 1)$ for the appropriate variable $x \in \mathbf{x}$. This is derivable from the original equation $x^2 - 1$ by multiplication by x.

Lastly, we show that the system of equations in Definition 6.5(3) for $\{\pm 1\}$-monomial equivalence is satisfiable iff the original 3-XOR instance was. Since we showed above that that $\{\pm 1\}$-diagonal equivalence equations are satisfiable iff the original 3-XOR instance was, we show the equisolvability of (10) and the equations of Definition 6.5(3).

Since diagonal matrices are monomial, any solution to (10) is a solution to the equations of Definition 6.5(3).

Conversely, suppose the equations of Definition 6.5(3) are solvable. Then there is a $\{\pm 1\}$-monomial matrix X given an equivalence between T and T'; we may write $X = DP$ where D is diagonal and P is a permutation matrix. Now, as the original 3-XOR instance was chosen uniformly at random, the support of T (the positions of its nonzero entries) is precisely a uniformly random 3-uniform hypergraph G. As T, T' have the same support by construction, we find that P must be an automorphism of G. But by [40, Lemma 6.9], uniformly random such hypergraphs have no nontrivial automorphisms with probability $1 - o(1)$. Thus $P = I$ and X must in fact be diagonal, hence a solution to (10). ◀

▶ Remark 6.8. We may avoid the heavy hammer of [40, Lemma 6.9] by "rigidifying" (in the sense of removing automorphisms) the system of 3-XOR equations before constructing the 3-way arrays as follows. The construction corresponds to a standard graph-theoretic gadget for removing automorphisms. Add new variables z and y_{ij} for $i = 1, \ldots, n$ and $j = 1, \ldots, n + i$, as well as the equations $x_i y_{ij} z = 1$ for all i, j, as well as $y_{ij}^2 = 1$ and $z^2 = 1$. The downside of this construction is that it quadratically increases the number of variables, which would result in a further quadratic loss in our lower bounds on Tensor Isomorphism.

6.2 From $\{\pm 1\}$-monomial equivalence to (unrestricted) monomial equivalence

▶ **Theorem 6.9.** *There is a linear-size (2, 6)-many-one reduction from*

$\{\pm 1\}$-*Monomial Equivalence of* $\{\pm 1\}$ *Multilinear Noncommutative Cubic Forms*

to

Monomial Equivalence of $\{\pm 1\}$ *Noncommutative Cubic Forms,*

over any ring R of characteristic $\neq 2$ such that $\{\pm 1\}$ are the only square roots of 1.

Furthermore, the reduction r has the property that, given any two $\{\pm 1\}$ multilinear noncommutative cubic forms f, f', any monomial equivalence between $r(f)$ and $r(f')$ must have all its nonzero entries sixth roots of unity, and this can be derived by a degree-6 PC proof.

▶ Remark 6.10. We note the difference between a reduction to $\sqrt[6]{1}$-Monomial Equivalence and a reduction to Monomial Equivalence with the property stated in the theorem. In the former case, the problem being reduced to only accepts $\sqrt[6]{1}$-monomial matrices as solutions (and then the goal of the reduction is to introduce gadgets to get this down to $\{\pm 1\}$). In the latter case, the problem being reduced to allows arbitrary monomial matrices as solutions, but the gadgets enforce that, on the reduced instances, any such monomial matrix must in fact have its nonzero entries being sixth roots of unity.

Proof. Let T be an $n \times n \times n$ 3-way array representing a multilinear noncommutative cubic form with all nonzero entries in ± 1. We extend T to $r(T)$ of size $2n \times 2n \times 2n$, by setting

$$\begin{aligned} r(T)_{ijk} &= T_{ijk} && i, j, k \in [n] \\ r(T)_{i,i,n+i} &= 1 && i \in [n] \\ r(T)_{n+i,n+i,n+i} &= 1 && i \in [n] \end{aligned}$$

and all other entries of $r(T)$ set to zero.

Many-one reduction. We first show that the map $(T, T') \mapsto (r(T), r(T'))$ is a many-one reduction. Suppose T, T' are $\{\pm 1\}$-monomially equivalent by a matrix X, where $X = DP$ with $D = \mathrm{diag}(x_1, \ldots, x_n)$ a diagonal matrix with $x_i \in \{\pm 1\}$ for all i, and P is a permutation matrix. Let π denote the permutation corresponding to P; that is, $P_{i,\pi(i)} = 1$ for all $i \in [n]$. Then we claim the $2n \times 2n$ matrix $X \oplus P = \begin{bmatrix} X & 0 \\ 0 & P \end{bmatrix}$ is a monomial equivalence of $r(T)$ with $r(T')$. Since $X \oplus P$ is block-diagonal, the upper-left X certainly sends the upper-left $n \times n \times n$ sub-array of $r(T)$ (which is just T) to that of $r(T')$ (which is just T'). So the only thing to check is what happens to the positions at indices greater than n.

Let $X' = X \oplus P$. We have

$$\begin{aligned}((X', X', X') \cdot r(T))_{i,i,n+i} &= r(T)_{\pi(i),\pi(i),n+\pi(i)} (X'_{i,\pi(i)})^2 X'_{n+i,n+\pi(i)} \\ &= r(T)_{\pi(i),\pi(i),n+\pi(i)} (X_{i,\pi(i)})^2 P_{i,\pi(i)} \\ &= 1 = r(T')_{i,i,n+i}.\end{aligned}$$

Similarly, we have:

$$((X', X', X') \cdot r(T))_{n+i,n+i,n+i} = r(T)_{n+\pi(i),n+\pi(i),n+\pi(i)} P^3_{i,\pi(i)} = 1 = r(T')_{n+i,n+i,n+i}$$

Because X' is monomial, it is easy to see that the zeros of $r(T)$ are sent to zeros of $r(T')$. Thus X' is a monomial equivalence of $r(T)$ with $r(T')$.

Conversely, suppose $r(T)$ and $r(T')$ are equivalent by a monomial matrix $Y = DP$, with D diagonal and P a permutation matrix corresponding to permutation $\pi \in S_{2n}$. We will show that this implies that T and T' are equivalent by a $\{\pm 1\}$ monomial matrix. Since T is multilinear, we have $T_{i,i,i} = r(T)_{i,i,i} = 0$. Since $r(T)_{n+j,n+j,n+j} = 1$ for all $j \in [n]$, the permutation π cannot send any element $> n$ to any element $\leq n$. Thus P is block-diagonal, say $P = \begin{bmatrix} P_1 & 0_n \\ 0_n & P_2 \end{bmatrix}$. Let π_1 (resp., π_2) be the permutation of $[n]$ corresponding to P_1 (resp., P_2).

Next, we claim $P_1 = P_2$. By considering the positions at indices $(i, i, n+i)$, we have:

$$((P, P, P) \cdot r(T))_{i,i,n+i} = r(T)_{\pi_1(i),\pi_1(i),n+\pi_2(i)}$$

But the latter is equal to the corresponding position in $r(T')$, which is 1 iff $\pi_1(i) = \pi_2(i)$. Since this holds for all i, we have $\pi_1 = \pi_2$, and thus $P_1 = P_2$.

Finally, we *do not* claim that the diagonal entries y_i themselves must be in ± 1. Rather, we will show that they are all sixth roots of unity. Then cubing them will yield a new $n \times n$ matrix D' all of whose diagonal entries are ± 1 such that $D'P_1$ is a ± 1-monomial equivalence of T with T'.

From the positions $(n+i, n+i, n+i)$, we have

$$\begin{aligned}1 &= r(T')_{n+\pi_1(i),n+\pi_1(i),n+\pi_1(i)} \\ &= ((Y, Y, Y) \cdot r(T))_{n+i,n+i,n+i} \\ &= y^3_{n+i}.\end{aligned}$$

But then, considering the positions $(i, i, n+i)$, we similarly get that $y_i^2 y_{n+i} = 1$. Cubing the latter equation, we get $y_i^6 y_{n+i}^3 = 1$. But as we already have $y_{n+i}^3 = 1$, this gives us $y_i^6 = 1$ by a degree-6 PC proof, as claimed in the "furthermore."

Now we use the fact that T, T' have all entries in $\{0, \pm 1\}$. Thus, each nonzero entry of $r(T)$ in the front-upper-left block (corresponding to T) gives us an equation of the form $y_i y_j y_k T_{ijk} = T'_{\pi_1(i),\pi_1(j),\pi_1(k)}$. Since the nonzero entries of T, T' are ± 1, this is thus an

equation of the form $y_i y_j y_k = \pm 1$. If we cube both sides of this equation, we get $y_i^3 y_j^3 y_k^3 = \pm 1$. But since we established above that $y_i^6 = 1$ for all i, we have that $y_i^3 \in \{\pm 1\}$ for all i. Thus, defining $x_i := y_i^3$ for $i = 1, \ldots, n$, we have $x_i \in \{\pm 1\}$ and letting $D' = \mathrm{diag}(x_1, \ldots, x_n)$, we have $D'P_1$ is a $\{\pm 1\}$-monomial equivalence from T to T'.

Low-degree PC reduction. We claim that the system of equations for $\{\pm 1\}$ monomial equivalence of T and T' is (2,6)-reducible to the system of equations for monomial equivalence of $r(T)$ and $r(T')$. Let X, X' be the $n \times n$ variable matrices for the equations for for $\{\pm 1\}$-monomial equivalence of the original tensors T and T', and let Y, Y' be the $2n \times 2n$ matrices for the equations for monomial equivalence of $r(T), r(T')$. The PC reduction is defined by the following substitution:

$$\begin{aligned} y_{ij} &\mapsto x_{ij} && i,j \in [n] \\ y_{n+i,n+j} &\mapsto x_{ij}^2 && i,j \in [n] \\ y_{i,n+j}, y_{n+i,j} &\mapsto 0 && i,j \in [n], \end{aligned}$$

and similarly for the y' variables being substituted by the x' variables. That is, we have

$$Y \mapsto \begin{bmatrix} X & 0_n \\ 0_n & X \circ X \end{bmatrix} \qquad Y' \mapsto \begin{bmatrix} X' & 0_n \\ 0_n & X' \circ X' \end{bmatrix},$$

where $X \circ X$ denotes the entrywise (aka Hadamard) product with itself, that is $(X \circ X)_{ij} = x_{ij}^2$. The reason to use $X \circ X$ here is that if X is $\{\pm 1\}$-valued and monomial, then $X \circ X$ is the permutation matrix with the same support as X; that is, this substitution is essentially the same as the one used in the proof above for the many-one reduction.

Now, taking advantage of the block structure in the substitution above and the block structure in $r(T), r(T')$, let us see what our equations become after substitution, and how to derive them from the equations for T, T'. This will complete the proof.

1. The set of equations $(Y, Y, Y) \cdot r(T) = r(T')$ becomes the set of equations $(X, X, X) \cdot T = T'$ (by examining the front-upper-left corner), as well as the equations

$$\sum_{i,j,k \in [2n]} y_{ii'} y_{jj'} y_{k,k'} r(T)_{ijk} = \begin{cases} 1 & i' = j' = k' - n \text{ or } i' = j' = k' > n \\ 0 & \text{otherwise.} \end{cases}$$

We deal with the three cases ($i' = j' = k' - n$, $i' = j' = k' > n$, or neither of these) separately.

a. Suppose $i' = j' = k' - n$. In this case, $y_{ii'}$ is only nonzero for $i \in [n]$, and similarly for $y_{jj'}$, while $y_{kk'}$ is only nonzero for $k > n$. Thus the substituted equation becomes

$$\sum_{i,j,k \in [n]} y_{ii'} y_{ji'} y_{n+k,n+i'} r(T)_{i,j,n+k} = \sum_{i,j,k \in [n]} x_{ii'} x_{ji'} x_{k,i'}^2 r(T)_{i,j,n+k} = 1$$

Now, the only positions in $r(T)$ of the form $(i, j, n+k)$ with $i, j, k \in [n]$ that are nonzero are those of the form $(i, i, n+i)$, so the preceding equation simplifies further to

$$\sum_{i \in [n]} x_{ii'} x_{ii'} x_{ii'}^2 = 1$$

i.e.,

$$\sum_{i \in [n]} x_{ii'}^4 = 1. \tag{11}$$

We will now show how to derive (11) from the equations for $\{\pm 1\}$-monomial equivalence of for T, T' (Definition 6.5). From the $\{0, \pm 1\}$ equation in Definition 6.5(3), if we multiply by $x_{ii'}$, we get

$$x_{ii'}^2(x_{ii'}^2 - 1), \tag{12}$$

i.e., the usual Boolean equation but for $x_{ii'}^2$ rather than $x_{ii'}$ itself. Next, from $x_{ii'}x_{i''i'}$ with $i \neq i''$, we may square this to get

$$x_{ii'}^2 x_{i''i'}^2. \tag{13}$$

and we similarly get $(x'_{i'i})^2(x'_{i'i''})^2$ when $i \neq i''$.
Lastly, from the equation $XX' = \mathrm{Id}$ and multiplying by $\sum_{i\in[n]} x_{ii'}x'_{i'i} + 1$, we obtain

$$(\sum_{i\in[n]} x_{ii'}x'_{i'i}+1)(\sum_{i\in[n]} x_{ii'}x'_{i'i}-1) = \sum_{i\in[n]} x_{ii'}^2 x_{i'i}^2 + \sum_{i,j\in[n]i\neq j} x_{ii'}x'_{i'i}x_{ji'}x'_{i'j} - 1 = \sum_{i\in[n]} x_{ii'}^2 x_{i'i}^2 - 1, \tag{14}$$

where we observed that from the axioms that $x_{ii'}x_{ji'} = 0$ for $i \neq j$ we may derive in degree 4 that the middle term $\sum_{i,j\in[n]i\neq j} x_{ii'}x_{ji'}x'_{i'i}x'_{i'j} = 0$.
Now, equations (12)–(14) are a degree-2 substitution instance of the equations in Lemma 6.11 with $c = 2, d = 1$. Thus, by Lemma 6.11, we can derive (11) from these in degree 6.

b. Suppose $i' = j' = k' > n$. In this case, the substitution makes all of $y_{ii'}, y_{jj'}, y_{kk'}$ equal to zero unless $i, j, k > n$. Thus we may write the equation, after substitution, as

$$\begin{aligned}&\sum_{i,j,k\in[n]} y_{n+i,i'}y_{n+j,i}y_{n+k,i}r(T)_{n+i,n+j,n+k}\\ =&\sum_{i,j,k\in[n]} x_{i,i'-n}^2 x_{j,i'-n}^2 x_{k,i'-n}^2 r(T)_{n+i,n+j,n+k}\\ =&r(T')_{i',i',i'} = 1.\end{aligned}$$

However, because the only entries $r(T)_{n+i,n+j,n+k}$ that are nonzero are those in which $i = j = k$, this simplifies further to:

$$\sum_{i\in[n]} x_{i,i'-n}^6 = 1.$$

This is a degree-2 substitution instance of Lemma 6.11 with $c = 3, d = 1$, so it can be derived in degree 6 from the equations derived in part (a).

c. Suppose neither of the previous two cases hold. The derivation will depend on which of i', j', k' lie in $[n]$ versus $\{n+1, \ldots, 2n\}$.

i. When all are in $[n]$, we are in the front-upper-left corner of the tensor, and we exactly get the equations $(X, X, X) \cdot T = T'$.

ii. When all three of i', j', k' are $> n$, the only nonzero entries of $r(T)$ are of the form $r(T)_{n+i,n+i,n+i}$, so the equation becomes

$$\sum_{i\in[n]} x_{i,i'-n}^2 x_{i,j'-n}^2 x_{i,k'-n}^2 = 0.$$

Since we have assumed $|\{i', j', k'\}| > 1$, there are at least two distinct indices among them, and thus each term in this sum is a multiple of one of our $x_{ij}x_{ij'}$ axioms with $j \neq j'$.

iii. Next, suppose instead that $i', j' \in [n], k' > n$. In this case, the only nonzero entries of Y after substitution are those with $i, j \in [n], k > n$. Thus the equation becomes

$$\sum_{i,j,k\in[n]} x_{ii'}x_{jj'}x^2_{k,k'-n}r(T)_{i,j,n+k} = 0$$

However, the only nonzero entries of $r(T)$ in which the first two coordinates are $\leq n$ and the third is $n+k$ are those of the form $i = j = k$, so the preceding becomes

$$\sum_{i\in[n]} x_{ii'}x_{ij'}x^2_{ik'-n} = 0.$$

Since we do not have $i' = j' = k' - n$ (as that was covered in a previous case), at least two of the column indices differ, and thus each term of this sum is divisible by one of the axioms of the form $x_{ij}x_{ij'}$ with $j \neq j'$.

iv. In all other cases, the corresponding entries of $r(T)$ are all zero, so the equation reduces to $0 = 0$.

2. The equations $YY' = Y'Y = \mathrm{Id}$ become $XX' = X'X = \mathrm{Id}$ and $(X \circ X)(X' \circ X') = (X' \circ X')(X \circ X) = \mathrm{Id}$. The first of these is one of our original equations, so it remains to derive the second. We show how to derive $(X \circ X)(X' \circ X') = \mathrm{Id}$; the other is similar. For clarity, let us write it out using indices:

$$\sum_j x^2_{ij}(x'_{jk})^2 \quad - \quad \delta_{ik} = 0 \qquad \forall i, k \in [n] \tag{15}$$

Starting from the equation $\sum_j x_{ij}x'_{jk} - \delta_{ik} = 0$, we multiply by $\sum_j x_{ij}x'_{jk}$, to get

$$\sum_j x^2_{ij}(x'_{jk})^2 + \sum_{j\neq j'} x_{ij}x'_{jk}x_{ij'}x_{j'k} - \delta_{ik}\sum_j x_{ij}x'_{jk}.$$

Note that every term in the middle summation here is divisible by some $x_{ij}x_{ij'}$ with $j \neq j'$, which is one of our equations, so we may cancel off those terms using those equations in degree 4. If $i \neq k$, then we are done. If $i = k$, then we add in our equation $\sum_j x_{ij}x'_{jk} - 1$ to get (15).

3. The equations $y_{ij}y_{ij'} = 0$ for $j \neq j'$ become 0 after substitution unless i, j, j' are either all in $[n]$ or all in $\{n+1, \ldots, 2n\}$. In the former case, the substituted equation is $x_{ij}x_{ij'} = 0$, which is already one of the original equations. In the latter case, the equation becomes $x^2_{ij}x^2_{ij'} = 0$; but this is easily derivable from $x_{ij}x_{ij'}$ by multiplying it by itself (degree 4). The equations saying there is at most one entry per column of Y are derived from those for X similarly.

This covers all the equations for monomial equivalence of $r(T), r(T')$, and thus we are done. ◀

▶ **Lemma 6.11.** *For any integers $d \geq 1, c \geq 1$, from the equations*

$$x_i(x_i^d - 1)(\forall i) \qquad x_ix_j(\forall i \neq j) \qquad \sum_{i=1}^n x_iy_i - 1$$

there is a degree-$\max\{d+2, cd\}$ PC derivation (over any ring R) of

$$\sum_{i\in[n]} x_i^{cd} - 1$$

Although in the proof above we only used the $d = 1$ and $c = 2, 3$, we will later have occasion to use this lemma with larger values of d and c, which is why we phrase it in this level of generality.

Proof. First we show it for $c = 1$, then derive the general case from that.

Let $S = \sum_{i\in[n]} x_i^d$, $D = \sum_{i\in[n]} x_i y_i$. Our first goal is to derive $S-1$. For each $i = 1, \ldots, n$, we can derive $x_i y_i(S-1)$ in degree $d+2$ as follows:

$$\begin{aligned} x_i y_i(S-1) &= x_i^{d+1} y_i + y_i \sum_{j\neq i} x_i x_j^d - x_i y_i \\ &= y_i(x_i^{d+1} - x_i) + y_i \sum_{j\neq i} x_i x_j^d = \underline{x_i(x_i^d - 1)} y_i + y_i \sum_{j\neq i} \underline{x_i x_j} x_j^{d-1}, \end{aligned}$$

where we have underlined the use of the axioms.

Summing up the preceding for all i, we derive $DS - D$ in degree $d+2$. Finally, we multiply the starting equation $D-1$ by S to get $SD - S$, also in degree $d+2$. Then we have

$$(DS - D) - (SD - S) + (D - 1) = S - 1 = \sum_i x_i^d - 1,$$

as desired.

For $c > 1$, we then sum the preceding with $\sum_{i\in[n]}(x_i^{(c-1)d-1} + x_i^{(c-2)d-1} + \cdots + x_i^{d-1})\underline{(x_i^{d+1} - x_i)} = \sum_{i\in[n]} x_i^{cd} - x_i^d$, which has degree cd. ◀

6.3 From monomial equivalence to general equivalence of noncommutative cubic forms

▶ **Theorem 6.12.** *There is a quadratic-size many-one reduction from*

Monomial Equivalence of Noncommutative Cubic Forms

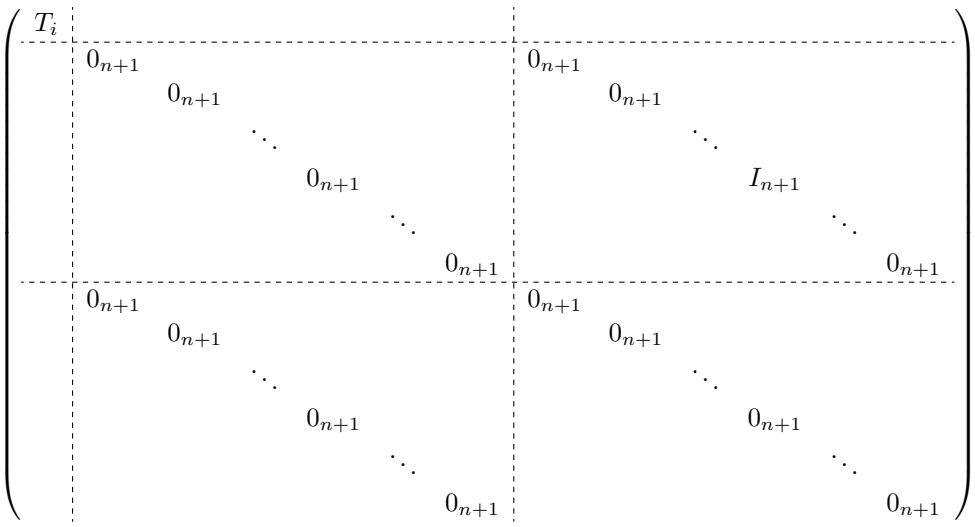
to

Equivalence of Noncommutative Cubic Forms,

over any field.

If furthermore the input cubic forms f, f' have the property that any monomial equivalence between them must have its nonzero scalars being d-th roots of unity, and the latter can be derived by PC in degree $d+1$, then the reduction above is a $(d, 2d)$-many-one reduction.

Proof. Let f be a noncommutative cubic form in variables $u_1, \ldots, u_n$. Then $r(f)$ will be a new noncommutative cubic form, in $n + 2n(n+1)$ variables $u_1, \ldots, u_n$, $v_{11}, v_{12}, \ldots, v_{n,n+1}$, $w_{11}, w_{12}, \ldots, w_{n,n+1}$, which is $r(f) = f + \sum_{i\in[n],j\in[n+1]} u_i v_{ij} w_{ij}$. In terms of the underlying three-way arrays, if we have $f = \sum_{i,j,k\in[n]} T_{ijk} u_i u_j u_k$, then we use $r(T)$ to denote the array underlying $r(f)$, which can be described as follows. The 3-way array $r(T)$ will have size $N \times N \times N$ where $N = n + 2n(n+1)$. Let T_i denote the i-th frontal slice of T_i, that is, T_i is the matrix such that $(T_i)_{jk} = T_{ijk}$. For $i = 1, \ldots, n$, the i-th frontal slice of $r(T)$ will be defined as:



where the I_{n+1} occurs in the i-th $(n+1) \times (n+1)$ block of its region. That is, the lower-right $2n(n+1) \times 2n(n+1)$ sub-matrix is the Kronecker product $E_{i,n+i} \otimes I_{n+1}$, where $E_{i,n+i}$ is the $2n \times 2n$ matrix with a 1 in position $(i, n+i)$ and zeros everywhere else. For the slices $i = n+1, \ldots, N$ we will have $r(T)_i = 0$.

Our main claim is that the map $(T, T') \mapsto (r(T), r(T)')$ is the reduction claimed in the theorem.

Many-one reduction. Suppose $X \cdot f = f'$ with X monomial. Write $X = PD$ with D diagonal and P a permutation matrix corresponding to the permutation $\pi \in S_n$. Then we claim that

$$Y = X \oplus ((PD^{-1}) \otimes I_{n+1}) \oplus (P \otimes I_{n+1})$$

is an equivalence between $r(f)$ and $r(f')$, where here we assume our variables are ordered as above. For we have

$$\begin{aligned}
Y \cdot r(f) &= \sum_{ijk \in [n]} T_{ijk} (Yu_i)(Yu_j)(Yu_k) + \sum_{i \in [n], j \in [n+1]} (Yu_i)(Yv_{ij})(Yw_{ij}) \\
&= \sum_{ijk \in [n]} T_{ijk} (Xu_i)(Xu_j)(Xu_k) + \sum_{i \in [n], j \in [n+1]} (Xu_i)(PD^{-1}v_{ij})(Pw_{ij}) \\
&= X \cdot f + \sum_{i \in [n], j \in [n+1]} D_{ii} u_{\pi(i)} (D_{ii}^{-1} v_{\pi(i),j}) w_{\pi(i),j} \\
&= f' + \sum_{i \in [n], j \in [n+1]} u_{\pi(i)} v_{\pi(i),j} w_{\pi(i),j} \\
&= r(f').
\end{aligned}$$

The final inequality here follows from the fact that π is a permutation, so the final sum includes all terms of the form $u_i v_{ij} w_{ij}$, just listed in a different order than originally.

Conversely, suppose $Y \cdot r(f) = r(f')$ for an arbitrary invertible $N \times N$ matrix Y. To find an equivalence between f and f', here we find it more useful to take the viewpoint of the 3-way arrays $r(T)$ and $r(T')$ corresponding to $r(f)$ and $r(f')$, respectively.

The way Y acts on the 3-way array $r(T)$ is to first take linear combinations of the frontal slices, say by replacing the i-th slice with $\sum_{j\in[N]} Y_{ij} r(T)_j$ (corresponding to the action of Y on the third variable in each monomial), and then to take each slice S and replace it by YSY^t (the left multiplication corresponds to the action on the first variable in each monomial, and the right multiplication corresponds to the action on the second variable in each monomial). As this latter transformation preserves the rank of each slice, we will use the ranks of linear combinations of the slices to reason about properties of Y.

▷ Claim 1. Y is a block-diagonal sum of an $n \times n$ matrix X and a $2n(n+1) \times 2n(n+1)$ matrix.

Proof of Claim 1. First we show that Y is block-triangular. To see this, note that since the last $2n(n+1)$ slices are zero, the action of Y by taking linear combinations of slices cannot send any of the first n slices to the last $2n(n+1)$ slices. That is, Y has the form $Y = \begin{bmatrix} X & Z \\ 0 & W \end{bmatrix}$ where X is $n \times n$ and W is $2n(n+1) \times 2n(n+1)$. It remains to show that Z must be zero.

Since Y is block-diagonal and invertible, we have that X and W are each invertible.

Let R be the tensor gotten from $r(T)$ by having Y act by taking linear combinations of the slices. That is, the i-th frontal slices of R is $R_i = \sum_{j\in[N]} Y_{ij} r(T)_j$. Since each slice $r(T)_i$ has its support in the upper-left $n \times n$ sub-matrix and the middle-right $n(n+1) \times n(n+1)$ sub-matrix, so does each slice R_i. Write

$$R_i = \begin{bmatrix} R_i^{(1,1)} & 0 & 0 \\ 0 & 0 & R_i^{(2,2)} \\ 0 & 0_{n(n+1)} & 0 \end{bmatrix},$$

where $R_i^{(1,1)}$ is $n \times n$ and $R_i^{(2,2)}$ is $n(n+1) \times n(n+1)$.

Now consider the action of Y that sends R_i to $YR_iY^t = r(T')_i$. We now break up Y further into blocks commensurate with how we wrote R_i above; write

$$Y = \begin{bmatrix} X & A & B \\ 0 & C & D \\ 0 & E & F \end{bmatrix} \qquad Z = \begin{bmatrix} A & B \end{bmatrix} \qquad W = \begin{bmatrix} C & D \\ E & F \end{bmatrix},$$

where A, B are $n \times n(n+1)$, and C, D, E, F are each $n(n+1) \times n(n+1)$. Then we have:

$$\begin{aligned} YR_iY^t &= \begin{bmatrix} X & A & B \\ 0 & C & D \\ 0 & E & F \end{bmatrix} \begin{bmatrix} R_i^{(1,1)} & 0 & 0 \\ 0 & 0 & R_i^{(2,2)} \\ 0 & 0_{n(n+1)} & 0 \end{bmatrix} \begin{bmatrix} X^t & 0 & 0 \\ A^t & C^t & E^t \\ B^t & D^t & F^t \end{bmatrix} \\ &= \begin{bmatrix} XR_i^{(1,1)} & 0 & AR_i^{(2,2)} \\ 0 & 0 & CR_i^{(2,2)} \\ 0 & 0 & ER_i^{(2,2)} \end{bmatrix} \begin{bmatrix} X^t & 0 & 0 \\ A^t & C^t & E^t \\ B^t & D^t & F^t \end{bmatrix} \\ &= \begin{bmatrix} XR_i^{(1,1)}X^t + AR_i^{(2,2)}B^t & AR_i^{(2,2)}D^t & AR_i^{(2,2)}F^t \\ CR_i^{(2,2)}B^t & * & * \\ ER_i^{(2,2)}B^t & * & * \end{bmatrix}, \end{aligned}$$

where we have put $*$'s in positions we won't need in the argument.

Next, since each of the first n slices of $r(T')$ must be of this form, and those slices have zeros in each block except the $(1,1)$ and $(2,3)$ blocks, by considering the blocks $(1,2), (1,3), (2,1), (3,1)$ we must have

$$AR_i^{(2,2)}D^t = 0 \qquad AR_i^{(2,2)}F^t = 0 \qquad CR_i^{(2,2)}B^t = 0 \qquad ER_i^{(2,2)}B^t = 0.$$

For reasons that will become clear below, we combine these into the two equations

$$AR_i^{(2,2)} \begin{bmatrix} D^t & F^t \end{bmatrix} = 0 \qquad \begin{bmatrix} C \\ E \end{bmatrix} R_i^{(2,2)} B^t = 0.$$

Note that the $n(n+1) \times 2n(n+1)$ matrices $\begin{bmatrix} D^t & F^t \end{bmatrix}$ and $\begin{bmatrix} C^t & E^t \end{bmatrix}$ must both be full rank, since otherwise $W = \begin{bmatrix} C & D \\ E & F \end{bmatrix}$ would not be invertible.

The sum of the (2,3) blocks (of size $n(n+1) \times n(n+1)$) of the first n slices of $r(T)$ is precisely the identity matrix $I_{n(n+1)}$. Thus, the linear span of these blocks contains an invertible matrix in it. Since Y is invertible, that linear span is the same as the linear span of the blocks $\{R_i^{(2,2)} : i \in [n]\}$. Thus the latter contains a full-rank matrix, say $\sum_{i=1}^n \alpha_i R_i^{(2,2)}$. But since we have $AR_i^{(2,2)} \begin{bmatrix} D^t & F^t \end{bmatrix} = 0$ for all i, we may left multiply by A and right-multiply by $\begin{bmatrix} D^t & F^t \end{bmatrix}$ to get $A \left(\sum_{i=1}^n \alpha_i R_i^{(2,2} \right) \begin{bmatrix} D^t & F^t \end{bmatrix} = \sum_{i=1}^n \alpha_i A R_i^{(2,2)} \begin{bmatrix} D^t & F^t \end{bmatrix} = 0$. But now we have that $\sum \alpha_i R_i^{(2,2)}$ is invertible, and $\begin{bmatrix} D^t & F^t \end{bmatrix}$ has full rank $n(n+1)$, so their product also has full rank $n(n+1)$. But then we have that A times a full rank matrix is equal to 0, hence A must be zero. The same argument, *mutatis mutandis*, using the equation $\begin{bmatrix} C \\ E \end{bmatrix} R_i^{(2,2)} B^t = 0$, gives us that $B = 0$. Hence Y is block-diagonal as claimed. ◁

Next, we use properties of the ranks of the slices coming from the I_{n+1} gadgets to show that X must in fact be monomial.

▷ Claim 2. $Y = \begin{bmatrix} X & 0 \\ 0 & W \end{bmatrix}$ where X is monomial.

Proof. In both $r(T)$ and $r(T')$, any linear combination consisting of k of the first n slices (with nonzero coefficients) has rank in the range $[k(n+1), k(n+1)+n]$, for any $k = 0, \ldots, n$. The lower bound can be seen by noting that any such linear combination is block-diagonal with k copies of I_{n+1} on the block diagonal of the $(2,3)$ block. The upper bound comes from the fact that these are the only nonzero blocks in the lower-right $2n(n+1) \times 2n(n+1)$ sub-matrix, and the only other nonzero entries are in the $n \times n$ upper-left sub-matrix, which has rank at most n because of its size.

Using notation from the proof of the preceding claim, since $YR_iY^t = r(T')_i$, and the latter has rank in the range $[n+1, 2n+1]$, R_i must also have rank in the same range. But this is only possible if R_i is a linear combination of precisely one of the first n slices of $r(T)$. Thus, X is monomial. ◁

From claim 2, we thus have that there is a permutation $\pi \in S_n$ and nonzero scalars $d_1, \ldots, d_n$ such that $R_i = d_i r(T)_{\pi(i)}$ for all $i = 1, \ldots, n$, where $X = DP$ with D the diagonal matrix with diagonal entries d_i and P the permutation matrix corresponding to π. Finally, in the proof of claim 1, we saw that the upper-left block of YR_iY^t was $XR_i^{(1,1)}X^t + AR_i^{(2,2)}B^t$, and then learned that $A = B = 0$. Putting these together, and recalling that the upper-left block of $r(T)_i$ is T_i, we thus get

$$(DP) d_i T_{\pi(i)} (DP)^t = T'_i$$

for all i. In other words, X is a monomial equivalence from T to T' (hence, from f to f'). This completes the proof that the construction gives a many-one reduction.

Low-degree PC reduction. To prove the "furthermore", suppose that the pair of cubic forms f, f' has the property that any monomial equivalence between them must have its nonzero entries being d-th roots of unity, for some $d \geq 1$, and that this can be derived – more specifically, the equations $y_{ij}^{d+1} - y_{ij}$ and similarly for y'_{ij} – in degree $d+1$.

Let Y, Y' be the variable matrices for (general) equivalence of $r(f), r(f')$; let X, X' be the variable matrices for monomial equivalence of f, f'. Consider the substitution

$$Y \mapsto \begin{bmatrix} X & 0 & \\ 0 & X^{\circ(d-1)} \otimes I_{n+1} & \\ 0 & 0 & X^{\circ d} \otimes I_{n+1} \end{bmatrix}$$
$$Y' \mapsto \begin{bmatrix} X' & 0 & \\ 0 & (X')^{\circ(d-1)} \otimes I_{n-1} & \\ 0 & 0 & (X')^{\circ d} \otimes I_{n+1} \end{bmatrix}, \tag{16}$$

where $X^{\circ(d-1)}$ denotes the $(d-1)$-fold Hadamard product $X \circ X \circ \cdots \circ X$, namely, $(X^{\circ(d-1)})_{ij} = x_{ij}^{d-1}$. We will show that the equations for equivalence of $r(f), r(f')$, after this substitution, can be derived from the equations for monomial equivalence of f, f' in low-degree PC.

(Note that the substitutions above correspond precisely to the forward direction of the many-one reduction, in which $X \oplus (D^{-1}P \otimes I_{n+1}) \oplus (P \otimes I_{n+1})$ served as an equivalence. For, once we have $x_{ij}^{d+1} - x_{ij}$, we have $X^{\circ(d-1)} = D^{d-1}P = D^{-1}P$, and $X^{\circ d} = D^d P = P$.)

Recall that these equations are $Y \cdot r(f) = r(f')$ and $YY' = Y'Y = \mathrm{Id}$. The latter equations are easier to handle so we begin with those. They become $X^{\circ c}(X')^{\circ c} = (X')^{\circ c}X^{\circ c} = \mathrm{Id}$ for $c \in \{1, d-1, d\}$. For $c = 1$, these are some of our starting equations. For $c > 1$, this is similar to the argument in Theorem 6.9 (see the argument around Equation (15)), iterated, resulting in a proof of degree $2c$ for any c – in this case, $2d$.

Now to the equation(s) $Y \cdot r(f) = r(f')$. After substitution, these become

$$\sum_{i,j,k \in [n]} T_{ijk}(Xu_i)(Xu_j)(Xu_k) + \sum_{i \in [n], j \in [n+1]} (Xu_i)(X^{\circ(d-1)}v_{ij})(X^{\circ d}w_{ij})$$
$$= \sum_{ijk} T'_{ijk} u_i u_j u_k + \sum_{ij} u_i v_{ij} w_{ij}. \tag{17}$$

Focusing on the first summations on both sides of the equation, we see these are precisely the equations $X \cdot f = f'$. After subtracting these off, we now deal with the remaining terms.

We have

$$\begin{aligned}
\sum_{ij} u_i v_{ij} w_{ij} &= \sum_{i \in [n], j \in [n+1]} (Xu_i)(X^{\circ(d-1)}v_{ij})(X^{\circ d}w_{ij}) \\
&= \sum_{i \in [n], j \in [n+1]} \left(\sum_{k \in [n]} x_{k,i} u_k \right) \left(\sum_{\ell \in [n]} x_{\ell,i}^{d-1} v_{\ell,j} \right) \left(\sum_{h \in [n]} x_{h,i}^{d} w_{h,j} \right) \\
&= \sum_{k,\ell \in [n], j \in [n+1]} u_k v_{\ell,j} w_{\ell,j} \left(\sum_{i \in [n]} x_{k,i} x_{\ell,i}^{d-1} x_{\ell,i}^{d} \right) \\
&\quad + \sum_{\substack{k,\ell,h \in [n], j \in [n+1] \\ \ell \neq h}} u_k v_{\ell,j} w_{\ell',j} \left(\sum_{i \in [n]} x_{k,i} x_{\ell,i}^{d-1} x_{h,i}^{d} \right)
\end{aligned}$$

This becomes the system of equations

$$\begin{aligned}
\delta_{k,\ell} &= \textstyle\sum_{i \in [n]} x_{k,i} x_{\ell,i}^{d-1} x_{\ell,i}^{d} && (\forall k, \ell \in [n]) \\
0 &= \textstyle\sum_{i \in [n]} x_{k,i} x_{\ell,i}^{d-1} x_{h,i}^{d} && (\forall k, \ell, h \in [n], \ell \neq h).
\end{aligned}$$

(Note that technically we should quantify over all $j \in [n+1]$, but j plays no role in these equations – it just serves to repeat the same equation $n+1$ times. This corresponds to the fact that the lower-right part of our matrices have the form $* \otimes I_{n+1}$.)

When $k \neq \ell$, every term in the first equation is a degree-$2d$ multiple of the monomial axiom $x_{k,i}x_{\ell,i}$. Similarly, every term in the second set of equations is a degree-$2d$ multiple of the monomial axiom $x_{\ell,i}x_{h,i}$. Thus all that remains is the first equation when $k = \ell$, namely, $1 = \sum_{i\in[n]} x_{k,i}x_{k,i}^{d-1}x_{k,i}^{d}$. This is derived in Lemma 6.11, with $c = 2$ in degree $2d$ (since $d > 1$, we have $\max\{2d, d+2\} = 2d$). This completes the proof that we have a $(d, 2d)$-reduction. ◀

▶ Remark 6.13. There is a slightly simpler and smaller many-one reduction, namely $f \mapsto f + \sum_{i\in[n],j\in[n+1]} u_i v_{ij}^2$. However, in using that reduction, the witness for the forward direction becomes $X \oplus (D^{-1/2}P \otimes I_{n+1})$. This square root introduces a square into the equations that made it difficult to show that it was also a PC reduction. The reduction above fixes this issue.

6.4 From cubic forms to tensors

Our reductions here are those from Futorny–Grochow–Sergeichuk [19, Cor. 3.4 and Thm. 2.1]. The many-one property follows from the results there. We prove that each of these reductions is in fact also a low-degree PC reduction between the corresponding polynomial solvability problems. They reduce first to a problem we call Block Tensor Isomorphism, and then from there to Tensor Isomorphism, so we begin by introducing the former problem and its associated equations.

▶ **Definition 6.14** (see Futorny–Grochow–Sergeichuk [19]). *A block $n \times m \times p$ 3-way array is a 3-way array together with a partition of its index sets $\{1,\dots,n\} = \{1,\dots,n_1\} \sqcup \{n_1+1, n_1+2,\dots,n_1+n_2\} \sqcup \cdots \sqcup \{\sum_{i=1}^{N-1} n_i + 1,\dots,n\}$, and similarly for the other two directions. Two block 3-way arrays are said to be* conformally partitioned *if they have the same size and the same partitions of their index sets. Two conformally partitioned 3-way arrays T, T' with block sizes as above are* block-isomorphic *(called "block-equivalent" in [19]) if there exist invertible matrices $S_{11},\dots,S_{1,N}, S_{21},\dots,S_{2M}, S_{31},\dots,S_{3P}$, where $S_{1,I}$ is of size $n_I \times n_I$, $S_{2,J}$ is of size $m_J \times m_J$, and $S_{2,K}$ is of size $p_K \times p_K$, such that the block-diagonal matrices give an isomorphism of tensors:*

$$(S_{11} \oplus S_{12} \oplus \cdots \oplus S_{1N}, S_{21} \oplus \cdots \oplus S_{2M}, S_{31} \oplus \cdots \oplus S_{3P}) \cdot T = T'.$$

Given two block 3-way arrays T, T' as above, the equations for Block Tensor Isomorphism are as folllows. There are $2(\sum_{I\in[N]} n_i + \sum_{J\in[M]} m_J + \sum_{K\in[P]} p_K)$ variables arranged into $2(N+M+P)$ square matrices X_I, X_I' (of size $n_I \times n_I$), Y_J, Y_J' (of size $m_J \times m_J$), and Z_K, Z_K' (of size $p_K \times p_K$). Then the equations are:

$$(X_1 \oplus \cdots \oplus X_N, Y_1 \oplus \cdots \oplus Y_M, Z_1 \oplus \cdots \oplus Z_P) \cdot T = T'$$

$$X_I X_I' = X_I' X_I = \mathrm{Id} \qquad Y_J Y_J' = Y_J' Y_J = \mathrm{Id} \qquad Z_K Z_K' = Z_K' Z_K = \mathrm{Id},$$

for all $I \in [N]$, $J \in [M]$, $K \in [P]$.

▶ **Lemma 6.15.** *The many-one reduction from*

Equivalence of Noncommutative Cubic Forms
to
Block Tensor Isomorphism

in [19, Cor. 3.4] is in fact a linear-size (1,3)-many-one reduction.

Proof. Given a noncommutative cubic form f in n variables, $f = \sum_{i,j,k\in[n]} T_{ijk}u_iu_ju_k$, we recall the block tensor $r(T)$ from [19, Cor. 3.4]. It is partitioned into $2\times 3\times 3$ many blocks, with the rows being partitioned into $n, 1$, the columns into $n, n, 1$, and the depths also into $n, n, 1$; thus its total size is $(n+1)\times(2n+1)\times(2n+1)$. Let E_{ijk} denote the tensor of this size whose only nonzero entry is a 1 in position (i, j, k). Then we define

$$r(T) = T + \sum_{i\in[n]} (E_{i,n+i,2n+1} + E_{i,2n+1,n+i} + E_{n+1,i,n+i} + E_{n+1,n+i,i}) + E_{n+1,2n+1,2n+1}$$

If you wanted to think of this as part of the tensor corresponding to a cubic form, that cubic form would have $n+1$ new variables $v_1, \ldots, v_n, z$, and the form would be:

$$r(f) := f + \sum_{i\in[n]} (u_iv_iz + u_izv_i + zu_iv_i + zv_iu_i) + z^3.$$

(This doesn't quite line up with the above description of a tensor, as the tensor corresponding to $r(f)$ would necessarily have all 3 side lengths the same, $2n+1$. However, there are n of the $2n+1$ rows in that tensor that are entirely zero, namely, the rows corresponding to those monomials that begin with a v_i.)

The equations for block isomorphism of $r(T)$ and $r(T')$ have the following variable matrices X, X' are $n\times n$, x, x' are 1×1, Y_1, Y_1', Y_2, Y_2' are $n\times n$, y, y' are 1×1, Z_1, Z_1', Z_2, Z_2' are $n\times n$, and z, z' are 1×1. Let U, U' be the $n\times n$ variable matrices for the equations for equivalence of the noncommutative cubic forms f, f'. We consider the following substitution:

$$X, Y_1, Z_1, Y_2', Z_2' \mapsto U \qquad X', Y_1', Z_1', Y_2, Z_2 \mapsto U' \qquad x, x', y, y', z, z' \mapsto 1.$$

Under this substitution, the equations for block isomorphism of $r(T), r(T')$ become

$$\begin{aligned}(U,U,U)\cdot T + \sum_{i\in[n]} &((U,U',1)\cdot E_{i,n+i,2n+1} + (U,1,U')\cdot E_{i,2n+1,n+i}\\ &+ (1,U,U')\cdot E_{n+1,i,n+i} + (1,U',U)\cdot E_{n+1,n+i,i}\\ &+(1,1,1)\cdot E_{n+1,2n+1,2n+1})\\ =&T' + \sum_{i\in[n]} (E_{i,n+i,2n+1} + E_{i,2n+1,n+i} + E_{n+1,i,n+i} + E_{n+1,n+i,i})\\ &+ E_{n+1,2n+1,2n+1}\end{aligned}$$

Now, because each summand inside the big sum corresponds to an identity matrix in a block (e.g. $\sum_{i\in[n]} E_{i,n+i,2n+1}$ is an identity matrix in rows $\{1,\ldots,n\}$, columns $\{n+1,\ldots,2n\}$, and depth $2n+1$), the above equations give us many instances of $UU' = \mathrm{Id}$ and $U'U = \mathrm{Id}$, which is one of our starting equations. We also get the equation $1 = 1$, and lastly, $(U,U,U)\cdot T = T'$, which is another one of our starting equations. Thus the equations we get here are in fact precisely the same as the equations we started with. As these are cubic equations and the substitutions were linear, it is a (1,3)-PC reduction. ◀

▶ **Lemma 6.16.** *When the number of blocks is $O(1)$, the many-one reduction from*

Block Tensor Isomorphism
to
Tensor Isomorphism

in [19, Thm. 2.1] is in fact a quadratic-size (1,3)-many-one reduction.

Note that the output of the reduction of Lemma 6.15 has $2 \times 3 \times 3$ many blocks, so the restriction to $O(1)$ many blocks in Lemma 6.16 presents no obstacle to our goal.

Proof. The key is to show how to effectively remove the partition in one of the three directions; then that reduction can be applied three times in the three separate directions. Let T, T' be block tensors of size $n \times m \times p$, with $N \times M \times P$ many blocks. The construction of [19, Lem. 2.2] shows how to construct from this a block tensor of quadratic size with $N \times M \times 1$ many blocks. We recall the construction here and show that it is a (1,3)-PC reduction.

Let $p_1, \ldots, p_P$ denote the sizes of the parts of the partition in the third direction. Let $r = \min\{n, m\} + 1$ – this will govern the rank of the identity matrix gadgets we add. Let $s = \sum_{K=1}^{P} 2^{K-1} r$ and $t = \sum_{K=1}^{P} 2^{K-1} r p_K$. Then the output tensor will have size $(n+s) \times (m+t) \times p$. (Note that, since $P = O(1)$, we have that s is linearly bounded in n, m and t is quadratic as a function of n, m, p.) Let $T_1, \ldots, T_p$ be the frontal slices of T. The i-th slice of $r(T)$ will be as follows. Suppose i is in the K-th block, and write $i = i_0 + \sum_{k=1}^{K} p_k$ with $1 \le i_0 \le p_{K+1}$. Write the slices T_i as $T_i = \begin{bmatrix} A_i & B_i \\ C_i & D_i \end{bmatrix}$, where A_i is $n_1 \times m_1$ – representing the first part of the partition of T into rows and columns, and D_i represents all the other parts. Then we construct:

$$r(T)_i := \left[\begin{array}{ccc|c|c|c|c} 0 & \cdots & 0 & & & & \\ \hline & \ddots & & & & & \\ \hline & & 0 \cdots I_{2^{K-1}r} \cdots 0 & & & & \\ \hline & & & \ddots & & & \\ \hline & & & & 0 \cdots 0 & & \\ \hline & & & & & A_i & B_i \\ \hline & & & & & C_i & D_i \end{array}\right],$$

where the $I_{2^{K-1}r}$ is in the i_0-th position within the K-th block-row and block-column as indicated by the dashed lines. Here the dashed lines *do not* represent additional parts of the partition, they are just for visual clarity. The solid lines indicate the first part of the new partition into rows and columns. The rows of C_i and D_i are partitioned into blocks the same as they were originally in T_i, and the columns of B_i and D_i are partitioned into parts in the same way as they were originally in T_i. That is, the entire big gadget in the upper-left gets prepended to the first parts of the row and column partitions. This is the many-one reduction.

Let $X_1, \ldots, X_N$, $Y_1, \ldots, Y_M$, and Z be variable matrices (with associated primes matrices X_1', etc.), with sizes as follows:

- X_1 has size $(s + n_1) \times (s + n_1)$
- X_I for $I \ge 2$ has size $n_I \times n_I$
- Y_1 has size $(t + m_1) \times (t + m_1)$
- Y_J for $J \ge 2$ has size $m_J \times m_J$
- Z has size $p \times p$.

We start from the equations for Block Isomorphism (but now where there is only one block in the third direction), namely

$$X_I X_I' = X_I' X_I = \mathrm{Id} \qquad Y_J Y_J' = Y_J' Y_J = \mathrm{Id} \qquad Z Z' = Z' Z = \mathrm{Id}$$

and

$$(X_1 \oplus \cdots \oplus X_N, Y_1 \oplus \cdots \oplus Y_N, Z) \cdot r(T) = r(T').$$

We make the following substitution (with the same substitutions, *mutatis mutandis*, for the primed variables):

- $X_1 \mapsto I_s \oplus \hat{X}_1$, where $\hat{X}_1$ is a matrix of variables of size $n_1 \times n_1$.
- For $I \geq 2$, X_I maps to itself.
- $Y_1 \mapsto I_t \oplus \hat{Y}_1$, where $\hat{Y}_1$ is a matrix of variables of size $m_1 \times m_1$.
- For $J \geq 2$, Y_J maps to itself.
- Z maps to a block matrix $Z_1 \oplus \cdots \oplus Z_P$, where for each $K \in [P]$, we have Z_K is a $p_K \times p_K$ matrix of variables.

Under these substitutions, the equations for Block Isomorphism of $r(T), r(T')$ become precisely the original equations for Block Isomorphism of T, T', together with equations of the form $I_s E_i I_t = E_i$, where E_i is the $s \times t$ gadget matrix in the upper-left in the i-th slice. Thus we get a $(1, 3)$-reduction.

Finally, this is then repeated in the other two directions to reduce the number of blocks in all three directions to one, thus giving an instance of Tensor Isomorphism. ◀

6.5 Putting it all together

Finally, we combine all the above to prove Theorem 6.2.

Proof of Theorem 6.2. Let $m = cn$ with $c \geq 10^4$. By Theorem 6.1, random 3XOR instances with clause density c require PC degree $\Omega(n/c^2) = \Omega(n)$ (in our case) to refute. The number of instances that the random distribution assigns nonzero probability is $\binom{2\binom{n}{3}}{m} \sim \binom{n^3}{cn} \geq n^{3cn}/(cn)^{cn} = c^{2cn \log n - cn} \geq c^{\Omega(n \log n)}$.

By Theorem 6.6, there is a (1,3)-many-one reduction from those instances to $\{\pm 1\}$-Monomial Equivalence of $\{\pm 1\}$ Multilinear Noncommutative Cubic Forms, where the number of variables in the cubic form is the same as the number of variables in the 3XOR instance. By Theorem 6.9 there is then a (2,6)-many-one reduction to Monomial Equivalence of $\{\pm 1\}$ Noncommutative Cubic Forms, where the number of variables in the output cubic form is linear in the original number of variables, and such that the output forms have the property that any monomial equivalence between them has all its nonzero entries being 6-th roots of unity. This thus satisfies the hypothesis of Theorem 6.12 with $d = 6$, so there is a (6,12)-many-one reduction to Equivalence of Noncommutative Cubic Forms, where the output has a quadratic number of variables compared to the input. Finally, combining Lemmata 6.15 and 6.16, we get a (1,3) reduction from Equivalence of Noncommutative Cubic Forms to Tensor Isomorphism, which further increases the size quadratically. In total, the size increases multiply, yielding a quartic size increase. The substitution degrees multiply and the derivation degrees we take the max, yielding a (12,12)-many-one reduction from Random 3XOR to Tensor Isomorphism on tensors of size $O(n^4) \times O(n^4) \times O(n^4)$. By Lemma 2.2, any PC refutation of these Tensor Isomorphism instances requires degree $\Omega(n)$. ◀

We note that our lower bound for tensor isomorphism also applies to the stronger Sum-of-Squares proof system. This is due to the fact that there is lower bound for random 3XOR in Sum-of-Squares, as shown by Grigoriev [21] and independently by Schoenbeck [44], which makes the dependence on the clause density explicit.

▶ **Theorem 6.17** ([44, Theorem 12]). *A random 3-XOR instance with clause density $\Delta = m/n = dn^\epsilon$, for all sufficiently large constants d, requires SoS degree $\Omega(n^{1-\epsilon})$ to refute, with probability $1 - o(1)$.*

In particular, this is a linear $\Omega(n)$ lower bound in the case of constant clause density ($\epsilon = 0$), which matches the PC lower bound of Theorem 6.1.

As we observe all of our reductions go through in Sum-of-Squares, since Sum-of-Squares simulates PC over the reals due to Berkholz [6]. Furthermore, this simulation preserves degrees of proofs up to a constant factor.

▶ **Theorem 6.18** ([6, Theorem 1.1]). *If a system of polynomial equations $\mathcal{F}$ over the reals has a PC refutation of degree d and size s, it also has a sum-of-squares refutation of degree $2d$ and size $poly(s)$.*

Hence, by combining Theorems 6.17, 6.18 and the PC reduction used to prove 6.2, we obtain the following lower bound for tensor isomorphism in Sum-of-Squares.

▶ **Theorem 6.19.** *Over the real numbers, there is a distribution on $n \times n \times n$ Tensor Isomorphism whose associated equations require SoS degree $\Omega(\sqrt[4]{n})$ to refute with probability $1 - o(1)$.*

7 Open Questions

Beyond Conjecture 1.7, we highlight several more questions we find interesting about the algebraic proof complexity of Tensor Isomorphism.

7.1 Degree

▶ **Open Question 7.1.** What is the correct value for the PC degree of rank-r Tensor Isomorphism?

Note that by using the reductions from Section 6, we can produce (random) $r \times r \times r$ tensors that require PC degree $\Omega(r^{1/4})$ to refute. However, the number of variables is $6r^2$, this lower bound is only $\Omega(N^{1/8})$ where N is the number of variables. Since their rank could be as large as $R = \Theta(r^2)$ (and indeed, very likely is), the upper bound we get from Theorem 4.1 is only $2^{O(r^4)}$ (without the $x^q - x$ axioms) or $O(r^4)$ (with the $x^q - x$ axioms, with $q = O(1)$). Even in the latter case, this leaves a polynomial gap between the lower and upper bounds (without those the gap is exponential).

We note that the upper bound in Theorem 4.1 without the $x^q - x$ equations already applies to the weaker Nullstellensatz proof system. Is there a polynomial upper bound on PC degree – as a function of rank – without the $x^q - x$ axioms?

7.2 Size

In the presence of the Boolean axioms, there is a size-degree tradeoff for PC (or even PCR – a system with the same degree bounds as PC, but is stronger when measuring size by number of monomials or number of symbols) [16, 2]. This implies that in the presence of the Boolean axioms, a good degree lower bound implies a good size lower bound. But TI does not have the Boolean axioms.

▶ **Open Question 7.2.** Get lower and upper bounds on the *size* of PC proofs for Tensor (Non-)Isomorphism. Are there subexponential size upper bounds, despite the polynomial degree lower bounds?

7.3 Other matrix problems

While many different tensor-related problems are all equivalent to TI, in the case of matrices, we have three genuinely different problems: matrix equivalence (2-TI), matrix conjugacy, and matrix congruence. Conjugacy is determined by the Rational Normal Form or Jordan Normal Form, while congruence depends on the field (e.g., over algebraically closed fields it only depends on rank, over $\mathbb{R}$ it depends on the signature, and over finite fields it depends on whether the determinant is a square or not).

▶ **Open Question 7.3.** What is the PC complexity (size, degree, etc.) of matrix conjugacy? Of matrix congruence?

More precisely, for conjugacy we have in mind the system of equations:

$$XM = M'X \qquad XX' = X'X = I,$$

and for congruence the system of equations:

$$XMX^T = M' \qquad XX' = X'X = I.$$

7.4 Bounded border rank

Not only can testing a tensor for bounded rank can be done in polynomial time (Remark 1.4), testing a tensor for bounded *border*-rank can also be done in polynomial time (see, e. g., [24]), by evaluating a polynomial number of easy-to-evaluate equations. While several partial results are available, the gap for what is known about the ratio between rank and border rank is quite large: there are 3-tensors known whose rank approaches 3 times their border rank [50], but the currently known upper bound is Lehmkuhl and Lickteig [33], who show that for tensors of border rank b, the ratio of rank to border rank is at most $c^{\Theta(nb)}$. See the Zuiddam's introduction [50] for more details.

▶ **Open Question 7.4.** What is the PC degree of testing isomorphism of tensors of bounded border-rank? Can such tests be done (by any method) in polynomial time?

7.5 Relating different reductions from Graph Isomorphism

While we chose a particular reduction from GI to TI for the lower bound in Section 5, we are aware of several others, including:

- GI to Permutational Code Equivalence [42, 35, 38], then to Matrix Lie Algebra Conjugacy [23], then to TI [19];
- GI to Semisimple Matrix Lie Algebra Conjugacy [23], and then to TI [19];
- GI to Alternating Matrix Space Isometry [26, 27], then to TI [19];
- GI to Algebra Isomorphism [20, 1], then to TI [19].

We believe all of these can be realized as low-degree PC reduction as well. In the first arXiv version of [26], they asked which of these might be equivalent in some sense (though there the final target was Alternating Matrix Space Isometry, another TI-complete problem, rather than TI itself). Here we make this question slightly more precise, in terms of PC reductions:

▶ **Open Question 7.5.** Which, if any, of the reductions above from Graph Isomorphism to Tensor Isomorphism are equivalent under low-degree PC?

References

1 Manindra Agrawal and Nitin Saxena. Automorphisms of finite rings and applications to complexity of problems. In *STACS 2005, 22nd Annual Symposium on Theoretical Aspects of Computer Science, Proceedings*, pages 1–17, 2005. doi:10.1007/978-3-540-31856-9_1.

2 Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM J. Comput.*, 34(1):67–88, 2004. doi:10.1137/S0097539701389944.

3 Albert Atserias and Elitza N. Maneva. Sherali–Adams relaxations and indistinguishability in counting logics. *SIAM J. Comput.*, 42(1):112–137, 2013. doi:10.1137/120867834.

4 László Babai. Graph isomorphism in quasipolynomial time [extended abstract]. In *STOC'16 – Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 684–697. ACM, New York, 2016. doi:10.1145/2897518.2897542.

5 Eli Ben-Sasson and Russell Impagliazzo. Random CNF's are hard for the Polynomial Calculus. In *40th Annual Symposium on Foundations of Computer Science, FOCS '99, 17-18 October, 1999, New York, NY, USA*, pages 415–421. IEEE Computer Society, 1999. (Journal version in *Comput. Complex.* 2010, doi:10.1007/s00037-010-0293-1). doi:10.1109/SFFCS.1999.814613.

6 Christoph Berkholz. The relation between polynomial calculus, sherali-adams, and sum-of-squares proofs. In *35th Symposium on Theoretical Aspects of Computer Science (STACS 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

7 Christoph Berkholz and Martin Grohe. Limitations of algebraic approaches to graph isomorphism testing. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming – 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 155–166. Springer, 2015.

8 Christoph Berkholz and Martin Grohe. Linear diophantine equations, group csps, and graph isomorphism. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 327–339. SIAM, 2017. Preprint arXiv:1607.04287 [cs.CC]. doi:10.1137/1.9781611974782.21.

9 Jendrik Brachter and Pascal Schweitzer. On the Weisfeiler–Leman dimension of finite groups. In Holger Hermanns, Lijun Zhang, Naoki Kobayashi, and Dale Miller, editors, *LICS '20: 35th Annual ACM/IEEE Symposium on Logic in Computer Science, Saarbrücken, Germany, July 8-11, 2020*, pages 287–300. ACM, 2020. doi:10.1145/3373718.3394786.

10 Jendrik Brachter and Pascal Schweitzer. A systematic study of isomorphism invariants of finite groups via the Weisfeiler–Leman dimension. In Shiri Chechik, Gonzalo Navarro, Eva Rotenberg, and Grzegorz Herman, editors, *30th Annual European Symposium on Algorithms, ESA 2022, September 5-9, 2022, Berlin/Potsdam, Germany*, volume 244 of *LIPIcs*, pages 27:1–27:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ESA.2022.27.

11 R. P. Brent. Algorithms for matrix multiplication. Stanford Computer Science Dept. Tech. Report STAN-CS-70-157, available online at https://apps.dtic.mil/sti/pdfs/AD0705509.pdf, 1970.

12 Peter A. Brooksbank, Joshua A. Grochow, Yinan Li, Youming Qiao, and James B. Wilson. Incorporating Weisfeiler–Leman into algorithms for group isomorphism. arXiv:1905.02518 [cs.CC], 2019.

13 Josh Buresh-Oppenheim, Matthew Clegg, Russell Impagliazzo, and Toniann Pitassi. Homogenization and the polynomial calculus. *Comput. Complex.*, 11(3-4):91–108, 2002. doi:10.1007/s00037-002-0171-6.

14 Samuel R. Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *J. Comput. Syst. Sci.*, 62(2):267–289, 2001. doi:10.1006/jcss.2000.1726.

15 Jin-Yi Cai, Martin Fürer, and Neil Immerman. An optimal lower bound on the number of variables for graph identification. *Combinatorica*, 12(4):389–410, 1992. doi:10.1007/BF01305232.

16 Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the Groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the Twenty-eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*, pages 174–183. ACM, New York, 1996. doi:10.1145/237814.237860.

17 Nathaniel A. Collins and Michael Levet. Count-free Weisfeiler–Leman and group isomorphism. arXiv:2212.11247 [cs.DS], 2022.

18 Jean-Charles Faugère and Ludovic Perret. Polynomial equivalence problems: Algorithmic and theoretical aspects. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 – June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 30–47. Springer, 2006. doi:10.1007/11761679_3.

19 Vyacheslav Futorny, Joshua A. Grochow, and Vladimir V. Sergeichuk. Wildness for tensors. *Linear Algebra Appl.*, 566:212–244, 2019. doi:10.1016/j.laa.2018.12.022.

20 D. Ju. Grigoriev. Complexity of "wild" matrix problems and of the isomorphism of algebras and graphs. *Zap. Nauchn. Sem. Leningrad. Otdel. Mat. Inst. Steklov. (LOMI)*, 105:10–17, 198, 1981. Theoretical applications of the methods of mathematical logic, III. doi:10.1007/BF01084390.

21 Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1-2):613–622, 2001.

22 Dima Grigoriev. Polynomial complexity of solving systems of few algebraic equations with small degrees. In Vladimir P. Gerdt, Wolfram Koepf, Ernst W. Mayr, and Evgenii V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing – 15th International Workshop, CASC 2013, Berlin, Germany, September 9-13, 2013. Proceedings*, volume 8136 of *Lecture Notes in Computer Science*, pages 136–139. Springer, 2013. doi:10.1007/978-3-319-02297-0_11.

23 Joshua A. Grochow. Matrix Lie algebra isomorphism. In *IEEE Conference on Computational Complexity (CCC12)*, pages 203–213, 2012. Also available as arXiv:1112.2012 [cs.CC] and ECCC Technical Report TR11-168. doi:10.1109/CCC.2012.34.

24 Joshua A. Grochow. Answer to "deciding bound on tensor rank for a fixed value". CSTheory StackExchange, https://cstheory.stackexchange.com/a/19518/129, 2013.

25 Joshua A. Grochow and Youming Qiao. On p-group isomorphism: Search-to-decision, counting-to-decision, and nilpotency class reductions via tensors. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 16:1–16:38. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.16.

26 Joshua A. Grochow and Youming Qiao. On the complexity of isomorphism problems for tensors, groups, and polynomials I: tensor isomorphism-completeness. In James R. Lee, editor, *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*, volume 185 of *LIPIcs*, pages 31:1–31:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ITCS.2021.31.

27 Xiaoyu He and Youming Qiao. On the Baer-Lovász-Tutte construction of groups from graphs: isomorphism types and homomorphism notions. *European J. Combin.*, 98:Paper No. 103404, 12, 2021. doi:10.1016/j.ejc.2021.103404.

28 Harald Andrés Helfgott. Isomorphismes de graphes en temps quasi-polynomial [d'après Babai et Luks, Weisfeiler–Leman,...]. *Astérisque*, (407):Exp. No. 1125, 135–182, 2019. Séminaire Bourbaki. Vol. 2016/2017. Exposés 1120–1135. English translation with appendices by Jitendra Bajpai and Daniele Dona available at arXiv:17010.04574 [math.GR]. doi:10.24033/ast.

29 Pavel Hrubes and Iddo Tzameret. Short proofs for the determinant identities. *SIAM J. Comput.*, 44(2):340–383, 2015. doi:10.1137/130917788.

30 Zhengfeng Ji, Youming Qiao, Fang Song, and Aaram Yun. General linear group action on tensors: A candidate for post-quantum cryptography. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography – 17th International Conference, TCC 2019, Nuremberg, Germany, December 1-5, 2019, Proceedings, Part I*, volume 11891 of *Lecture Notes in Computer Science*, pages 251–281. Springer, 2019. Preprint `arXiv:1906.04330 [cs.CR]`. `doi:10.1007/978-3-030-36030-6_11`.

31 J. M. Landsberg. *Tensors: geometry and applications*, volume 128 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, RI, 2012. `doi:10.1090/gsm/128`.

32 Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM J. Optim.*, 11(3):796–817, 2000/01. `doi:10.1137/S1052623400366802`.

33 Thomas Lehmkuhl and Thomas Lickteig. On the order of approximation in approximative triadic decompositions of tensors. *Theoret. Comput. Sci.*, 66(1):1–14, 1989. `doi:10.1016/0304-3975(89)90141-2`.

34 Thomas Lickteig. Typical tensorial rank. *Linear Algebra Appl.*, 69:95–120, 1985. `doi:10.1016/0024-3795(85)90070-9`.

35 Eugene M. Luks. Permutation groups and polynomial-time computation. In *Groups and computation (New Brunswick, NJ, 1991)*, volume 11 of *DIMACS Ser. Discrete Math. Theoret. Comput. Sci.*, pages 139–175. Amer. Math. Soc., Providence, RI, 1993.

36 Brendan D. McKay. Practical graph isomorphism. *Congr. Numer.*, 30:45–87, 1981.

37 Brendan D. McKay and Adolfo Piperno. Practical graph isomorphism, II. *J. Symbolic Comput.*, 60:94–112, 2014. `doi:10.1016/j.jsc.2013.09.003`.

38 Takunari Miyazaki. Luks's reduction of graph isomorphism to code equivalence. Comment to E. W. Clark, `https://groups.google.com/forum/#!msg/sci.math.research/puZxGj9HXKI/CeyH2yyyNFUJ`, 1996.

39 Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry Vetrov. Tensorizing neural networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems – Volume 1*, NIPS'15, pages 442–450. MIT Press, 2015.

40 Ryan O'Donnell, John Wright, Chenggang Wu, and Yuan Zhou. Hardness of robust graph isomorphism, lasserre gaps, and asymmetry of random graphs. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1659–1677. SIAM, 2014. Preprint available as arXiv:`1401.2436 [cs.CC]`. `doi:10.1137/1.9781611973402.120`.

41 Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): two new families of asymmetric algorithms. In *Advances in Cryptology – EUROCRYPT '96, International Conference on the Theory and Application of Cryptographic Techniques, Saragossa, Spain, May 12-16, 1996, Proceeding*, pages 33–48, 1996. `doi:10.1007/3-540-68339-9_4`.

42 Erez Petrank and Ron M. Roth. Is code equivalence easy to decide? *IEEE Trans. Inf. Theory*, 43(5):1602–1604, 1997. `doi:10.1109/18.623157`.

43 Alexander A. Razborov. Lower bounds for the polynomial calculus. *Comput. Complex.*, 7(4):291–324, 1998. `doi:10.1007/s000370050013`.

44 Grant Schoenebeck. Linear level lasserre lower bounds for certain k-csps. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 593–602. IEEE, 2008.

45 Aaron Snook, Grant Schoenebeck, and Paolo Codenotti. Graph Isomorphism and the Lasserre hierarchy. `arXiv:1401.0758 [cs.CC]`, 2014.

46 Michael Soltys. *The complexity of derivations of matrix identities*. PhD thesis, University of Toronto, 2001. Availalble on ECCC at `https://eccc.weizmann.ac.il/resources/pdf/soltys.pdf`.

47 Michael Soltys and Stephen Cook. The proof complexity of linear algebra. *Ann. Pure Appl. Logic*, 130(1-3):277–323, 2004. `doi:10.1016/j.apal.2003.10.018`.

48 Martín Sombra. A sparse effective Nullstellensatz. *Adv. in Appl. Math.*, 22(2):271–295, 1999. `doi:10.1006/aama.1998.0633`.

49 Gang Tang, Dung Hoang Duong, Antoine Joux, Thomas Plantard, Youming Qiao, and Willy Susilo. Practical post-quantum signature schemes from isomorphism problems of trilinear forms. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022 – 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30 – June 3, 2022, Proceedings, Part III*, volume 13277 of *Lecture Notes in Computer Science*, pages 582–612. Springer, 2022. doi:10.1007/978-3-031-07082-2_21.

50 Jeroen Zuiddam. A note on the gap between rank and border rank. *Linear Algebra Appl.*, 525:33–44, 2017. doi:10.1016/j.laa.2017.03.015.

Generative Models of Huge Objects

Lunjia Hu
Department of Computer Science, Stanford University, CA, USA

Inbal Rachel Livni Navon
Department of Computer Science, Stanford University, CA, USA

Omer Reingold
Department of Computer Science, Stanford University, CA, USA

Abstract

This work initiates the systematic study of explicit distributions that are indistinguishable from a *single* exponential-size combinatorial object. In this we extend the work of Goldreich, Goldwasser and Nussboim (SICOMP 2010) that focused on the implementation of huge objects that are indistinguishable from the uniform distribution, satisfying some global properties (which they coined truthfulness). Indistinguishability from a single object is motivated by the study of generative models in learning theory and regularity lemmas in graph theory. Problems that are well understood in the setting of pseudorandomness present significant challenges and at times are impossible when considering generative models of huge objects.

We demonstrate the versatility of this study by providing a learning algorithm for huge indistinguishable objects in several natural settings including: dense functions and graphs with a truthfulness requirement on the number of ones in the function or edges in the graphs, and a version of the weak regularity lemma for *sparse* graphs that satisfy some global properties. These and other results generalize basic pseudorandom objects as well as notions introduced in algorithmic fairness. The results rely on notions and techniques from a variety of areas including learning theory, complexity theory, cryptography, and game theory.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation → Random network models; Theory of computation → Generating random combinatorial structures

Keywords and phrases pseudorandomness, generative models, regularity lemma

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.5

Related Version *Full Version*: `https://arxiv.org/abs/2302.12823` [17]

Funding *Lunjia Hu*: Supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness, Omer Reingold's NSF Award IIS-1908774, and Moses Charikar's Simons Investigators award.
Inbal Rachel Livni Navon: Supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness, the Sloan Foundation Grant 2020-13941, and the Zuckerman STEM Leadership Program.
Omer Reingold: Supported by the Simons Foundation Collaboration on the Theory of Algorithmic Fairness and the Simons Foundation Investigators award 689988.

1 Introduction

A pseudorandom distribution is indistinguishable from the uniform distribution to a set of computationally bounded distinguishers. Pseudorandomness is a cornerstone of many areas of computer science and mathematics. The variability of pseudorandom distributions stems from the different objects they can generate (bit strings, functions, permutations and more) and the different computational bounds that can be imposed on the distinguishers. In the area of cryptography, it is typical to consider powerful distinguishers that are at least

COMPUTATIONAL COMPLEXITY CONFERENCE

Table 1 Comparison between problem settings.

	What do we imitate?	What do we construct?
Pseudorandomness	distribution of objects	distribution of objects
Explicit construction (e.g. expander graphs)	distribution of objects	single object
Our setup: generative models	single object	distribution of objects

polynomial time, giving rise to central notions such as pseudorandom generators [3, 41], pseudorandom functions [8] and pseudorandom permutations [27]. More limited distinguishers give rise to other fundamental notions such as k-wise independent hashing and ε-biased distributions (cf. [40, 29]). In the area of explicit combinatorial constructions, we typically try to emulate the uniform distribution by a single object, rather than with a distribution. A primer example is the fundamental notion of expander graphs (see [16, 39] for surveys), with its multiple variants (including various notions of randomness extractors). These are graphs that are indistinguishable from a uniformly selected graph to a limited set of distinguishers (such as distinguishers that check if a random edge crosses a given cut).

In these classic areas of pseudorandomness, a distribution, or even a single object, is constructed to emulate a distribution (typically, the uniform distribution). In this paper we ask for a distribution to emulate *a single object* (Table 1). This reversal may seem absurd from the perspective of pseudorandomness but makes perfect sense from the perspective of generative models. An early exposure of the TOC community to generative models was with respect to the World Wide Web. These were models that produce distributions of graphs that imitate some properties of the Web, such as power law on the degrees of nodes (see [28] for a survey). At any given point, the web is a single graph, but it is also a very large graph that does not have a simple description. Generative models gave a useful way to analyze, or estimate through experimentation, the expected performance of protocols on the Web.

Other well studied generative models are the stochastic block model [15] and the more elaborate mixed membership stochastic block model [1]. Consider a graph representing some connections between individuals, such as the connectivity of the social network. The stochastic block model partitions the vertices into disjoint communities and for every two communities assigns a probability of connection. This model represents a distribution over graphs where for each two vertices, an edge is placed independently with the probability assigned to the pair of communities of its end points. (In the mixed-membership model, each vertex is assigned a distribution over communities.) These models help identify useful substructures within a social structure such as sub-communities or different social roles. But given a single social network, B^*, what is an appropriate model to capture it? After all, a model describes a distribution over networks rather than the single network we are trying to explain. A prevalent approach is to aim at the maximum likelihood model. Out of all models, the probability of sampling B^* is maximized under the maximum likelihood model. Heuristics for estimating the maximum likelihood model have been playing a major roles in the generative-model literature and in its application in practice. It should be noted that the probability that the model would produce B^* is often very small. In this light, the meaningfulness of a maximum-likelihood models may be debated and may depend on a particular setting.

From the perspective of indistinguishability, it may be more natural to seek a model that produce a distribution that is indistinguishable from B^* to a meaningful set of distinguishers. For example, in the case of the stochastic block model, natural distinguishers are defined

by two sets of vertices U and V and ask what is the probability that a random edge in the graph crosses from U to V. A stochastic block model that fool all such distinguishers is exactly what is given by the Frieze-Kannan regularity lemma (also known as the weak regularity lemma) [7]. The indistinguishability perspective on generative models and known connections between learning and pseudrandomness, which we will discuss shortly, are both a motivation as well as the starting point of this work.

1.1 Overall Goal: Indistinguishable Generative Models of Huge Objects

In many of the applications of generative models, such as modeling the Web or a social network, the objects being modeled are huge. In this paper, we aim at a *systematic theory of efficiently learning and implementing huge generative models*. Our models will generate a distribution of objects satisfying some global properties that are indistinguishable from a fixed combinatorial object. Such a theory presents non-trivial challenges that do not manifest themselves neither when generating huge pseudorandom objects, nor in generative models of polynomial-size objects.

Concretely, we assume that we have access to an object B^* with exponential size. For example, B^* could be a function with an exponentially large domain, or a graph with exponentially many vertices and edges. We are most interested in the case where the object B^* is too large to read or process as a whole, and we have to access it by sampling: for example, when B^* represents a function $f : X \to \{0,1\}$ with $|X|$ being exponentially large, we may access B^* by asking for random pairs $(x, f(x)) \in X \times \{0,1\}$ (sample access) or random inputs $x \in X$ conditioned on $f(x) = 1$ (support access). Given access to the huge object B^*, our goal is to create a generative model M for B^*. Here, our model M represents a distribution over objects, and we want to ensure that this distribution is indistinguishable from B^* to all distinguishers D in a class $\mathcal{D}$. Specifically, if we use $D^B \in \{\text{“accept”}, \text{“reject”}\}$ to denote the output of distinguisher D given sample/support access to object B, our indistinguishability requirement is that for every $D \in \mathcal{D}$,

$$|\Pr[D^{B^*} = \text{“accept”}] - \mathbb{E}_{B\sim M}[\Pr[D^B = \text{“accept”}]]| \leq \varepsilon.$$

We aim for building an *efficient* learner L that can output a model M satisfying the indistinguishability requirement above when given sample/support access to the true object B^*. When B^* is exponentially large (which is the case we are interested in), the output model M also needs to generate exponentially large objects, and thus we cannot expect an efficient learner to directly output M. Instead, we want our learner to output an *efficient implementation* of M, which, roughly speaking, is a randomized algorithm that can efficiently provide sample/support access to objects drawn from M (Definitions 13 and 14).

Our goal of learning a generative model M indistinguishable from the true object B^* is analogous to the problem addressed by Goldreich, Goldwasser and Nussboim [9] in the area of pseudorandomness. They study the problem of efficiently implementing a distribution of huge objects, satisfying some global properties, that are indistinguishable from the uniform distribution of such objects. Follow-up works of [9] such as [31, 30] study efficient implementations that are indistinguishable from certain distributions of huge random graphs. All these works aim to achieve indistinguishability from a known distribution of objects, whereas in our problem of learning a generative model, we assume that the true object B^* is initially *unknown*, and to collect information about B^*, we additionally need a learner L that can use sample/support access to B^* to *efficiently* construct an implementation of an indistinguishable model.

Beyond indistinguishability, we also aim to achieve the notion of *truthfulness* introduced in [9]. To demonstrate this notion, consider pseudorandom permutations $f_s : \{0,1\}^n \mapsto \{0,1\}^n$ [27]. A distribution of permutations is pseudorandom if it is indistinguishable from the uniform distribution of permutations. It should be noted that a pseudorandom $\{0,1\}^n \mapsto \{0,1\}^n$ function [8] is also indistinguishable from a random permutation over $\{0,1\}^n$ (as long as the number of queries are sufficiently smaller than $2^{n/2}$). Nevertheless, insisting that the pseudorandom objects satisfy the global condition of being a permutation is critical in the applications of pseudorandom permutations. This motivates the distinction of [9] between indistinguishability (that the pseudorandom objects are indistinguishable from a uniform object to a class of distinguishers) and *truthfulness* which is a global property that needs to hold exactly or approximately in a statistical sense. In our setup, a generative model M is truthful if every object B drawn from the distribution represented by M satisfies a certain global property. For example, when the true object B^* is a function $f^* : X \to \{0,1\}$ with support size $|\{x \in X : f^*(x) = 1\}|$ being k, a truthful requirement on a generative model M for B^* may restrict M to always generate functions with support size k.

The study of implementing huge pseudorandom objects [32, 9, 31, 30] has pseudorandom functions and permutations as vital building blocks. Besides these building blocks, our techniques for learning generative models of huge objects also come from connections to the regularity lemma and especially the work of Trevisan, Tulsiani and Vadhan [37]. In [37], they construct an efficiently-implementable function $f : X \mapsto [0,1]$ which is indistinguishable from some $f^* : X \mapsto [0,1]$ to a family of distinguishers represented by functions $g : X \mapsto [0,1]$. Indistinguishability here means that $|\mathbb{E}[f(x)g(x)] - \mathbb{E}[f^*(x)g(x)]|$ is smaller than some error parameter ε. After [37], the problem of creating indistinguishable functions and its applications to cryptography are further studied in [38, 22, 34, 35, 36, 4]. These works assume that the true function f^* is known and they do not explicitly deal with the problem of learning f^*, but the corresponding learning task has been studied in the algorithmic fairness literature through the notions of multi-accuracy, multi-calibration, and outcome indistinguishability [14, 23, 5, 13, 6]. When applying techniques from these works to solve some problems in our setting, we need to deal with additional challenges such as the truthfulness requirement that we want our generative model to satisfy.

1.2 Our Results

The main conceptual contribution of this paper is in suggesting a new frontier for the study of indistinguishability, which is highly motivated and technically challenging. As we introduce in Section 1.1, the notion of indistinguishability from a single huge object combines at its core the areas of learning theory and pseudorandomness which, as recent research uncovered, have deep connections, providing a way to describe and address a rich landscape of natural problems. Below we summarize the main problems we address in this new framework.

Truthful Learning That Preserves Support Size

Suppose we have sample access to a function $f^* : \{0,1\}^n \to \{0,1\}$ and we want to build an indistinguishable generative model for f^*. Here sample access allows us to observe pairs $(x, f^*(x))$ with x drawn uniformly at random from $\{0,1\}^n$, and accordingly, we assume that every distinguisher also decides to accept or reject based on such a random pair $(x, f(x))$ from a function f that may or may not be the true f^*. This task of learning a generative model for a binary function is closely related to the task of *no-access outcome indistinguishability* studied in [5], and it has been observed that the task can be reduced to multi-accuracy. Indeed,

assuming that the distinguishers have bounded complexity and can be learned efficiently, using previous algorithms in [14, 23, 5], we can design an efficient learner that constructs a generative model indistinguishable from f^* (Theorem 19). The model constructed this way is specified using a predictor $p : \{0,1\}^n \to [0,1]$, and the model represents the distribution of functions $f : \{0,1\}^n \to \{0,1\}$ where the function value $f(x)$ is distributed independently for every $x \in \{0,1\}^n$ according to the Bernoulli distribution $\mathsf{Ber}(p(x))$ with mean $p(x)$.

Learning generative models for binary functions becomes a more challenging task when we additionally enforce truthfulness requirements. A natural choice of truthfulness requirement is to preserve the support size of the function. Assuming that we know the support size $|\{x \in \{0,1\}^n : f^*(x) = 1\}|$ of f^* is k, we would like our model to only generate functions that also have support size k. We show how to build an efficient learner that can output such a truthful model which is also indistinguishable from the true function f^*:

▶ **Theorem 1** (Informal statement of Theorem 20). *Let $\mathcal{B}$ be the class of sample-access objects induced by binary functions $f : \{0,1\}^n \to \{0,1\}$ satisfying $|\mathsf{supp}(f)| = k$, where $\mathsf{supp}(f) := \{x \in \{0,1\}^n : f(x) = 1\}$. Let $\mathcal{D}$ be a class of distinguishers that is efficiently learnable. There exists an efficient (ε, δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$ and the learner always outputs an efficient implementation of a model M that is truthful w.r.t. $\mathcal{B}$.*

Note that the truthfulness requirement on the support size cannot be enforced simply using computationally-bounded distinguishers, because computing the support size of a function f exactly requires reading the values $f(x)$ for all the exponentially many inputs $x \in \{0,1\}^n$. Also, this truthfulness requirement cannot be satisfied directly by a generative model specified by a predictor p, where the function value $f(x)$ is distributed according to $\mathsf{Ber}(p(x))$ independently of the function values $f(x')$ of other individuals $x' \neq x$. To enforce a fixed support size, the function values of different individuals must coordinate in a global manner, requiring us to use new techniques. We create a binary tree with leaves corresponding to the function domain $\{0,1\}^n$, and following ideas in previous work such as [9], we assign support size budgets from the root to the leaves. However, the true function f^* is a single unknown object and is very different from the uniform distribution considered in [9], so there are no closed-from distributions (such as the binomial distributions used in [9]) that can guide us to distribute the support size budget from a node two its two children. Instead, we need to *estimate* how the budget should be divided, and this leads to accumulated error towards the leaves and forces us to stop before reaching the leaves. To efficiently propagate the budgets to the leaves, we solve a zero-sum game where player C chooses the budgets for the leaves and player D distinguishes them from the target. We show that if player D uses the multiplicative weights algorithm to minimize regret, we can create an indistinguishable and truthful model from the empirical distribution over the optimal responses from player C.

Learning a Function with Support Access

In the classic setting when learning a function $f : \{0,1\} \to \{0,1\}$, the learner receives random samples of form $(x, f(x))$ for a uniform $x \in \{0,1\}$. In this work, we also consider a function object in which we receive a random *positive entry*. That is, the learner receives random x's such that $f(x) = 1$. This type of random access is natural is certain situations, for example when we have information on the individuals that graduated some program, but not on those that did not.

▶ **Theorem 2** (Informal statement of Theorem 21). *Let $\alpha > 0$, and let $f : \{0,1\} \to \{0,1\}$ be a function such that $\Pr[f(x) = 1] = \alpha$. Let $\mathcal{D}$ be a collection of distinguishers, each $D \in \mathcal{D}$ associated with a set $S_D \subseteq [N]$ and accepts x if $x \in S_D$. If there exists a weak agnostic learner for $\mathcal{D}$, then there exists a learning algorithm L running in time $\mathrm{poly}(n)$, that receives random elements from the set $\{x|f(x) = 1\}$ and outputs a model M that is indistinguishable from f to all $D \in \mathcal{D}$.*

The theorem holds when there is a weak agnostic learner for the collection of distinguishers $\mathcal{D}$ under the distribution of a random support element (i.e. random x s.t. $f(x) = 1$). In the full version [17] we show that if a collection of distinguishers $\mathcal{D}$ has a weak agnostic learner over the standard sample access distribution, and the learner is a statistical query algorithm, then there is a learner for $\mathcal{D}$ also under the distribution of random support element.

The proof of the theorem is similar to the classic boosting argument, with an additional step that the learner performs of keeping the support size of the model approximately the same as support size of f. This step is necessary because under the distribution of a random support element, the boosting algorithm is only promised to work when the support sizes of f and the model are approximately equal.

Learning an object under the distribution of random support element is potentially very useful in the case of *sparse objects*. For a sparse function f, if we choose a uniform $x \in \{0,1\}$, then $f(x) = 0$ with high probability, and a learner cannot hope to learn anything non-trivial with random samples of form $(x, f(x))$. Unfortunately, the above theorem does not hold for sparse functions, but in the next part we show how this theorem can be used to learn different sparse objects - sparse graphs.

Learning Sparse Graphs Without Dense Subgraphs

Suppose $G = ([N], E)$ is a graph represented by the N^2 length string of its adjacency matrix. In this representation, receiving a random edge from G is equivalent to receiving a random support element from the function representing its adjacency matrix. Therefore, Theorem 2 implies that we can learn a model for G that is indistinguishable for a set of distinguishers $\mathcal{D}$ that have a weak agnostic learner. The theorem only holds for functions f with a constant fraction of 1 entries, which corresponds to a dense graph. What about sparse graphs?

Learning a sparse graph, or a sparse object in general, is a very challenging task because of the huge domain. The weak regularity lemma [7] has error that is proportional to N^2, which is too much in the case of sparse graphs (an empty graph is indistinguishable from a sparse graph with this error). Therefore in the setting of a sparse graphs it is more natural to require an ε error from the distinguisher under the distribution of receiving a random edge. Under this distribution, the error of the distinguishers scales with the number of edges. We show a learner for a specific class of sparse graphs, those that have no dense subgraphs. We note that a random sparse graph has no dense subgraphs, so many graphs have this property.

▶ **Theorem 3** (Informal statement of Theorem 25). *Let $G = ([N], E)$ be a sparse graph with no dense subgraphs. Let $\mathcal{D}$ be a collection of distinguishers, each $D \in \mathcal{D}$ associated with two sets $U_D, V_D \subseteq [N]$ and accepts an edge (u, v) if $u \in U_D, v \in V_D$. If there exists a weak agnostic learner for $\mathcal{D}$, then there exists a learning algorithm L running in time $\mathrm{polylog}(N)$, that receives random edges from G and outputs a model M that is indistinguishable from f to all $D \in \mathcal{D}$.*

The model that the learner outputs is *dense*, i.e. the model outputs graphs with $\Theta(N^2)$ many edges. This is done because of technical reasons – to allow us to use rejection sampling when training the model. This brings us to the question, is there a dense graph that is

indistinguishable from our sparse graph G? The answer to this question depends on G, and in the full version [17] we show that if a sparse graph G has a very dense subgraph, then there is no dense graph that is indistinguishable from G.

The proof of the theorem has two parts, in the first part we show that for every sparse graph G with no dense subgraphs, there exists a dense graph H that is indistinguishable from G. In this part of the proof we apply the strong regularity lemma for sparse graphs [26, 33] on G, and use the resulting partition to build the dense indistinguishable graph H. This part of the proof is existential, and we do know how to find H efficiently, as the strong regularity lemma does not have an efficient algorithm for finding the partition. It is not possible to use the weak regularity lemma or its variants [7], because its error is too large. In the second part of the proof, we reduce the learning G to learning H, and show that the resulted model M is indistinguishable from G to all distinguishers $D \in \mathcal{D}$.

Other Results on Learning Generative Models

In this work we also show indistinguishable models in several other settings

- Let $f : \{0,1\}^n \to \{0,1\}^n$ be a function. Learning such function is harder than learning a binary function because the large domain makes f a sparse object (when viewed as a graph for example it is an out-degree one graph). For such functions, we show that there exists a learner that given samples from the distribution $(x, f(x))$, outputs a model that is indistinguishable against the following set of distinguishers $\mathcal{D} = \{(S_D, j_D) | S_D \subset \{0,1\}^n, j_D \in [n]\}$ such that $D = 1 \iff x \in S_D, f(x)_j = 1$. This appears on Section 3.4.
- In Section 4.1 we apply the theorems for functions on the adjacency matrix of a dense graph $G = ([N], E)$. For a set of distinguishers $\mathcal{D}$ that have a weak agnostic leaner, we have an efficient learner that outputs an indistinguishable model when G is:
  1. A dense graph when the learner receives random adjacency matrix entries.
  2. A dense graph with a fixed total number of edges $m = \Theta(N^2)$.
  3. A directed graph with a fixed out-degree $m = \Theta(N)$.
  4. A dense graph when the learner receives random edges.
- For a directed graph $G = (\{0,1\}^n, E)$ with constant out-degree d, we can treat each of the d outgoing degrees as a function $f_i : \{0,1\}^n \to \{0,1\}^n$. For the same set of distinguishers that we can handle in the case of a length-preserving functions (the first item in this list), we provide an efficient learner.
- In the case of a uniform degree *undirected graph*, we provide in Section 4.3 a learning algorithm for an indistinghuishable model, albeit for a somewhat limited set of distinguishers.

Impossibility Results

As we discussed earlier, our goal of learning a generative model is closely related to the goal in [9] of implementing huge random objects, but a key difference is that we assume the groundtruth is a single unknown object B^*, whereas [9] considers a known uniform distribution of objects. This means that we need an additional learning procedure to collect information about B^*, and we show in Section 5 that our task of efficiently learning a generative model is only possible when the distinguisher class is efficiently learnable.

Besides the requirement of learning, our setting is more challenging than the setting in [9] in many other ways. We demonstrate this by another two impossibility results on fooling entry-access distinguishers and fooling stronger distinguishers than the model.

When we consider a pseudorandom function, f_s, the function is indistinguishable from the uniform distribution to distinguishers that have entry access to the function (allowed to ask for an arbitrary string x and get $f_s(x)$). Furthermore, while f_s is computable in a fixed polynomial time, the distinguishers can run in any polynomial time (and under reasonable assumptions, even exponential time). [9] and subsequent work inherit these two properties - indistinguishability to distinguishers that are computationally more complex than the models and have entry access to the model. In Section 5 we argue that neither of these properties is achievable in our setting.

For the impossibility of fooling distinguishers with entry access, in Theorem 27 we give the example of a class $\mathcal{D}$ that contains a distinguisher D_x for every input $x \in \{0,1\}^n$ which queries the function value $f(x)$ for a function f and outputs "accept" if and only if $f(x) = 1$. We argue that every model M that is indistinguishable from the true f^* for the set of distinguishers $\mathcal{D}$ has to be very close to f^*. Since the size of f^* is exponential and f^* is unknown, no efficient learner can output a model that is close to f^*. We also show an example, using an idea from [37], of a distinguisher and a true function f^*, such that the distinguisher can tell apart f^* from any model M with a low complexity compared to the distinguisher (Theorem 28). This highlights the fact that in our setting, the generative model and the learner constructing the model have to be computationally comparable or stronger than the distinguishers.

1.3 Related Work

As mentioned in Section 1.1, [9] introduced the problem of creating an indistinguishable implementation of a random object. [9] as well as follow-up works [31, 30] also present a collection of positive results for dense and sparse graphs or functions with a variety of truthfulness conditions and access models of the distinguishers.

The connection between generative models and indistinguishability has been manifested through the invention of generative adversarial networks (GANs) [10, 2]. Intuitively, a GAN is trained to imitate a distribution of objects (say images). The generator is trained in concert with a discriminator that could be interpreted as a distinguisher. Through a sequence of rounds, the generator is trained to fool the discriminator which is then trained to fail the generator. GANs highlight the connection between generative models and indistinguishability [21], but they do not naturally fall into our framework as they are more directly described in terms of indistinguishability of two distributions.

The connection between indistinguishability and learning theory has been established in many previous works (e.g. [37] applies the boosting technique from learning theory). More recently, in the context of algorithmic fairness, the relation between learning theory and indistinguishability has been dramatically expanded in the notions of multicalibration and outcome indistinguishability [14, 5], in applications to learning and statistical inference through the notions of omnipredictors and universal adapatability [12, 24, 18, 11, 25] and in the emergence of research uncovering intricate and exciting connections while studying the sample complexity of indistinguishability from a learning-theoretic perspective [20, 19].

It is possible to view our learning setting as a 2-players zero-sum game, between the learner and the distinguishers, in which the learner's goal is to output a model for an indistinguishable object and the distinguishers try to tell apart the input and the model. In this setting, there is a relation between min-max theorems and regularity-lemma theorems. Such theorems prove that it is possible to express a complex object f by a function of a few simpler objects $g_1, \dots g_t$ that, in our setting, represent the distinguishers [37, 38]. There have been works improving the parameters and also using such theorems for applications in cryptography [38, 22, 34, 35, 36, 4]. In this work, our setting is slightly different, as we assume that the object f is complex and unknown, and the learner has to learn it. The proof

of Theorem 3 has an intermediate step that is existential and has a similar structure to a weak regularity lemma theorems, but since the required error there is too small, we derive it from the sparse strong regularity lemma.

▶ Note 4. This is an abridged version of the paper. We refer the readers to the full version [17] for proofs and other contents that are omitted in this version.

2 Preliminaries

Throughout the paper, we are interested in learning objects such as functions and graphs, and we are particularly interested when these objects have exponential sizes (e.g. functions with exponentially large domains and graphs with exponentially many vertices and edges). We typically use B to denote an object, and use $\mathcal{B}$ to denote a class of objects. We view an object B as a function $B : Q \to \Delta_A$ that maps a query $q \in Q$ to a distribution $B(q)$ over answers in A.

2.1 Functions

When the object is a function $f : X \to Y$, we consider three access types. For sample access, B returns a random pair $(x, f(x))$. For support access, it returns a random x such that $f(x) = 1$. For entry access, upon querying x, B returns $f(x)$.

▶ **Definition 5** (Function-induced sample-access object). *Let $f : X \to Y$ be a function and let $B : Q \to \Delta_A$ be an object. We say B is the* sample-access *object* induced *by f if $Q = \{\perp\}, A = X \times Y$, and $B(\perp)$ is the distribution of $(x, f(x)) \in A$ where x is drawn uniformly from X.*

▶ **Definition 6** (Function-induced support-access object). *Let $f : X \to \{0, 1\}$ be a binary function. We define the support of f to be $\mathsf{supp}(f) := \{x \in X : f(x) = 1\}$. Let $B : Q \to \Delta_A$ be an object. Assuming $\mathsf{supp}(f) \neq \emptyset$, we say B is the* support-access *object induced by f if $Q = \{\perp\}, A = X$, and $B(\perp)$ is the uniform distribution over $\mathsf{supp}(f) \subseteq X$.*

▶ **Definition 7** (Function-induced entry-access object). *Let $f : X \to Y$ be a function and let $B : Q \to \Delta_A$ be an object. We say B is the* entry-access *object* induced *by f if $Q = X, A = Y$, and for every $q \in Q$, $B(q)$ is the singleton distribution such that $a \sim B(q)$ equals to $f(q)$ deterministically.*

In this paper, we show positive results for learning generative models of functions with sample access and support access (Section 3) whereas we show impossibility results for entry access (Section 5). This separation is mainly because entry access makes the distinguishers stronger and thus makes indistinguishability harder to achieve (see Definitions 10 and 11 below).

2.2 Graphs

For a graph $G = (V, E)$ where we assume V has exponential size, we define two access types, sample-access which corresponds to a random adjacency matrix entry, and support-access which corresponds to a random edge in the graph.

▶ **Definition 8** (Graph-induced sample-access object). *Let $G = (V, E)$ be a directed or undirected graph and let $B : Q \to \Delta_A$ be an object. We say B is the* sample-access *object induced by G if $Q = \{\perp\}, A = V \times V \times \{0, 1\}$, and $B(\perp)$ is the distribution of $(u, v, y) \in A$ where (u, v) is drawn uniformly from $V \times V$, $y = 1$ if $(u, v) \in E$ and $y = 0$ otherwise.*

▶ **Definition 9** (Graph-induced support-access object). *Let $G = (V, E)$ be a directed or undirected graph and let $B : Q \to \Delta_A$ be an object. Assuming $E \neq \emptyset$, we say B is the* support-access *object induced by G if $Q = \{\perp\}, A = V \times V$, and $B(\perp)$ is the uniform distribution over $E \subseteq V \times V$.*

2.3 Indistinguishability

Each learner we design in this paper has access to a ground-truth object B^*, and it aims to output an object B that is *indistinguishable* from B^*. In many cases, the learner does not just output a single object B, but a distribution over objects, and we refer to such distributions as *models*. Below we formally define the notion of indistinguishability.

▶ **Definition 10** (Distinguisher). *A* distinguisher *D is an algorithm that when given access to an object $B : Q \to \Delta_A$, outputs* "accept" *or* "reject"*. That is, the distinguisher is allowed to make queries $q \in Q$ to the model, and for each query q the distinguisher receives an answer $a \in A$ drawn independently from $B(q) \in \Delta_A$. We allow the distinguisher D itself to be randomized, and we use random variable D^B to denote the output of the distinguisher D in* {"accept", "reject"} *when given access to B.*

▶ **Definition 11** (Indistinguishability). *Let $B^* : Q \to \Delta_A$ be an object, and let model M be a distribution over objects $B : Q \to \Delta_A$. We say model M is* ε-indistinguishable *from object B^* w.r.t. a distinguisher D if*

$$|\Pr[D^{B^*} = \text{“accept”}] - \mathbb{E}_{B\sim M}[\Pr[D^B = \text{“accept”}]]| \leq \varepsilon. \tag{1}$$

We say model M is ε-indistinguishable from object B^* w.r.t. a class $\mathcal{D}$ of distinguishers if (1) *holds for every $D \in \mathcal{D}$.*

2.4 Truthfulness

In addition to indistinguishability, another desirable property of a model is *truthfulness* introduced in [9]. Truthfulness requires every object generated from the model to satisfy a certain (usually global) property which we formalize using an object class $\mathcal{B}$:

▶ **Definition 12** (Truthfulness). *We say a model M is* truthful *w.r.t. an object class $\mathcal{B}$ if*

$$\Pr_{B\sim M}[B \in \mathcal{B}] = 1.$$

2.5 Implementations

Our goal is to design efficient learners, and thus we cannot expect the learner to output a model M explicitly, especially when the objects drawn from M are huge. Instead, our learner outputs an efficient *implementation* of a model, defined as follows.

▶ **Definition 13** (Ordinary Implementation). *For $\ell \in \mathbb{Z}_{\geq 0}$, let T be a randomized algorithm that takes $(r, q) \in \{0,1\}^\ell \times Q$ as input, and outputs $T(r, q) \in A$. We say T is an* ordinary implementation *of a model M with* seed length *ℓ if for every* seed *$r \in \{0,1\}^\ell$ there exists an object $B_r : Q \to \Delta_A$ such that*

1. *for every $q \in Q$, $T(r, q)$ is distributed according to $B_r(q)$, where the randomness in $T(r, q)$ comes from the internal randomness in algorithm T;*
2. *B_r is distributed according to M when r is drawn uniformly from $\{0,1\}^\ell$.*

While our goal is to output an ordinary implementation with a polynomial-length seed, following the approach in [9], it is more convenient to first build implementations using a random oracle and then transform the implementation to an ordinary one using Lemma 15.

▶ **Definition 14** (Random-Oracle Implementation). *Let T be a randomized algorithm that takes a function $r : \{0,1\}^* \to \{0,1\}$ as an oracle. On an input $q \in Q$, the algorithm T outputs $T^r(q) \in A$. We say T is a* random-oracle implementation *of a model M if for every $r : \{0,1\}^* \to \{0,1\}$ there exists an object $B_r : Q \to \Delta_A$ such that*

1. *for every $q \in Q$, $T^r(q)$ is distributed according to $B_r(q)$, where the randomness in $T^r(q)$ comes from the internal randomness in algorithm T;*
2. *B_r is distributed according to M when r is a uniformly random function from $\{0,1\}^*$ to $\{0,1\}$.*

▶ **Lemma 15** (Theorem 2.9 in [9]). *Suppose that one-way functions exist. There exists an algorithm H with the following properties. Let $\mathcal{D}$ be a class of distinguishers where each $D \in \mathcal{D}$ is a circuit of size at most W for some $W \geq 1$. Let T be a random-oracle implementation of a model M with circuit complexity at most W. Given W, T and an arbitrary $\varepsilon \in (0,1)$ as input, the algorithm H runs in time $\mathrm{poly}(W, 1/\varepsilon)$ and outputs an ordinary implementation T' of a model M' where T' has seed length and circuit complexity both being $\mathrm{poly}(W, 1/\varepsilon)$, and*

$$|\mathbb{E}_{B' \sim M'} \Pr[D^{B'} = \text{“accept”}] - \mathbb{E}_{B \sim M} \Pr[D^{B} = \text{“accept”}]| \leq \varepsilon \quad \text{for every } D \in \mathcal{D}.$$

Moreover, if M is truthful w.r.t. an object class $\mathcal{B}$, then M' is also truthful w.r.t. $\mathcal{B}$.

Lemma 15 can be proved by using a pseudorandom function to emulate the random oracle.

2.6 Learning

We describe the learners we aim to design in the definition below.

▶ **Definition 16** (Learner). *Let $\mathcal{B}$ be a class of objects $B : Q \to \Delta_A$ and $\mathcal{D}$ be a class of distinguishers. An (ε, δ)-*learner *L for the class $\mathcal{B}$ w.r.t. $\mathcal{D}$ is an algorithm with the following properties. For any $B^* \in \mathcal{B}$, given access to B^*, the learner outputs an implementation T of a model M such that with probability at least $1 - \delta$, M is ε-indistinguishable from B^* w.r.t. $\mathcal{D}$.*

2.7 Other Notations

For $v \in \mathbb{R}$, we define $\mathsf{cap}(v)$ by capping its value into $[0,1]$, i.e.,

$$\mathsf{cap}(v) = \begin{cases} v, & \text{if } 0 \leq v \leq 1; \\ 1, & \text{if } v > 1; \\ 0, & \text{if } v < 0. \end{cases}$$

Given a list of values $(v_1, \ldots, v_t)$ we define $\mathsf{Lcap}(v_1, \ldots, v_t)$ by summing over the list and capping the value to $[0,1]$ in every iteration. We formally define it recursively:

$$\begin{aligned} \mathsf{Lcap}(v_1) &= \mathsf{cap}(v_1), \\ \mathsf{Lcap}(v_1, \ldots, v_t) &= \mathsf{cap}(\mathsf{Lcap}(v_1, \ldots, v_{t-1}) + v_t). \end{aligned} \tag{2}$$

3 Learning Functions with Exponentially Large Domains

The goal of this section is to efficiently learn a generative model that is indistinguishable from a target function $f^* : X \to Y$ to a class $\mathcal{D}$ of distinguishers. We allow the domain X of the function to have exponential size $N := |X|$, and require our learner to run in time $\text{polylog}(N)$. This means that the learner cannot read the entire function f^*, and can only access it via random sample. Throughout the paper, our learners output an efficient random-oracle implementation T of a model M, which can be turned in to an efficient ordinary implementation by Lemma 15.

3.1 Learning Sample-Access Binary Functions

We start by studying the case where the target object B^* is the sample-access object induced by a binary function $f^* : X \to \{0,1\}$ (Definition 5). We assume that every distinguisher $D \in \mathcal{D}$ satisfies the following: when given access to a sample-access object B induced by a function $f : X \to \{0,1\}$, the distinguisher asks a single query $\perp$, receives an answer $a = (x, y) \sim B(\perp)$, and outputs $D(x, y) \in \{\text{“accept”}, \text{“reject”}\}$. We allow the distinguisher itself to be randomized, and each distinguisher D defines a function $g_D : X \to [-1, 1]$ such that

$$g_D(x) = \Pr[D(x,1) = \text{“accept”}] - \Pr[D(x,0) = \text{“accept”}] \quad \text{for every } x \in X.$$

We use the following claim to relate a distinguisher $D \in \mathcal{D}$ to the function $g_D : X \to [-1, 1]$:

▷ Claim 17. For every distinguisher $D \in \mathcal{D}$, the following equation holds for every $x \in X$ and $y_1, y_2 \in \{0,1\}$:

$$\Pr[D(x,y_1) = \text{“accept”}] - \Pr[D(x,y_2) = \text{“accept”}] = y_1 g_D(x) - y_2 g_D(x).$$

The claim can be easily proved by considering the four possible choices of $(y_1, y_2) \in \{0,1\} \times \{0,1\}$.

As we show later in Section 5, it is necessary to impose certain learnability assumptions on the distinguishers. To that end, we assume that there is an *auditor* for the function class $\mathcal{G} := \{g_D : D \in \mathcal{D}\}$, defined as follows:

▶ **Definition 18** (Auditor). *Let $\mathcal{D}$ and $\mathcal{G}$ be defined as above. We say an algorithm Λ is an $(\varepsilon, \gamma, \delta)$-auditor for $\mathcal{G}$ if it satisfies the following property. Given access to a sample-access object B^* induced by a function $f^* : X \to \{0,1\}$ and taking a predictor $p : X \to [0,1]$ as an oracle, if there exists $g \in \mathcal{G}$ such that*

$$|\,\mathbb{E}[f^*(x)g(x)] - \mathbb{E}[p(x)g(x)]| \geq \varepsilon,$$

then Λ outputs $\hat{g} : X \to [-1,1]$ satisfying the following with probability at least $1 - \delta$:

$$\mathbb{E}[f^*(x)\hat{g}(x)] - \mathbb{E}[p(x)\hat{g}(x)] \geq \gamma.$$

The auditor defined above can be viewed as a weak agnostic learner for the class $\mathcal{G}$. When the domain X is $\{0,1\}^n$ with size $N = 2^n$, many classes allow efficient auditors that run in time $\text{poly}(n) = \text{polylog}(N)$. Using an auditor for $\mathcal{G}$, we prove the following theorem:

▶ **Theorem 19.** *Let the distinguisher class $\mathcal{D}$ and the function class $\mathcal{G}$ be defined above. Let $\varepsilon, \gamma, \delta, \delta' \in (0, 1/2)$ be parameters satisfying $\gamma \leq \varepsilon$ and $\delta' \leq c\delta\gamma^2$ for a sufficiently small absolute constant $c > 0$. Let $\mathcal{B}$ be the class of sample-access objects induced by binary functions $f : X \to \{0,1\}$. Let Λ be an $(\varepsilon, \gamma, \delta')$-auditor for $\mathcal{G}$ (Definition 18). Then there*

exists an (ε, δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$. Moreover, if the auditor Λ runs in time at most W_1 and always outputs a function with circuit size at most W_2, then the learner L runs in time $\mathrm{poly}(\gamma^{-1}, \log(\delta^{-1}), W_1)$ *and always outputs implementations with circuit complexity* $\tilde{O}(\gamma^{-2} W_2)$.

We prove the theorem by applying results from algorithmic fairness [37, 14, 23], see the full version for more details.

3.2 Truthful Learning That Preserves Support Size

Some desirable properties of a generative model are global and cannot be enforced using computationally bounded distinguishers alone. This motivated [9] to introduce the notion of *truthfulness* that ensures such global properties beyond indistinguishability. Here we focus on a natural global property of a binary function $f : X \to \{0,1\}$: the size of its support $\mathsf{supp}(f) := \{x \in X : f(x) = 1\}$. The model M we create in Section 3.1 using the multiaccurate predictor p may generate functions f with support size different from the target function f^*. Indeed, even if $\sum_{x \in X} p(x) = |\mathsf{supp}(f^*)|$, a random function f with each entry $f(x)$ independently drawn from $\mathsf{Ber}(p(x))$ is not guaranteed to satisfy $|\mathsf{supp}(f)| = |\mathsf{supp}(f^*)|$. Now we show an efficient learner that outputs truthful models that preserve the support size of the generated functions.

An overview of the proof appears in the introduction, and the proof appears in the full version [17].

▶ **Theorem 20.** *Let the distinguisher class $\mathcal{D}$ and the function class $\mathcal{G}$ be defined as in Section 3.1. Let $\varepsilon, \gamma, \delta, \delta' \in (0, 1/2)$ be parameters satisfying $\gamma \leq \varepsilon$ and $\delta' \leq c\delta\gamma^2$ for a sufficiently small absolute constant $c > 0$. Let $\mathcal{B}$ be the class of sample-access objects induced by binary functions $f : X \to \{0,1\}$ satisfying $|\mathsf{supp}(f)| = k$, where $X = \{0,1\}^n$. Let Λ be an $(\varepsilon, \gamma, \delta')$-auditor for $\mathcal{G}$ (Definition 18). Then there exists an (ε, δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$ and the learner always outputs a random-oracle implementation of a model M that is truthful w.r.t. $\mathcal{B}$. Moreover, if the auditor Λ runs in time at most W_1 and always outputs a function with circuit size at most W_2, then the learner L runs in time* $\mathrm{poly}(n, \gamma^{-1}, \log(\delta^{-1}), W_1)$ *and always outputs an implementation with circuit complexity* $\mathrm{poly}(n, \gamma^{-1}, W_2)$.

3.3 Learning Support-Access Binary Functions

In the previous sections we showed how to learn and construct an implementation of an indistinguishable model for a function induced sample-access object B^*, i.e. there exists a function $f^* : X \to \{0,1\}$, and the distinguishers and learner both receives random samples of form $(x, f^*(x))$. In this section, we learn a support-access object induced by a function see Definition 6. For a binary function $f^* : X \to \{0,1\}$, the support-access object B^* induced by f^* outputs random samples from the set $\{x | f^*(x) = 1\}$. We show how to construct an efficient implementation of a model that is indistinguishable from B^*.

Distinguishers: Let $\mathcal{D}$ be a set of distinguishers, such that each $D \in \mathcal{D}$ has an associated subset $S_D \subset X$. Given access to a sample x from a object B,the distinguisher accepts if $x \in S_D$. We assume that for every $D \in \mathcal{D}$, the set S_D has known size and an efficient description, that on input x answers if $x \in S_D$.

Auditor: Let $\mathcal{D}$ be defined as above. We say that an algorithm $\Lambda^{B^*,p}$ is an $(\varepsilon, \gamma, \delta)$ auditor for the collection of sets $\mathcal{S} = \{S_D | D \in \mathcal{D}\}$ if it has the following properties. Given access to a function-induced support access-object B^* and query access to a predictor $p : X \to [0,1]$. If there exists $S \in \mathcal{S}$ and $b \in \{-1, 1\}$ such that

$b\left(\Pr_{x\sim B^*(\perp)}[x \in S] - \Pr_{x\sim p}[x \in S]\right) > \varepsilon$. Then the auditor returns a set S' such that with probability $1-\delta$, $b\left(\Pr_{x\sim B^*(\perp)}[x \in S'] - \Pr_{x\sim p}[x \in S']\right) > \gamma$. Where $x \sim p$ is the distribution generated from the predictor p, i.e. $\Pr_{x\sim p}[x = x'] = p(x')/\sum_{x''\in X} p(x'')$.

▶ **Theorem 21.** *Let $\alpha \in [0,1]$ be a parameter, and let $\mathcal{B}$ be a collection of support access object induced by binary functions, such that $\forall B \in \mathcal{B}, \mathbb{E}_{x\in X}[f_B(x)] = \alpha$. Let $\mathcal{D}$ be a collection of distinguishers as described above.*

Let $\varepsilon, \gamma, \delta', \delta''$ be parameters such that $\delta' \leq c\delta\gamma^2\alpha^{-2}$ for a sufficiently small constant c. Let Λ be an $(\varepsilon, \gamma, \delta')$ auditor Λ for $\mathcal{D}$. Then there exists a $(2\varepsilon, \delta)$-learner L for $\mathcal{B}$ with respect to the distinguisher class $\mathcal{D}$. The learner L runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_1, W_2)$, where W_1 is the running time of the auditor Λ and W_2 the circuit complexity of its output. The implementation T that the learner outputs runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_2)$

The learner L in the theorem above has access to an auditor Λ that can audit support-access objects. In the full version [17] we discuss under which conditions such auditor exists. The learning algorithm is similar to the classic boosting algorithm, with an additional step of keeping the expected value of the model to be approximately α. We note that if the function is sparse, i.e. α is very small, then the algorithm is no longer efficient.

3.4 Learning Bit-String Functions

In this section we are interested in learning a function $f : \{0,1\}^n \to \{0,1\}^n$. This is a harder than learning a binary function, because the range of the function is very large. Therefor we only learn an indistinguishable model with respect to a very limited set of distinguishers, with a product structure. In this setting, the sampling distribution is a pair $(x, f(x))$ for a random input x.

Distinguishers: Let $\mathcal{D}$, such that each distinguisher $D \in \mathcal{D}$ has an set $S_D \subset \{0,1\}^n$ and a coordinate $j \in [n]$. The distinguisher D accept a sample $(x, f(x))$ if $x \in S$ and $f(x)_j = 1$.

Auditor: Let $\mathcal{D}$ be defined as above. We say that an algorithm $\Lambda^{B^*,p}$ is an $(\varepsilon, \gamma, \delta)$ auditor for the collection of sets $\mathcal{S}$ if it has the following properties. Given access to a function-induced support access-object B^* and query access to a set of n predictors $p_1, \ldots p_n$, such that $p_j : \{0,1\}^n \to [0,1]$. If there exists $S \in \mathcal{S}$ and $j \in [n]$ such that $b\left(\Pr_x[x \in S, f(x)_j = 1] - \mathbb{E}_x[p_j(x)\cdot \mathbf{1}(x \in S)]\right) > \varepsilon$. Then the auditor returns a set $S' \subseteq \{0,1\}^n$ and $j \in [n]$ such that $b\left(\Pr_x[x \in S', f(x)_j = 1] - \mathbb{E}_x[p_j(x)\cdot \mathbf{1}(x \in S')]\right) > \gamma$ with probability $1-\delta$.

▶ **Theorem 22.** *Let $\mathcal{B}$ be a collection of support access object induced by functions $f : \{0,1\}^n \to \{0,1\}^n$. Let $\mathcal{D}$ be a collection of distinguishers as described above.*

Let $\varepsilon, \gamma, \delta', \delta''$ be parameters such that $\delta' \leq c\delta\gamma^2 n^{-1}$ for a sufficiently small constant c. Let Λ be an $(\varepsilon, \gamma, \delta')$ auditor for $\mathcal{D}$. Then there exists a $(2\varepsilon, \delta)$-learner L to $\mathcal{B}$ with respect to the distinguisher class $\mathcal{D}$ and the learner L runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_1, W_2)$, where W_1 is the running time of the auditor Λ and W_2 the circuit complexity of its output. The implementation T that the learner outputs runs in time $\mathrm{poly}(\gamma^{-1}\log(\delta^{-1})\alpha^{-1}, W_2)$

4 Learning Exponential-Size Graphs

4.1 Learning Dense Graphs

The most basic setting for graphs is the dense model, where the graph-induced sample-access object B induced by a graph $G = ([N], E)$ can be thought of as getting a random entry $((u,v), b)$ for $u, v \in [N], b \in \{0,1\}$ from the adjacency matrix of G (see Definition 8). We can

think of the adjacency matrix of the graph as a function, where the graph imposes some extra structure on the distinguishers. Therefore, some of the results from Section 3 follow directly.

Distinguishers: Let $\mathcal{D}$ be a collection of distinguishers. Each distinguisher has two sets of vertices $U_D, V_D \subset [N]$. When getting a random entry $((u,v),b)$ from the graph-induced sample-access object B^*, it accepts if $u \in U_D$, $v \in V_D$ and $b = 1$.

Auditors: An algorithm Λ is an $(\varepsilon,\gamma,\delta)$-auditor for a collection of tuples $\mathcal{S}$ containing pairs $(U,V) \subset [N] \times [N]$ if it satisfies the following. The auditor received query access to a predictor $p : [N] \times [N] \to [0,1]$ and to a graph-induced sample-access object B^* induced by a graph $G^* = ([N], E^*)$. If there exists $(U,V) \in \mathcal{S}$ such that

$$b \cdot \left(\Pr_{(u,v)\in[N]\times[N]}[u \in U, v \in V, (u,v) \in E^*] - \mathbb{E}_{(u,v)\in[N]\times[N]}[\mathbf{1}(u \in U, v \in V)p(u,v)] \right) \geq \varepsilon.$$

Then with probability $(1-\delta)$, it outputs U', V' such that

$$b \cdot \left(\Pr_{(u,v)\in[N]\times[N]}[u \in U', v \in V', (u,v) \in E^*] - \mathbb{E}_{(u,v)\in[N]\times[N]}[\mathbf{1}(u \in U', v \in V')p(u,v)] \right) \geq \gamma.$$

The setting of a dense graph can be derived directly from the function theorem, by applying it on the adjacency matrix of the graph.

▶ **Corollary 23** (Corollary of Theorem 19). *Let the distinguisher class $\mathcal{D}$ be defined above. Let $\varepsilon, \gamma, \delta, \delta' > 0$ be parameters satisfying $\delta' \leq c\delta\gamma^2$ for a sufficiently small absolute constant $c > 0$. Let $\mathcal{B}$ be the class of sample-access objects induced by graphs over vertex set $[N]$. Let Λ be an $(\varepsilon,\gamma,\delta')$-auditor for $\mathcal{S} = \{(U_D, V_D) | D \in \mathcal{D}\}$. Then there exists an (ε,δ)-learner L for $\mathcal{B}$ w.r.t. $\mathcal{D}$. Moreover, if the auditor Λ runs in time at most W_1 and always outputs a function with circuit size at most W_2, then the learner L runs in time $\mathrm{poly}(\gamma^{-1}, \log(\delta^{-1}), W_1)$ and always outputs implementations with circuit complexity $O(\gamma^{-2}W_2)$.*

The same holds also for learning a graph with a fixed number of edges, by applying Theorem 20. Similarly, we can derive a theorem on directed graphs by applying Theorem 19 or Section 3.2 on the directed graph adjacency matrix. It is also possible to generate a directed graph with a fixed out-degree by applying Section 3.2 for every vertex individually.

4.2 Learning Sparse Graphs Without Dense Subgraphs

For a sparse graph, a sample-access graph object is not useful, because a random entry in the adjacency matrix of the graph is nearly always 0. We study graph-induced support-access objects (Definition 9), which corresponds to an object B^* induced by a sparse graph $G = ([N], E)$, $B(\perp)$ that outputs a random edge in the graph $(u,v) \in E$.

Applying Theorem 21 implies a corollary for support-access graphs, but the theorem is only efficient for dense graphs. In this section we show how to create *a dense model for a sparse graph*, as long as the sparse graph does not have a subgraph which is too dense. We do so by using the strong regularity lemma for sparse graphs [26, 33].

Distinguishers: Let $\mathcal{D}$ be a collection of distinguishers. Each distinguisher has two sets of vertices $U_D, V_D \subset [N]$. A distinguisher D on input (u,v) accepts if $u \in U_D$ and $v \in V_D$.

Auditors: An algorithm Λ is an $(\varepsilon,\varepsilon',\delta)$-auditor for a collection pairs of sets $\mathcal{S}$, such that $(U,V) \in \mathcal{S}$, $U, V \subset [N]$ if it satisfies the following. The auditor received query access to a predictor $p : [N] \times [N] \to [0,1]$ and access to a graph-induced support-access object

B^* representing a graph $G^* = ([N], E^*)$. If there exists a pair of sets $(U, V) \subset \mathcal{S}$ and a bit b such that $b \cdot \left(\Pr_{(u,v)\sim B^*(\perp)}[u \in U, v \in V] - \Pr_{(u,v)\sim p}[u \in U, v \in V]\right) \geq \varepsilon$, Then Λ outputs sets $U', V' \subset [N]$ such that

$$b \cdot \left(\Pr_{(u,v)\sim B^*(\perp)}[u \in U', v \in V'] - \Pr_{(u,v)\sim p}[u \in U', v \in V'] \right) \geq \varepsilon'.$$

The distribution $(u, v) \sim p$ is defined by the predictor p, i.e. for all $u, v \in [N]$ we have $\Pr_{(u',v')\sim p}[u' = u, v' = v] = p(u, v)/\sum_{u'',v''\in[N]} p(u'', v'')$.

Graph Notations and Definitions

For a graph $G = ([N], E)$ and $U, V \subset [N]$, we define $E_G(U, V) = \{(u, v) \in E | u \in U, v \in V\}$ to be the set of edges between U, V in G. We denote by $\rho_G(U, V)$ the edge density between U, V in G, $\rho_G(U, V) = \frac{|E_G(U,V)|}{|U||V|}$. We denote by $\rho_G = \rho_G([N], [N])$ the edge density of the graph. We use the definition of upper-uniform graphs from [26, 33] with a small additional requirement also for small sets.

▶ **Definition 24** (Upper-uniform graphs). *A graph $G = ([N], E)$ is (η, γ)-upper uniform, if for every two disjoint sets $U, V \subset [N]$, with $\min\{|U|, |V|\} \geq \eta N$ we have that $\rho_G(U, V) \leq \gamma\rho_G$, and for U, V such that $\min\{|U|, |V|\} < \eta N$, we have that $|E(U, V)| \leq \gamma\eta\rho_G N^2$.*

We remark that a random sparse graph is upper-uniform for constants η, γ with high probability.

▶ **Theorem 25.** *For every parameter $\gamma, \varepsilon, \varepsilon', \lambda'\lambda'' > 0$, such that $\lambda' \leq c\lambda\varepsilon'^2$ for a sufficiently large constant c. Then there exists $\eta \in [0, 1]$ such that the following holds. Let $\mathcal{B}$ be a collection of graph-induced support access objects, such that for each $B^* \in \mathcal{B}$, the graph it represents G_{B^*} is (η, γ)-upper-uniform.*

Let $\mathcal{D}$ be a collection of distinguishers. If there exists an $(\varepsilon, \varepsilon', \delta')$-auditor Λ for the collection of sets $\mathcal{C} = \{(U_D, V_D) | D \in \mathcal{D}\}$, then there exists an (ε, δ'')-learning algorithm L for all $B^* \in \mathcal{B}$ with respect to $\mathcal{D}$.

The proof of the theorem appears in the full version [17]. In addition, we show how this theorem can be combined with Theorem 20 to create a sparse uniform out-degree graph.

4.3 Learning Uniform Degree Graphs

Suppose we are interested in generating a truthful model for a uniform degree d graph. That is, we want that all graph in our model has a uniform degree d. In previous sections we discussed directed graphs with uniform out-degree. For undirected graphs, in [9] the authors show a construction of a graph indistinguishable from random, by applying a random permutation on a large girth expander. In this work we restrict the set of distinguishers to those that can be described by a partition, and create a model by learning the densities of the edges between each part in the partition and permuting the edges.

Distinguishers: Then the set of distinguishers $\mathcal{D}$ contains distinguishers D with sets (U_D, V_D) such that $U, V \in \mathcal{U}$. Every distinguisher D accepts an edge (u, v) if $u \in U, v \in V$. Let $\mathcal{U} = \{U \subset [N] | \exists D \text{ s.t. } U = U_D \text{ or } U = V_D\}$. We assume that $\mathcal{U}$ is a partition with t parts, and that $|U_J|$ is linear in N.

▶ **Lemma 26.** *Let $\mathcal{B}$ be a collection of graph-induced support-access objects, such that for all $B^* \in \mathcal{B}$, the graph G_{B^*} has a uniform degree d. Let $\mathcal{D}$ be the distinguishers class defined above. Then for every constant ε there exists an $(\varepsilon, \delta)-$ learning algorithm L for the class $\mathcal{B}$ with respect to $\mathcal{D}$. The algorithm runs in time* $\text{poly}(1/\varepsilon, \log(1/\delta))$.

5 Impossibilities

A main difference in our work from [9] is in the target distribution/object we aim to be indistinguishable from. In [9], the target distribution is fixed and uniform over many objects, whereas in our setup the target is a single object which is initially unknown, and a learner is needed to access the target object to make it possible to create an indistinguishable model. This difference makes our setup challenging, and below we show example tasks that are impossible to achieve in our setup because of this difference.

5.1 Fooling Distinguishers with Entry-Access is Hard

In [9], the distinguishers can query for specific entries of an object. Such distinguishers can be impossible to fool in our setup. For example, suppose the target object B^* is the *entry-access* object induced by a function $f^* : X \to \{0,1\}$ (Definition 7), and suppose our learner aims to output a model M of entry-access objects B induced by functions $f : X \to \{0,1\}$. For every $x \in X$, suppose there is a distinguisher that queries for the value of $f(x)$ and outputs "accept" if and only if $f(x) = 1$. To fool these distinguishers, we have to learn the target function f^* exactly, which is clearly impossible if the domain X has exponential size and the learner can only make polynomially many queries.

▶ **Theorem 27.** *Let X be a non-empty finite set. Let $\mathcal{B}$ be the class of entry-access objects induced by all functions $f : X \to \{0,1\}$. Let $\mathcal{D}$ be the class of distinguishers D_x for every $x \in X$ where given an object B, the distinguisher D_x outputs* "accept" *if and only if the answer $a \sim B(x)$ is equal to 1. Let L be an (ε, δ)-learner for the class $\mathcal{B}$ w.r.t. $\mathcal{D}$ for $\varepsilon, \delta < 1/2$. Then L needs to query every input $x \in X$ in the worst case.*

5.2 Learned Model Needs to be Stronger than Distinguishers

The model learned in [9] can fool distinguishers with significantly larger circuit complexity than the model itself. Below we show that this can become impossible in our setup where the target is a single object.

▶ **Theorem 28** (Remark 1.6 in [37])**.** *Let $n, W > 1$ be positive integers satisfying $W \log W \leq 2^n/C$ for a sufficiently large absolute constant $C > 0$. There exists a sample-access object B^* induced by a function $f^* : \{0,1\}^n \to \{0,1\}$ and a distinguisher D with circuit complexity $\tilde{O}(nW)$ such that for any model M with circuit complexity at most W, it holds that*

$$|\Pr[D^{B^*} = \text{"accept"}] - \mathbb{E}_{B\sim M}[\Pr[D^B = \text{"accept"}]]| > 1/3. \tag{3}$$

5.3 The Distinguisher Class Needs to be Learnable

Since the target distribution in [9] is fixed, no learning is needed in order to produce an indistinguishable model. In our setup, the learning task is usually performed using an auditor, which can be viewed as a weak agnostic learner for the class of distinguishers. A natural question is whether we can still achieve indistinguishability if such a weak agnostic learner does not exist. Previous works [14, 11] have shown negative answers to this question for certain notions of indistinguishability (such as calibrated multiaccuracy) by showing that these notions imply (strong) agnostic learning for the distinguisher class. The indistinguishability notion we use for generative models is closer to multiaccuracy, and below we show that efficiently achieving this notion requires the distinguisher class to be efficiently realizably learnable. For a true function $f^* : X \to \{0,1\}$, multiaccuracy requires a predictor $p : X \to [0,1]$ to satisfy

$$|\,\mathbb{E}[(f^*(x) - p(x))g(x)]| \leq \varepsilon \tag{4}$$

for every function g in a class $\mathcal{G}$. Now consider the case where $\mathcal{G}$ consists of functions $g : X \to \{-1, 1\}$. For an arbitrary $g^* \in \mathcal{G}$, suppose the true function f^* satisfies $f^*(x) = 1$ if $g^*(x) = 1$ and $f^*(x) = 0$ if $g^*(x) = -1$. Then (4) implies

$$\mathbb{E}\,|f^*(x) - p(x)| \leq \varepsilon. \tag{5}$$

Now we define $\hat{g}(x) = 1$ if $p(x) \geq 1/2$, and define $\hat{g}(x) = -1$ if $p(x) < 1/2$. It is easy to check that if $\hat{g}(x) \neq g^*(x)$ for some $x \in X$, then $|f^*(x) - p(x)| \geq 1/2$, and thus (5) implies the following realizable learning guarantee for the class $\mathcal{G}$:

$$\Pr[\hat{g}(x) \neq g^*(x)] \leq 2\varepsilon.$$

References

1 Edoardo M Airoldi, David M Blei, Stephen E Fienberg, and Eric P Xing. Mixed membership stochastic blockmodels. *Journal of machine learning research: JMLR*, 9:1981–2014, 2008.

2 Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 224–232. PMLR, 06–11 August 2017. URL: https://proceedings.mlr.press/v70/arora17a.html.

3 Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudorandom bits. *SIAM Journal on Computing*, 13(4):850–864, 1984. doi:10.1137/0213053.

4 Yi-Hsiu Chen, Kai-Min Chung, and Jyun-Jie Liao. On the complexity of simulating auxiliary input. In *Advances in cryptology—EUROCRYPT 2018. Part III*, volume 10822 of *Lecture Notes in Comput. Sci.*, pages 371–390. Springer, Cham, 2018. doi:10.1007/978-3-319-78372-7_12.

5 Cynthia Dwork, Michael P Kim, Omer Reingold, Guy N Rothblum, and Gal Yona. Outcome indistinguishability. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1095–1108, 2021.

6 Cynthia Dwork, Daniel Lee, Huijia Lin, and Pranay Tankala. New insights into multicalibration. *arXiv preprint*, 2023. arXiv:2301.08837.

7 Alan Frieze and Ravi Kannan. Quick approximation to matrices and applications. *Combinatorica*, 19(2):175–220, 1999.

8 Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

9 Oded Goldreich, Shafi Goldwasser, and Asaf Nussboim. On the implementation of huge random objects. *SIAM Journal on Computing*, 39(7):2761–2822, 2010.

10 Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL: https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

11 Parikshit Gopalan, Lunjia Hu, Michael P. Kim, Omer Reingold, and Udi Wieder. Loss Minimization Through the Lens Of Outcome Indistinguishability. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 60:1–60:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.60.

12 Parikshit Gopalan, Adam Tauman Kalai, Omer Reingold, Vatsal Sharan, and Udi Wieder. Omnipredictors. In Mark Braverman, editor, *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 – February 3, 2022, Berkeley, CA, USA*, volume 215 of *LIPIcs*, pages 79:1–79:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.ITCS.2022.79.

13 Parikshit Gopalan, Michael P Kim, Mihir A Singhal, and Shengjia Zhao. Low-degree multicalibration. In Po-Ling Loh and Maxim Raginsky, editors, *Proceedings of Thirty Fifth Conference on Learning Theory*, volume 178 of *Proceedings of Machine Learning Research*, pages 3193–3234. PMLR, 02–05 July 2022. URL: https://proceedings.mlr.press/v178/gopalan22a.html.

14 Ursula Hébert-Johnson, Michael Kim, Omer Reingold, and Guy Rothblum. Multicalibration: Calibration for the (computationally-identifiable) masses. In *International Conference on Machine Learning*, pages 1939–1948. PMLR, 2018.

15 Paul W. Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social Networks*, 5(2):109–137, 1983. doi:10.1016/0378-8733(83)90021-7.

16 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the AMS*, 43(4):439–561, 2006.

17 Lunjia Hu, Inbal Livni-Navon, and Omer Reingold. Generative models of huge objects, 2023. arXiv:2302.12823.

18 Lunjia Hu, Inbal Livni-Navon, Omer Reingold, and Chutong Yang. Omnipredictors for constrained optimization. *arXiv preprint*, 2022. arXiv:2209.07463.

19 Lunjia Hu and Charlotte Peale. Comparative Learning: A Sample Complexity Theory for Two Hypothesis Classes. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 72:1–72:30, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.72.

20 Lunjia Hu, Charlotte Peale, and Omer Reingold. Metric entropy duality and the sample complexity of outcome indistinguishability. In Sanjoy Dasgupta and Nika Haghtalab, editors, *Proceedings of The 33rd International Conference on Algorithmic Learning Theory*, volume 167 of *Proceedings of Machine Learning Research*, pages 515–552. PMLR, 29 March–01 April 2022. URL: https://proceedings.mlr.press/v167/hu22a.html.

21 Russell Impagliazzo. Lecture on learning models: connections between boosting, hard-core distributions, dense models, GAN, and regularity I. https://www.ias.edu/video/csdm/2017/1113-RussellImpagliazzo, 2017.

22 Dimitar Jetchev and Krzysztof Pietrzak. How to fake auxiliary input. In *Theory of cryptography*, volume 8349 of *Lecture Notes in Comput. Sci.*, pages 566–590. Springer, Heidelberg, 2014. doi:10.1007/978-3-642-54242-8_24.

23 Michael P Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, pages 247–254, 2019.

24 Michael P. Kim, Christoph Kern, Shafi Goldwasser, Frauke Kreuter, and Omer Reingold. Universal adaptability: Target-independent inference that competes with propensity scoring. *Proceedings of the National Academy of Sciences*, 119(4):e2108097119, 2022. doi:10.1073/pnas.2108097119.

25 Michael P. Kim and Juan C. Perdomo. Making Decisions Under Outcome Performativity. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 79:1–79:15, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITCS.2023.79.

26 Yoshiharu Kohayakawa and Vojtech Rödl. Szemerédi's regularity lemma and quasi-randomness. *Recent advances in algorithms and combinatorics*, pages 289–351, 2003.

27 Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

28 Michael Mitzenmacher. A Brief History of Generative Models for Power Law and Lognormal Distributions. *Internet Mathematics*, 1(2):226–251, 2003.

29 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM Journal on Computing*, 22(4):838–856, August 1993.

30 Moni Naor and Asaf Nussboim. Implementing huge sparse random graphs. In Moses Charikar, Klaus Jansen, Omer Reingold, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, 10th International Workshop, APPROX 2007, and 11th International Workshop, RANDOM 2007, Princeton, NJ, USA, August 20-22, 2007, Proceedings*, volume 4627 of *Lecture Notes in Computer Science*, pages 596–608. Springer, 2007. doi:10.1007/978-3-540-74208-1_43.

31 Moni Naor, Asaf Nussboim, and Eran Tromer. Efficiently constructible huge graphs that preserve first order properties of random graphs. In Joe Kilian, editor, *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, volume 3378 of *Lecture Notes in Computer Science*, pages 66–85. Springer, 2005. doi:10.1007/978-3-540-30576-7_5.

32 Moni Naor and Omer Reingold. Constructing pseudo-random permutations with a prescribed structure. *J. Cryptol.*, 15(2):97–102, January 2002. doi:10.1007/s00145-001-0008-5.

33 Alexander Scott. Szemerédi's regularity lemma for matrices and sparse graphs. *Combinatorics, Probability and Computing*, 20(3):455–466, 2011.

34 Maciej Skórski. Simulating auxiliary inputs, revisited. In *Theory of cryptography. Part I*, volume 9985 of *Lecture Notes in Comput. Sci.*, pages 159–179. Springer, Berlin, 2016. doi:10.1007/978-3-662-53641-4_7.

35 Maciej Skórski. A subgradient algorithm for computational distances and applications to cryptography. *Cryptology ePrint Archive*, 2016.

36 Maciej Skórski. A cryptographic view of regularity lemmas: simpler unified proofs and refined bounds. In *Theory and applications of models of computation*, volume 10185 of *Lecture Notes in Comput. Sci.*, pages 586–599. Springer, Cham, 2017. doi:10.1007/978-3-319-55911-7.

37 Luca Trevisan, Madhur Tulsiani, and Salil P. Vadhan. Regularity, boosting, and efficiently simulating every high-entropy distribution. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 126–136. IEEE Computer Society, 2009. doi:10.1109/CCC.2009.41.

38 Salil Vadhan and Colin Jia Zheng. A uniform min-max theorem with applications in cryptography. In *Advances in Cryptology–CRYPTO 2013: 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 93–110. Springer, 2013.

39 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

40 Mark N. Wegman and J. Lawrence Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, 1981.

41 Andrew C. Yao. Theory and applications of trapdoor functions. In *23rd annual symposium on foundations of computer science (Chicago, Ill., 1982)*, pages 80–91. IEEE, New York, 1982.

Bounded Relativization

Shuichi Hirahara
National Institute of Informatics, Tokyo, Japan

Zhenjian Lu
University of Oxford, UK

Hanlin Ren
University of Oxford, UK

Abstract

Relativization is one of the most fundamental concepts in complexity theory, which explains the difficulty of resolving major open problems. In this paper, we propose a weaker notion of relativization called *bounded relativization*. For a complexity class $\mathfrak{C}$, we say that a statement is $\mathfrak{C}$-*relativizing* if the statement holds relative to every oracle $\mathcal{O} \in \mathfrak{C}$. It is easy to see that every result that relativizes also $\mathfrak{C}$-relativizes for every complexity class $\mathfrak{C}$. On the other hand, we observe that many non-relativizing results, such as $\mathsf{IP} = \mathsf{PSPACE}$, are in fact PSPACE-relativizing.

First, we use the idea of bounded relativization to obtain new lower bound results, including the following nearly maximum circuit lower bound: for every constant $\varepsilon > 0$,

$$\mathsf{BPE}^{\mathrm{MCSP}}/_{2^{\varepsilon n}} \not\subseteq \mathsf{SIZE}[2^n/n].$$

We prove this by PSPACE-relativizing the recent pseudodeterministic pseudorandom generator by Lu, Oliveira, and Santhanam (STOC 2021).

Next, we study the limitations of PSPACE-relativizing proof techniques, and show that a seemingly minor improvement over the known results using PSPACE-relativizing techniques would imply a breakthrough separation $\mathsf{NP} \neq \mathsf{L}$. For example:

- Impagliazzo and Wigderson (JCSS 2001) proved that if $\mathsf{EXP} \neq \mathsf{BPP}$, then BPP admits infinitely-often subexponential-time heuristic derandomization. We show that their result is PSPACE-relativizing, and that improving it to worst-case derandomization using PSPACE-relativizing techniques implies $\mathsf{NP} \neq \mathsf{L}$.
- Oliveira and Santhanam (STOC 2017) recently proved that every dense subset in P admits an infinitely-often subexponential-time pseudodeterministic construction, which we observe is PSPACE-relativizing. Improving this to almost-everywhere (pseudodeterministic) or (infinitely-often) deterministic constructions by PSPACE-relativizing techniques implies $\mathsf{NP} \neq \mathsf{L}$.
- Santhanam (SICOMP 2009) proved that pr-MA does not have fixed polynomial-size circuits. This lower bound can be shown PSPACE-relativizing, and we show that improving it to an almost-everywhere lower bound using PSPACE-relativizing techniques implies $\mathsf{NP} \neq \mathsf{L}$.

In fact, we show that if we can use PSPACE-relativizing techniques to obtain the above-mentioned improvements, then $\mathsf{PSPACE} \neq \mathsf{EXPH}$. We obtain our barrier results by constructing suitable oracles computable in EXPH relative to which these improvements are impossible.

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Oracles and decision trees; Theory of computation → Circuit complexity; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases relativization, circuit lower bound, derandomization, explicit construction, pseudodeterministic algorithms, interactive proofs

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.6

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2023/070/` [37]

Funding *Shuichi Hirahara*: Supported by JST, PRESTO Grant Number JPMJPR2024, Japan.
Hanlin Ren: Received support from DIMACS through grant number CCF-1836666 from the National Science Foundation.

Acknowledgements We thank Rahul Santhanam for helpful discussions and for pointing out that the oracle in Theorem 52 can be constructed in EXPH, using [31] only as a black-box. We thank Lijie Chen for helpful discussions regarding [19], Ian Mertz for discussions about the statement (*), and Ryan Williams for useful discussions about time-space tradeoffs for SAT. We thank Lijie Chen (again) and an anonymous CCC reviewer for pointing out an error in a previous version of this paper. Part of this work was completed when the authors are visiting the Simons Institute for the Theory of Computing, participating in the *Meta-Complexity* program.

1 Introduction

The *relativization* barrier, introduced by Baker, Gill, and Solovay [11], is an influential meta-mathematical barrier in complexity theory. Techniques based on simulation and diagonalization tend to *relativize*, in the sense that they also work in an "oracle world" where every machine has access to an oracle $\mathcal{O}$. On the other hand, the P vs. NP question cannot be solved in a relativizing way: [11] showed that there exists an oracle $\mathcal{A}$ such that $\mathsf{P}^{\mathcal{A}} = \mathsf{NP}^{\mathcal{A}}$, as well as another oracle $\mathcal{B}$ such that $\mathsf{P}^{\mathcal{B}} \neq \mathsf{NP}^{\mathcal{B}}$. Relativization has been successful not only in explaining the difficulty of resolving the P vs. NP question, but also in pinning down the *exact* place where we are stuck – for example, there are oracles relative to which $\mathsf{BPP} = \mathsf{EXP}^{\mathsf{NP}}$ [32, 14], or $\mathsf{P}^{\mathsf{NP}} \subseteq \mathsf{SIZE}[O(n)]$ [80].

In the early 1990s, the interactive proof results such as $\mathsf{IP} = \mathsf{PSPACE}$ [55, 66] generated much excitement among complexity theorists, as they are the first examples of "truly compelling" [4] non-relativizing results in complexity theory. Indeed, $\mathsf{coNP}^{\mathcal{O}} \not\subseteq \mathsf{IP}^{\mathcal{O}}$ relative to a random oracle $\mathcal{O}$ [28, 16]. The $\mathsf{IP} = \mathsf{PSPACE}$ result and its underlying technique, *arithmetization*, has significantly expanded our knowledge about circuit lower bounds [13, 40, 73, 1, 65, 27], derandomization [42, 70, 20, 21, 19], Karp–Lipton theorems [55, 10, 39, 18], meta-complexity [5, 61, 39, 60, 54], and other areas.

Still, it seems that arithmetization alone would not suffice to resolve the P vs. NP question. How far can we push these techniques? As relativization does not capture the "current techniques" anymore [16, 13, 1], what are the other barriers preventing us from making progress?

Aaronson and Wigderson [3] proposed the *algebrization* barrier to capture the limitations of arithmetization. Roughly speaking, an inclusion $\mathfrak{C} \subseteq \mathfrak{D}$ *algebrizes* if $\mathfrak{C}^{\mathcal{O}} \subseteq \mathfrak{D}^{\widetilde{\mathcal{O}}}$ for every oracle $\mathcal{O}$ and every low-degree extension $\widetilde{\mathcal{O}}$ of $\mathcal{O}$. This framework captures most results proved using arithmetization, such as $\mathsf{IP} = \mathsf{PSPACE}$ [66] and $\mathsf{MIP} = \mathsf{NEXP}$ [9].[1] On the other hand, [3] constructed oracles $\mathcal{O}_1$, $\mathcal{O}_2$, and $\mathcal{O}_3$ such that $\mathsf{NP}^{\widetilde{\mathcal{O}_1}} \subseteq \mathsf{P}^{\mathcal{O}_1}$, $\mathsf{NEXP}^{\widetilde{\mathcal{O}_2}} \subseteq \mathsf{P}^{\mathcal{O}_2}/_{\mathrm{poly}}$, and $\mathsf{RP}^{\mathcal{O}_3} \not\subseteq \mathsf{P}^{\widetilde{\mathcal{O}_3}}$. Therefore, techniques based on arithmetization are not enough for proving long-standing conjectures in complexity theory such as $\mathsf{P} \neq \mathsf{NP}$, $\mathsf{NEXP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$, and $\mathsf{RP} = \mathsf{P}$.

It turns out that the algebrization barrier suffers from subtleties. For example, it was unclear how to formalize algebrization for complexity-theoretic statements that are neither separation nor inclusion; moreover, algebrization is not closed under *modus ponens*. Impagliazzo, Kabanets, and Kolokolova [38] and Aydınlıoğlu and Bach [8] revised the definition of algebrization to fix these issues. For more discussion of algebrization, see Section 1.4.

[1] A subtle issue on relativizing NEXP is that we need to restrict the NEXP machine to only query $\mathcal{O}$ on polynomially-long inputs. In other words, while we can show that $\mathsf{NEXP}^{\mathcal{O}[\mathrm{poly}]} \subseteq \mathsf{MIP}^{\widetilde{\mathcal{O}}}$ for every oracle $\mathcal{O}$ and its low-degree extension $\widetilde{\mathcal{O}}$, the statement $\mathsf{NEXP}^{\mathcal{O}} \subseteq \mathsf{MIP}^{\widetilde{\mathcal{O}}}$ is false in general. We refer the readers to Remark 2 for more details on relativizing space-bounded and exponential-time classes.

Unfortunately, algebrization did not become as popular as relativization in proving barrier results and predicting the limitations of "current techniques". Perhaps one reason is that algebrization barriers are harder to demonstrate, as one needs to construct an oracle $\mathcal{O}$ that diagonalizes against its low-degree extension $\widetilde{\mathcal{O}}$.[2] For many complexity theoretic statements that are slightly more complicated than inclusions ("$\mathfrak{C} \subseteq \mathfrak{D}$") or separations ("$\mathfrak{C} \not\subseteq \mathfrak{D}$"), it appears much harder to demonstrate algebrization barriers than relativization barriers.

1.1 Bounded Relativization

This paper takes one step back and considers relativization in the presence of non-relativizing techniques. Our main message is perhaps surprising: the shadow of the relativization barrier has *never* gone away, even though we already have powerful non-relativizing techniques at hand!

Specifically, we put forward a notion of *bounded relativization*. Roughly speaking, for a complexity class $\mathfrak{C}$, a complexity-theoretic statement is $\mathfrak{C}$-*relativizing* if it is true relative to every oracle $\mathcal{O} \in \mathfrak{C}$. It is easy to see that every statement that relativizes also $\mathfrak{C}$-relativizes, for every complexity class $\mathfrak{C}$. On the other hand, even though $\mathsf{IP} = \mathsf{PSPACE}$ is not relativizing, it is easily seen to be PSPACE-relativizing:

▶ **Proposition 1.** *For every oracle* $\mathcal{O} \in \mathsf{PSPACE}$, $\mathsf{IP}^{\mathcal{O}} = \mathsf{PSPACE}^{\mathcal{O}}$.

Proof. Since the proof of $\mathsf{IP} \subseteq \mathsf{PSPACE}$ is relativizing, we have $\mathsf{IP}^{\mathcal{O}} \subseteq \mathsf{PSPACE}^{\mathcal{O}}$.

On the other hand, since $\mathcal{O} \in \mathsf{PSPACE}$, we have $\mathsf{PSPACE}^{\mathcal{O}} = \mathsf{PSPACE} = \mathsf{IP} \subseteq \mathsf{IP}^{\mathcal{O}}$. ◀

▶ Remark 2. In this paper, when we relativize space-bounded machines (in particular PSPACE), we assume the query tape is counted into the space bound. That is, a $\mathsf{PSPACE}^{\mathcal{O}}$ machine can only query $\mathcal{O}$ on polynomially-long inputs. It is easy to see that under this definition, $\mathsf{PSPACE}^{\mathsf{PSPACE}} = \mathsf{PSPACE}$.

On the other hand, we allow $\mathsf{EXP}^{\mathcal{O}}$ or $\mathsf{NEXP}^{\mathcal{O}}$ machines to issue exponentially-long queries to $\mathcal{O}$. When we restrict the query length to be a polynomial, we write $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]}$ and $\mathsf{NEXP}^{\mathcal{O}[\mathrm{poly}]}$ instead.

How is Proposition 1 helpful in studying the limitation of $\mathsf{IP} = \mathsf{PSPACE}$ as a technique? Say that we want to prove "a slight improvement of [42]" using "current techniques". (For now, the exact meaning of "a slight improvement of [42]" is not important; a concrete example will be given in Section 1.3.1. We also do not formally define "current techniques" here.) Suppose we construct an oracle $\mathcal{O}$, *in the usual, Baker–Gill–Solovay sense of relativization*, relative to which the "slight improvement of [42]" is impossible (see, e.g., Theorem 10).

- Suppose, in addition, that $\mathcal{O} \in \mathsf{PSPACE}$. It follows immediately that, if the term "current techniques" is interpreted as "PSPACE-relativizing techniques", then "current techniques" cannot prove the desired "slight improvement".
- In reality, it is often the case that we do not know how to put $\mathcal{O}$ in PSPACE; however, we can still show that $\mathcal{O} \in \mathsf{EXPH}$ in many cases. Then, the oracle $\mathcal{O}$ tells us that, if there is a PSPACE-relativizing proof of the "slight improvement of [42]", then this proof also implies $\mathsf{PSPACE} \neq \mathsf{EXPH}$. The latter would be a breakthrough in complexity theory, and in particular, it implies $\mathsf{L} \neq \mathsf{NP}$.

[2] Or, one needs to make sure the oracle $\mathcal{O}$ satisfies the so-called *Arithmetic Checkability Theorem* in the sense of [38].

To summarize, any EXPH-computable oracle presents the following barrier to "current techniques":

> If "current techniques" are PSPACE-relativizing and cannot separate L from NP, then "current techniques" also cannot prove the "slight improvement of [42]" (or any statement that is not EXPH-relativizing).

How "relativizing" are interactive proof results? The above discussion suggests that IP = PSPACE is actually "mildly relativizing" in the sense that it requires significant computational power (namely, super-polynomial space) to create an oracle world in which it is false. Using the same reasoning as Proposition 1, we can see that other interactive proof results, such as MIP = NEXP [9] and $\mathsf{MIP}^* = \mathsf{RE}$ [44], are also "mildly relativizing".

▶ **Proposition 3.** *The following are true:*

- **(Relativization of** MIP = NEXP**)** *For every* $\mathcal{O} \in \mathsf{NEXP} \cap \mathsf{coNEXP}$, $\mathsf{MIP}^{\mathcal{O}} = \mathsf{NEXP}^{\mathcal{O}[\mathrm{poly}]}$.
- **(Relativization of** $\mathsf{MIP}^* = \mathsf{RE}$**)** *For every* $\mathcal{O} \in \mathsf{R}$ *(i.e.,* $\mathcal{O}$ *is computable),* $(\mathsf{MIP}^*)^{\mathcal{O}} = \mathsf{RE}^{\mathcal{O}}$.

Proof. One direction relativizes: $\mathsf{MIP}^{\mathcal{O}} \subseteq \mathsf{NEXP}^{\mathcal{O}[\mathrm{poly}]}$, and $(\mathsf{MIP}^*)^{\mathcal{O}} \subseteq \mathsf{RE}^{\mathcal{O}}$. For the other direction, notice that $\mathsf{NEXP}^{\mathcal{O}[\mathrm{poly}]} \subseteq \mathsf{NEXP} = \mathsf{MIP} \subseteq \mathsf{MIP}^{\mathcal{O}}$ and $\mathsf{RE}^{\mathcal{O}} \subseteq \mathsf{RE} = \mathsf{MIP}^* \subseteq (\mathsf{MIP}^*)^{\mathcal{O}}$. ◀

Consequently, many results proved by combining interactive proof techniques and other relativizing techniques are also $\mathfrak{C}$-relativizing for a large complexity class $\mathfrak{C}$.[3] (It seems fair to say that most of them are PSPACE-relativizing.) This paper will explore the following two directions:

1. The positive direction: we relativize some previous results with an oracle $\mathcal{O} \in \mathsf{PSPACE}$ to obtain new results.
2. The negative direction: we construct oracle worlds in EXPH and demonstrate that certain seemingly minor improvements over known results might be hard to prove by PSPACE-relativizing techniques.

1.2 New Lower and Upper Bounds via Bounded Relativization

We first explain our positive results. Lu, Oliveira, and Santhanam [54] constructed a pseudodeterministic pseudorandom generator (PRG) with sub-polynomial seed length and one bit of advice that infinitely-often fools uniform algorithms. That is, for every $\varepsilon > 0$, they constructed a function $G : \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$ computable in randomized polynomial time with one bit of advice, such that for every polynomial-time algorithm $\mathcal{A}$, G fools $\mathcal{A}$ on infinitely many input lengths. The term "pseudodeterministic" means that the randomized algorithm for G, on input $z \in \{0,1\}^{n^\varepsilon}$, will output the *fixed* string $G(z)$ with high probability.

Although the PRG construction in [54] is not relativizing, we observe that it is still PSPACE-relativizing. We exploit this fact to prove new circuit lower bounds and design new pseudodeterministic algorithms for the range avoidance problem.

[3] The fact that these results are $\mathfrak{C}$-relativizing might not be as obvious as Proposition 3, as one still needs to look into the proof of these results and replace the usage of interactive proof results by their bounded-relativizing counterparts. For example, using IP = PSPACE, Santhanam [65] showed that there is a problem in pr-MA without size-n^{100} circuits. To show this result is PSPACE-relativizing, one apparently needs to follow the proof of [65] and replace all occurrences of "IP = PSPACE" by Proposition 1.

A Nearly Maximum Circuit Lower Bound. Our first result shows that $\mathsf{BPE}^{\mathrm{MCSP}}/_{2^{\varepsilon n}}$ requires circuits of nearly maximum size. Here, MCSP is the *minimum circuit size problem* [46] that takes as input the length-N truth table of a function $f : \{0,1\}^{\log N} \to \{0,1\}$ and a size parameter s, and decides whether f can be computed by a size-s circuit.

▶ **Theorem 4.** *For every constant $\varepsilon > 0$, $\mathsf{BPE}^{\mathrm{MCSP}}/_{2^{\varepsilon n}}$ cannot be computed by circuits of size $2^n/n$.*

Note that every function can be computed by a circuit of size $(1 + o(1))2^n/n$ [56, 29]. Also, it is trivial that $\mathsf{E}/_{2^n}$ contains a language with maximum circuit complexity. For comparison, the advice length in our lower bound in Theorem 4 is only $2^{\varepsilon n}$, for an arbitrarily small constant $\varepsilon > 0$. Previously, the smallest complexity class known to require maximum circuit complexity is $\mathsf{E}^{\Sigma_2^p}$ [57]. It was proved in [39] that $\mathsf{ZPEXP}^{\mathrm{MCSP}}$ does not have polynomial-size circuits, but an exponential-size lower bound for this class is open.

Proof Idea of Theorem 4. Theorem 4 is proved by PSPACE-relativizing the PRG construction in [54]. Let $\mathcal{O} := \mathrm{MCSP} \in \mathsf{PSPACE}$. We can relativize the PRG in [54] with the MCSP oracle and obtain:

> For every $\varepsilon > 0$, there is a function $G : \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$ computable in randomized polynomial time *with access to an* MCSP *oracle* and one bit of advice, such that for every polynomial-time algorithm $\mathcal{A}$ *with access to an* MCSP *oracle*, G fools $\mathcal{A}$ on infinitely many input lengths.

Let $\mathcal{A}$ be the following algorithm: given the truth table of a function $f : \{0,1\}^{\log N} \to \{0,1\}$, use the MCSP oracle to determine if the circuit complexity of f is at least $2^n/n$ (where $n := \log N$), and outputs 1 if and only if this is the case. Since most truth tables have circuit complexity at least $2^n/n$ [67, 56, 29], and $\mathcal{A}$ is fooled by the PRG G, it follows that there is a seed $z \in \{0,1\}^{N^{\varepsilon/2}}$ such that the function whose truth table is $G(z)$ also has circuit complexity at least $2^n/n$.

Consider the following language L. On input $x \in \{0,1\}^n$, it takes advice (z, α), where $z \in \{0,1\}^{2^{\varepsilon n/2}}$ is defined as above, and $\alpha \in \{0,1\}$ is the one-bit advice used by G. It computes the truth table $G(z)$ and accepts the input x if and only if the x-th bit of $G(z)$ is 1. Then L does not have circuits of size $2^n/n$. On the other hand, L can be computed by a $\mathsf{BPE}^{\mathrm{MCSP}}$ machine taking $2^{\varepsilon n}$ bits of advice. ◀

Besides the nearly-maximum circuit lower bound, we also prove new circuit lower bounds for meta-complexity problems and design new algorithms for range avoidance. We omit the proof ideas below since the main idea is essentially the same, namely relativizing the PRG in [54] to the MCSP oracle.

Circuit Lower Bounds for Meta-Complexity Problems. *Meta-computational* problems play a central role in a recently emerging area of research called *meta-complexity*, and have diverse applications in complexity theory, cryptography and learning. Roughly speaking, meta-computational problems are those that ask about the complexity (e.g., circuit complexity, time-bounded Kolmogorov complexity) of their inputs, and meta-complexity refers to the complexity of computing these problems themselves.

Before stating our result, we first recall some basic definitions. For a string x and a time bound function t, $\mathrm{K}^t(x)$, the *t-time-bounded Kolmogorov complexity* of x, is the minimum length of a string d such that $U(d)$ outputs x within $t(|x|)$ steps, where U is a time-optimal universal Turing machine fixed in advance. $\mathrm{rK}^t(x)$, the *randomized t-time-bounded*

Kolmogorov complexity of x, is defined in the same way as $\mathrm{K}^t(x)$ except that U in this case is a *randomized* universal Turing machine and we want x to be outputted with probability at least $2/3$. We can also define oracle versions of these complexity measures. For example, $\mathrm{rK}^{t,\mathcal{O}}$ can be defined in the same way as rK^t except that the universal Turing machine in this case has access to the oracle $\mathcal{O}$. (See Definition 21 for the formal definitions.)

Previously [35] showed that the problem of computing the $\mathrm{K}^{\mathrm{poly}}$ complexity of a given string is hard against fixed-polynomial-time deterministic algorithms. Later [54] showed a similar lower bound for computing $\mathrm{rK}^{\mathrm{poly}}$ but against fixed-polynomial-time randomized algorithms. Here, we show that computing $\mathrm{rK}^{\mathrm{poly},\mathrm{MCSP}}$ cannot be done using fixed-polynomial-size *circuits*.

▶ **Theorem 5** (Informal; see Theorem 32)**.** *For every $k \geq 1$, there is a polynomial t such that the problem of computing the $\mathrm{rK}^{t,\mathrm{MCSP}}$ complexity of a given string cannot be done using circuits of size n^k.*

In proving Theorem 5, we construct an efficient pseudodeterministic PRG, using MCSP oracle and a short advice string, that can fool polynomial-size *circuits*, which may be of independent interest. (See Theorem 33.)

Pseudodeterministic Construction for Range Avoidance. In the range avoidance problem, given a circuit $C\colon \{0,1\}^n \to \{0,1\}^{n+1}$, we are asked to find a string $x \in \{0,1\}^{n+1}$ that is not in the range of C. This problem is complete for the class called APEPP that corresponds to explicit constructions of objects whose existence can be shown using the probabilistic method [49, 51, 64]. It is open whether there is a deterministic polynomial-time algorithm with access to an NP oracle that solves the range avoidance problem; indeed, this open question is equivalent to the circuit lower bound $\mathsf{E}^{\mathsf{NP}} \not\subseteq \mathsf{SIZE}[2^{n/2}]$ [51].

Here, we present a new algorithm that *pseudodeterministically* solves the range avoidance problem for polynomial-size circuits. A pseudodeterministic algorithm is a probabilistic algorithm that "behaves like a deterministic algorithm" in the sense that it returns a *fixed* output with high probability over its internal randomness. Our algorithm runs in polynomial time, using an NP oracle and an advice of length n^ε (that is independent of the input circuit), and it works for infinitely many n, where n is the input length of the given circuit.

▶ **Theorem 6** (Informal; see Theorem 36)**.** *For every $\varepsilon > 0$ and $c \geq 1$, there exists a polynomial-time pseudodeterministic advice-taking oracle-algorithm A such that for infinitely many n, given a circuit $C\colon \{0,1\}^n \to \{0,1\}^{n+1}$ of size n^c, the algorithm A, with access to an NP oracle and n^ε bits of advice, outputs an $(n+1)$-bit string that is not in the range of C.*

1.3 Barriers for PSPACE-Relativizing Techniques

Next, we explain our negative results. We present EXPH-relativization barriers for "slightly improving" known results in *uniform derandomization*, *explicit construction*, and *circuit lower bounds*.

1.3.1 Uniform Derandomization

Standard "hardness vs. randomness" paradigm [59, 41] requires lower bounds against non-uniform circuits, such as $\mathsf{E} \not\subseteq \mathsf{i.o.SIZE}[2^{\varepsilon n}]$ for some constant $\varepsilon > 0$. In their seminal work, Impagliazzo and Wigderson [42] showed that hardness against uniform algorithms

also implies weak forms of derandomization: in particular, if $\mathsf{EXP} \neq \mathsf{BPP}$, then every algorithm in BPP can be derandomized in subexponential time infinitely often on average, i.e., $\mathsf{BPP} \subseteq \mathsf{i.o.heur\text{-}SUBEXP}$.[4]

We observe that their techniques are PSPACE-relativizing:

▶ **Proposition 7** (Uniform Derandomization in [42] is PSPACE-Relativizing). *Let* $\mathcal{O} \in \mathsf{PSPACE}$. *If* $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]} \neq \mathsf{BPP}^{\mathcal{O}}$, *then* $\mathsf{BPP}^{\mathcal{O}} \subseteq \mathsf{i.o.heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$.

More recently, Chen, Rothblum, and Tell [19] presented a uniform derandomization result on *almost all input lengths*: Given any language in PSPACE that is almost-everywhere hard against probabilistic algorithms, we can derandomize RP and BPP on average on almost every input length, where the derandomization of BPP requires some advice.[5] Their techniques are also PSPACE-relativizing:

▶ **Proposition 8** (Uniform Derandomization in [19] is PSPACE-Relativizing). *Let* $\mathcal{O} \in \mathsf{PSPACE}$ *and suppose that* $\mathsf{PSPACE}^{\mathcal{O}} \not\subseteq \mathsf{i.o.BPP}^{\mathcal{O}}$. *Then*

$$\mathsf{RP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]} \quad \textit{and} \quad \mathsf{BPP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]}/_{O(\log n)}.$$

The above results only obtain average-case derandomization instead of worst-case ones. Can we obtain worst-case derandomization based on uniform hardness assumptions, such as $\mathsf{EXP} \neq \mathsf{BPP}$? Consider the following "slight improvement of [42]":

▶ **Conjecture 9** (Worst-Case Uniform Derandomization). $\mathsf{EXP} \neq \mathsf{BPP} \implies \mathsf{BPP} \subseteq \mathsf{i.o.SUBEXP}$.

We present the following oracle suggesting that resolving Conjecture 9 would require new techniques.

▶ **Theorem 10.** *There is an oracle* $\mathcal{O} \in \mathsf{EXPH}$ *such that*

$$\mathsf{RP}^{\mathcal{O}} \not\subseteq \mathsf{i.o.DTIME}^{\mathcal{O}}[2^n] \quad \textit{and} \quad \mathsf{UP}^{\mathcal{O}} \not\subseteq \mathsf{BPTIME}^{\mathcal{O}}[2^n].$$

In this oracle world, we have $\mathsf{UP} \not\subseteq \mathsf{BPTIME}[2^n]$ (which is much stronger than $\mathsf{EXP} \neq \mathsf{BPP}$), while at the same time, worst-case derandomization of RP into deterministic fixed-exponential (2^n) time is impossible. This oracle is in EXPH, and therefore any PSPACE-relativizing proof of Conjecture 9 would also imply a breakthrough separation $\mathsf{PSPACE} \neq \mathsf{EXPH}$.

An open question left from [19] is whether the $O(\log n)$-bit advice can be removed in their derandomization of BPP. We present some evidence that this is difficult using current techniques:

▶ **Theorem 11.** *There is an oracle* $\mathcal{O}$ *such that*

$$\mathsf{BPP}^{\mathcal{O}} \not\subseteq \mathsf{heur\text{-}DTIME}^{\mathcal{O}}[2^n] \quad \textit{and} \quad \mathsf{UP}^{\mathcal{O}} \not\subseteq \mathsf{i.o.BPTIME}^{\mathcal{O}}[2^n].$$

Unfortunately, we do not know if the oracle $\mathcal{O}$ constructed in Theorem 11 is in EXPH. Nevertheless, under an assumption that is unlikely yet seems difficult to rule out, we show how to construct the oracle $\mathcal{O}$ in polynomial space. The assumption is that

$$\mathsf{SAT} \in \mathsf{DTIME}[n \cdot \mathrm{polylog}(n)] \cap \mathsf{NC}, \tag{*}$$

i.e., SAT admits both a near-linear-time sequential algorithm and a $\mathrm{polylog}(n)$-time parallel algorithm.

[4] Here, $\mathsf{heur\text{-}SUBEXP}$ is the class of problems solvable by a deterministic subexponential time heuristic over every samplable distribution; see Definition 38 for details.

[5] We only state the "low-end" version of [19] here; note that they also proved some "high-end" derandomization results.

▶ **Theorem 12.** *Suppose that* $\mathsf{SAT} \in \mathsf{DTIME}[n \cdot \mathrm{polylog}(n)] \cap \mathsf{NC}$. *Then there is an oracle* $\mathcal{O}$ *satisfying the conclusions of Theorem 11 that can be computed in polynomial space.*

It follows that if we use PSPACE-relativizing techniques to eliminate the $O(\log n)$-bit advice in the derandomization of [19], then we can refute (*), thus proving a non-trivial lower bound for SAT. Thus, we can still say that the oracle $\mathcal{O}$ in Theorem 11 presents some evidence that "current proof techniques" do not suffice to eliminate the $O(\log n)$-bit advice in Proposition 8.

1.3.2 Explicit Constructions

Next, we consider bounded relativization for explicit constructions. A property $Q \subseteq \{0,1\}^*$ is *dense* if for every input length $n \in \mathbb{N}$, we have that $|Q \cap \{0,1\}^n| \geq 2^n/\mathrm{poly}(n)$. Given any dense property Q decidable in polynomial time and an input length n, how hard is it to construct a length-n string that is in Q? This is a central open question in derandomization.

Recently, Oliveira and Santhanam [62] showed how to make progress if we allow the construction algorithm to be *pseudodeterministic*. Recall that an algorithm is *pseudodeterministic* if it outputs the same valid answer with high probability (despite being probabilistic). It was shown in [62] that for every dense property $Q \in \mathsf{P}$, there is a subexponential-time pseudodeterministic construction algorithm that succeeds infinitely often. We observe that this result is PSPACE-relativizing:

▶ **Proposition 13** ([62] is PSPACE-Relativizing; Informal)**.** *For every oracle* $\mathcal{O} \in \mathsf{PSPACE}$ *and every dense property* $Q \in \mathsf{P}^{\mathcal{O}}$*, there is a zero-error pseudodeterministic* $\mathcal{O}$*-oracle algorithm that on input* 1^n *outputs some element in* $Q \cap \{0,1\}^n$ *in subexponential time, for infinitely many* $n \in \mathbb{N}$.

Scott Aaronson's blog [2] contains a nice description of the result in [62], where he also mentioned that the result has "merely the following four caveats": (1) the algorithm runs in *subexponential* time instead of polynomial time; (2) the algorithm is not deterministic but *pseudodeterministic*; (3) instead of being almost-everywhere, the algorithm only succeeds on infinitely many input lengths; and (4) [62] only proved the *existence* of such an algorithm, but were unable to say what the algorithm is. In this paper, we show that caveats (2) and (3) cannot be improved by EXPH-relativizing techniques.

We start with (3), namely that EXPH-relativizing techniques cannot improve the pseudodeterministic construction algorithm in [62] to work on almost every input length.

▶ **Theorem 14.** *There is an oracle* $\mathcal{O} \in \mathsf{EXPH}$ *and a dense property* $Q \in \mathsf{P}^{\mathcal{O}}$ *such that every pseudodeterministic algorithm that runs in* $2^{o(n)}$ *time and attempts to find a string in* $Q \cap \{0,1\}^n$ *fails on infinitely many input lengths* n.

Goldwasser, Impagliazzo, Pitassi, and Santhanam [31] showed that the pseudodeterministic query complexity of a certain NP search problem is $\Omega(\sqrt{N})$, where N is the input length. We prove Theorem 14 by using the results in [31] *directly as a black box*. Each time we diagonalize against a certain pseudodeterministic machine by finding the lexicographically first counterexample (which is guaranteed to exist by [31]). This procedure, even when naïvely implemented, is computable in EXPH. We think this is the interesting aspect of Theorem 14, as it illustrates our main point that the ghost of (bounded) relativization has never gone away from complexity theory: for many natural complexity-theoretic statements that are non-relativizing, the straightforward counterexample oracle construction is already in EXPH.

Now we move to (2). We show that any deterministic construction algorithm for dense properties has to overcome an EXP-relativization barrier, even if the algorithm is allowed to only succeed on infinitely many input lengths:

▶ **Theorem 15.** *There is an oracle $\mathcal{O} \in \mathsf{EXP}$ and a dense property $Q \in \mathsf{P}^{\mathcal{O}}$ such that every deterministic algorithm that runs in $2^n/n^{\omega(1)}$ time fails to find a string in $Q \cap \{0,1\}^n$ on almost every input length n.*

In fact, the complexity of deterministic construction is known to be equivalent to the complexity of approximating Levin's Kt-complexity for a certain range of parameters [34]. Here, the Kt-complexity of a string x is defined to be the minimum of $|M| + \log t$ over all t-time Turing machines M of length $|M|$ such that M outputs x (see Definition 22 for a precise definition). Informally, MKtP is the problem of computing $\mathrm{Kt}(x)$ for a given string x. Although MKtP is EXP-complete under non-uniform polynomial-time reductions [5], it is a long-standing open question to show that $\mathsf{MKtP} \notin \mathsf{P}$. Using the connection between MKtP and deterministic construction [34], we show that Kt-complexity can be efficiently approximated under the oracle of Theorem 15.

▶ **Theorem 16** (Informal; see Theorem 55)**.** *There exists an oracle $\mathcal{O} \in \mathsf{EXP}$ under which $\mathrm{Kt}(x)$ can be approximated to within a factor of $(1+\varepsilon)$ for every constant $\varepsilon > 0$ in time $n^{O(\log n)}$ on input x of length n.*

Previously, Ren and Santhanam [63] constructed an oracle under which approximating $\mathrm{Kt}(x)$ to within a factor of $(2+\varepsilon)$ is in P. Theorem 16 improves this oracle construction, at the cost of increasing the running time from $n^{O(1)}$ to $n^{O(\log n)}$. We note that there are lower bounds for the randomized variant of MKtP [60, 36]. Their proof techniques are PSPACE-relativizing [60] and relativizing [36], respectively. Theorem 16 suggests that such proof techniques cannot be used to show a lower bound for approximating Kt-complexity without resolving $\mathsf{L} \neq \mathsf{P}$.

The property Q of Theorem 15 is a dense subset of $\left\{x : \mathrm{Kt}^{\mathcal{O}}(x) \geq n - c\log n\right\}$. To see why any $2^n/n^{\omega(1)}$-time algorithm $A^{\mathcal{O}}$ fails to find a string in $Q \cap \{0,1\}^n$, observe that the output of $A^{\mathcal{O}}$ on input 1^n has $\mathrm{Kt}^{\mathcal{O}}$-complexity at most $O(\log n) + \log(2^n/n^{\omega(1)}) \leq n - \omega(\log n)$, which is not in Q. To obtain Theorem 16, we regard Q as an errorless heuristic algorithm for $\mathrm{Kt}^{\mathcal{O}}$, and use the worst-case to average-case reduction for $\mathrm{Kt}^{\mathcal{O}}$ [33] to obtain a worst-case approximation algorithm for $\mathrm{Kt}^{\mathcal{O}}$-complexity.

1.3.3 Circuit Lower Bounds

Finally, we consider circuit lower bounds for Merlin–Arthur classes. Santhanam [65], improving [13], showed that for every integer k, there is a function computable in $\mathsf{MA}/_1$ (Merlin–Arthur protocols with one bit of advice) that does not have size-n^k circuits. We observe that this result is PSPACE-relativizing:

▶ **Proposition 17** ([65] is PSPACE-Relativizing)**.** *Let $\mathcal{O} \in \mathsf{PSPACE}$. Then for every constant $k \geq 1$, $\mathsf{MA}^{\mathcal{O}}/_1 \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[n^k]$.*

By showing that (a variant of) the oracle constructed by [13] is in EXPH, we show that Santhanam's circuit lower bound cannot be improved to an almost-everywhere circuit lower bound by EXPH-relativizing proof techniques.

▶ **Theorem 18.** *There is an oracle $\mathcal{O} \in \mathsf{EXPH}$ such that $\text{pr-}\mathsf{MATIME}^{\mathcal{O}}[2^n] \subseteq \text{i.o.}\,\mathsf{SIZE}^{\mathcal{O}}[O(n)]$.*

In fact, it is unknown if the fixed polynomial-size lower bound for Σ_2^p due to Kannan [47] (i.e., $\Sigma_2^p \not\subseteq \mathsf{SIZE}[n^k]$ for every constant k) can be improved to an almost-everywhere circuit lower bound. We are not aware of any relativization barrier for improving Kannan's lower bound. To the best of our knowledge, Theorem 18 is the first evidence that "current proof techniques" cannot improve fixed polynomial-size circuit lower bounds for $\Sigma_2^p \supseteq$ pr-MA to almost everywhere. Note that the smallest complexity class known to be outside $\mathsf{i.o.SIZE}[n^k]$ for every k is $\Delta_3^p = \mathsf{P}^{\Sigma_2^p}$ [57].

▶ Remark 19 (The Infinite-Often Phenomenon). As noticed in [17], *almost-everywhere* separations in structural complexity theory are significantly harder to prove than infinitely-often separations. The PSPACE-relativization barrier provides an explanation of such difficulties – it is much easier to construct oracles (in EXPH) such that certain separation fails infinitely often. On the other hand, achieving almost-everywhere separations in oracle worlds might be much harder, and in some cases (such as in Theorem 18) the resulting oracle is not in EXPH anymore.

Besides circuit lower bounds for pr-MA, another notorious open problem mentioned in [17] is proving almost-everywhere NTIME hierarchies. Indeed, there is an oracle $\mathcal{O}$ relative to which $\mathsf{NTIME}[2^n] \subseteq \mathsf{i.o.RP}$ [12], and it is easy to see that this oracle can be implemented in EXPH.

1.4 Comparison with Algebrization

Our paper and the line of work on the algebrization barrier [23, 7, 3, 38, 8] share a common motivation, namely, to capture the limitations of "current techniques" after the interactive proof results such as $\mathsf{IP} = \mathsf{PSPACE}$ were proved via arithmetization. Therefore, we feel it necessary to compare the two barriers. However, before we make the comparison, let us briefly review the different variants of algebrization [23, 3, 38, 8].

1.4.1 Variants of Algebrization

Already in 1994, Fortnow [23] showed that every oracle $\mathcal{O}$ is many-one reducible to some oracle $\mathcal{A}$ such that $\mathsf{IP}^{\mathcal{A}} = \mathsf{PSPACE}^{\mathcal{A}}$; the oracle $\mathcal{A}$ is the "algebraic extension" of $\mathcal{O}$ as defined in [23, Section 5.3]. However, the definition of "algebraic extension" is rather involved and Fortnow did not show any unprovability results relative to "algebraic-extended" oracles.

Aaronson and Wigderson [3] defined an inclusion $\mathfrak{C} \subseteq \mathfrak{D}$ to *algebrize* if for every oracle $\mathcal{O}$ and its low-degree extension $\widetilde{\mathcal{O}}$, $\mathfrak{C}^{\mathcal{O}} \subseteq \mathfrak{D}^{\widetilde{\mathcal{O}}}$. This simplifies Fortnow's definition in the following sense. Roughly speaking, Fortnow's "algebraic extension" of $\mathcal{O}$ is another (Boolean) oracle that encodes all information of $\mathcal{O}$, $\widetilde{\mathcal{O}}$, $\widetilde{\widetilde{\mathcal{O}}}$, $\widetilde{\widetilde{\widetilde{\mathcal{O}}}}$, $\cdots$; the Aaronson–Wigderson definition is simpler in that it only considers $\mathcal{O}$ and $\widetilde{\mathcal{O}}$. However, this comes with a price of asymmetry that the RHS of an inclusion has access to $\widetilde{\mathcal{O}}$ but the LHS does not (e.g., $\mathsf{PSPACE}^{\mathcal{O}} \subseteq \mathsf{IP}^{\widetilde{\mathcal{O}}}$). Consequently, the Aaronson–Wigderson definition of algebrization is not closed under *modus ponens*: for example, although $\mathsf{NEXP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ does not algebrize [3, Theorem 5.6], it is possible that one could find an intermediate class $\mathfrak{C}$ such that both $\mathfrak{C} \subseteq \mathsf{NEXP}$ and $\mathfrak{C} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ algebrizes, thus using algebrizing techniques to separate NEXP from $\mathsf{P}/_{\mathrm{poly}}$.

Partially due to the above drawback, there are two subsequent works [38, 8] that aim at a more satisfactory definition of algebrization. Both definitions have the spirit that a statement is algebrizing if it holds relative to every oracle $\mathcal{O}$ *satisfying certain algebraic properties*. For example, a statement is IKK-algebrizing (IKK stands for Impagliazzo–Kabanets–Kolokolova) if it holds relative to every oracle $\mathcal{O}$ that satisfies the so-called *Arithmetic Checkability*

Theorem (ACT) [38]; a statement is *affine-relativizing* [8] if it holds relative to every affine extension – the result of a particular error correcting code applied to the characteristic string of a language. These variants of algebrization are indeed closed under *modus ponens*, and an algebrization barrier for a statement is simply an oracle that satisfies the algebraic property and under which the statement is false.

1.4.2 Comparison

Recall that a statement is $\mathfrak{C}$-relativizing if it holds relative to every oracle whose complexity is in $\mathfrak{C}$. In bounded relativization, we are interested in the complexity of the oracle instead of its algebraic properties. Therefore, the notions of (say) PSPACE-relativization and (say) IKK-algebrization appear incomparable: there are oracles outside PSPACE (in particular, undecidable ones) that satisfy the ACT; on the other hand, it is unknown if there are oracles in PSPACE that do not satisfy the ACT.

Moreover, as a framework for barrier results, we think that the main advantage of bounded relativization is its *simplicity*:

- To demonstrate a barrier result via bounded relativization, one only needs to construct a corresponding oracle *in the usual sense of relativization*, and show an upper bound on the complexity of the oracle (e.g., in EXPH). On the other hand, one needs to work against low-degree extensions of Boolean oracles or ensure the ACT in order to prove an algebrization barrier result.
- On the other hand, as demonstrated in Propositions 1 and 3, interactive proof characterizations of complexity classes are bounded-relativizing for trivial reasons. If $\mathsf{MIP}^* = \mathsf{RE}$ is true, then there is a one-liner proof of it being R-relativizing. In contrast, it appears that one needs to examine the 200-page proof of [44] thoroughly to tell whether it is algebrizing.

However, as we discussed before, the bounded relativization barrier has its own subtle disadvantages. An oracle construction in EXPH (such as Theorem 10) only tells us the following information: "if PSPACE-relativizing techniques can prove certain statements (such as Conjecture 9), then we can also separate EXPH from PSPACE". We interpret this as limitations of PSPACE-relativizing techniques, since we do not think such techniques are powerful enough to separate EXPH from PSPACE.[6] Hence, the bounded relativization barrier does not explain the difficulty of (e.g.) proving $\mathsf{PSPACE} \neq \mathsf{EXPH}$, since this difficulty itself is assumed to indicate barriers for proving other statements. We leave it as an interesting open problem to justify the statement that *although arithmetization-based techniques such as* $\mathsf{IP} = \mathsf{PSPACE}$ *are already available, fundamentally new techniques are needed to prove separation results such as* $\mathsf{PSPACE} \neq \mathsf{EXPH}$.[7]

[6] Although we should not be too pessimistic about the resolution of grand challenges in complexity theory such as PSPACE vs. EXPH, it seems extremely unlikely to us that they will be resolved by building an EXPH-computable oracle world inside which a certain statement is false, and proving the same statement using PSPACE-relativizing techniques. We suspect that techniques fundamentally "deeper" than those used to build oracles (e.g., diagonalization) are needed.

[7] Or, to refute this statement by proving $\mathsf{PSPACE} \neq \mathsf{EXPH}$!

2 Preliminaries

2.1 Definitions and Notations

For $n, i \in \mathbb{N}$ and $x \in \{0,1\}^n$, where $i \leq n$, we let $x_{[i]}$ denote the i-bit prefix of x. We use $\mathcal{U}_n$ to denote the uniform distribution over all length-n strings.

We assume the reader is familiar with basic complexity classes such as PSPACE, EXP, IP, BPP, MA; the definitions of these classes can be found in [30, 6]. One complexity class that occurs frequently in this paper and that might not be familiar to the broader audience, though, is EXPH, the exponential-time hierarchy.

▶ **Definition 20.** *A language L is in* EXPH *if there is a constant k, an exponential bound $e(n) = 2^{\mathrm{poly}(n)}$, and a $\mathrm{poly}(e(n))$-time algorithm M such that, for every input $x \in \{0,1\}^*$,*

$$x \in L \iff \exists y_1 \forall y_2 \exists y_3 \forall y_4 \ldots \exists y_k, M(x, y_1, \ldots, y_k) = 1,$$

where $y_i \in \{0,1\}^{e(n)}$ for each $i \in [k]$.

Equivalently, a language is in EXPH if it can be decided in exponential ($2^{\mathrm{poly}(n)}$) time with access to a PH oracle.

Pseudodeterministic Algorithms. A pseudodeterministic algorithm is a probabilistic algorithm that returns a *fixed* output with high probability on *any* given input. We will also consider pseudodeterministic *oracle* (or *advice-taking*) algorithms. Such an algorithm will maintain its pseudodeterministic behavior (i.e., outputting a fixed answer with high probability for any input) when a correct oracle (or advice) is given.

Time-Bounded Kolmogorov Complexity. We fix a time-optimal universal Turing machine U and a time-optimal randomized Turing machine V.

▶ **Definition 21.** *For a string $x \in \{0,1\}^*$, a time bound function $t\colon \mathbb{N} \to \mathbb{N}$, and an oracle $\mathcal{O}$, we define*

- ***(t-time-bounded Kolmogorov complexity)***

$$\mathrm{K}^{t,\mathcal{O}}(x) := \min_{d \in \{0,1\}^*} \Big\{ |d| : U^{\mathcal{O}}(d) \text{ outputs } x \text{ in at most } t(|x|) \text{ steps} \Big\}.$$

- ***(Randomized t-time-bounded Kolmogorov complexity)***

$$\mathrm{rK}^{t,\mathcal{O}}(x) := \min_{d \in \{0,1\}^*} \Big\{ |d| : V^{\mathcal{O}}(d) \text{ runs in at most } t(|x|) \text{ steps and } \Pr[V^{\mathcal{O}}(d) = x] \geq 2/3 \Big\}.$$

We also consider Levin's notion of time-bounded Kolmogorov complexity [52].

▶ **Definition 22.** *For a string $x \in \{0,1\}^*$ and an oracle $\mathcal{O}$, we define*

$$\mathrm{Kt}^{\mathcal{O}}(x) := \min_{d \in \{0,1\}^*, t \in \mathbb{N}} \Big\{ |d| + \lceil \log t \rceil : U^{\mathcal{O}}(d) \text{ outputs } x \text{ in at most } t \text{ steps} \Big\}.$$

2.2 Technical Tools

▶ **Theorem 23** ([68])**.** *For every oracle $\mathcal{O}$, $\mathsf{BPP}^{\mathcal{O}} \subseteq (\Sigma_2^p)^{\mathcal{O}}$.*

We also need the following theorem for approximate counting that runs in linear time with a constant number of alternations. (In particular, Theorem 24 implies that approximate counting can be done in PH.)

▶ **Theorem 24** (Approximate Counting in Quasi-Linear-Time Hierarchy). *There is a* $\Sigma_5\mathsf{TIME}[n \cdot \mathrm{polylog}(n)]$ *machine* M *such that on input* (φ, K)*, accepts if* φ *has at least* K *satisfying assignments, and rejects if* φ *has at most* $K/2$ *satisfying assignments.*

Proof Sketch. Using universal hashing [69], we can show that there is a $\mathsf{BPTIME}[n \cdot \mathrm{polylog}(n)]^{\mathsf{SAT}}$ machine M' that approximates the number of satisfying assignments of φ'. Let $\mathcal{H}$ be a family of pairwise-independent hash functions whose output range is $\{0,1\}^{\lceil \log K \rceil}$, that is computable in near-linear time. (See, e.g., [72, Problem 3.3, (2)] for an example based on Toeplitz matrices.) The machine M' randomly samples $h \sim \mathcal{H}$ and $r \sim \{0,1\}^{\lceil \log K \rceil}$, and uses the SAT oracle to decide whether there is a satisfying assignment x of φ' such that $h(x) = r$.

Finally, the lemma follows from the fact that for every oracle $\mathcal{O}$, $\mathsf{BPTIME}^{\mathcal{O}}[t] \subseteq \Sigma_3\mathsf{TIME}[t \cdot \mathrm{polylog}(t)]^{\mathcal{O}}$ [74]. ◀

3 New Lower and Upper Bounds via Bounded Relativization

We start by showing a PSPACE-relativizing version of the pseudodeterministic PRG in [54].

3.1 A PSPACE-Relativizing, Pseudodeterministic, Efficient PRG

▶ **Theorem 25** (A PSPACE-Relativizing Efficient Pseudodeterministic PRG with One Bit of Advice). *For every* $\mathcal{O} \in \mathsf{PSPACE}$*,* $\varepsilon > 0$ *and* $c, d \geq 1$*, there exists a generator* $G = \{G_n\}_{n \in \mathbb{N}}$ *with* $G_n \colon \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$ *for which the following holds:*

Efficiency: *There is a probabilistic polynomial-time algorithm* A *such that for every* $n \in \mathbb{N}$*, on input* 1^n *and* $z \in n^\varepsilon$*,* A*, with oracle access to* $\mathcal{O}$ *and one advice bit* $\alpha(n) \in \{0,1\}$ *that is independent of* z*, outputs* $G_n(z)$ *with probability* $\geq 2/3$*.*

Pseudorandomness: *For every language* $L \in \mathsf{DTIME}^{\mathcal{O}}[n^c]$*, there exist infinitely many input lengths* n *such that*

$$\left| \Pr_{x \sim \{0,1\}^n}[L(x) = 1] - \Pr_{z \sim \{0,1\}^{n^\varepsilon}}[L(G_n(z)) = 1] \right| \leq \frac{1}{n^d}.$$

To show Theorem 25, we consider two cases.

Case 1: $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$

▶ **Lemma 26.** *Let* $\mathcal{O} \in \mathsf{PSPACE}$*. If* $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$*, then there is a PRG with seed length* $O(\log n)$*, computable in pseudodeterministic polynomial time with oracle access to* $\mathcal{O}$*, that fools* $\mathsf{DTIME}^{\mathcal{O}}[n]$ *with error* $1/n$ *for all but finitely many* n*.*

We need the following "relativizing version" of a hardness-to-randomness construction.

▶ **Lemma 27** ([41, 50, 62])**.** *Let* $\mathcal{O}$ *be any language. If there is an* $\varepsilon > 0$ *and a Boolean function* $h \in \mathsf{BPE}^{\mathcal{O}}$ *that requires* $\mathcal{O}$*-oracle circuits of size* $2^{\varepsilon m}$ *on all but finitely many input length* m*, then there is a PRG with* $O(\log n)$ *seed length, computable in pseudodeterministic polynomial time with oracle access to* $\mathcal{O}$*, that fools* $\mathsf{DTIME}^{\mathcal{O}}[n]$ *with error* $1/n$ *for all but finitely many* n*.*

Proof of Lemma 26. Since $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$, by a simple padding argument, we have that $\mathsf{DSPACE}\left[2^{O(n)}\right] \subseteq \mathsf{BPE}^{\mathcal{O}}$.

By direct diagonalization, there is a language $L \in \mathsf{DSPACE}\left[2^{O(n)}\right]$ such that, for all but finitely many input lengths n, L does not have $\mathcal{O}$-oracle circuits of size $2^{0.9n}$. Indeed, using $2^{O(n)}$ space, we can find the lexicographically first truth table of length 2^n that cannot be

computed by $\mathcal{O}$-oracle circuits of size $2^{0.9n}$. This can be done because $\mathcal{O} \in \mathsf{PSPACE}$ and we can simulate any such circuit in $2^{O(n)}$ space. By the simulation in the previous paragraph, we have that $L \in \mathsf{BPE}^{\mathcal{O}}$. Now the desired conclusion follows from Lemma 27. ◀

Case 2: $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$

▶ **Lemma 28.** *Let $\mathcal{O}$ be any language and suppose $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$. Then for every $\varepsilon > 0$ and $c, d \geq 1$, there exists a* PRG *with seed length n^{ε}, computable in pseudodeterministic polynomial time with oracle access to $\mathcal{O}$ and one bit of advice, that fools $\mathsf{DTIME}^{\mathcal{O}}[n^c]$ with error $1/n^d$ for infinitely many n.*

Proof Sketch. The proof is essentially the same as that of [54]. The only difference is that instead of assuming $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}$ in [54], here we assume $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$. We provide a high-level description of the proof here. For details, we refer the reader to [54, Sections 1.2 and 3.1].

As in [54], Lemma 28 can be shown by combining hierarchy theorems for probabilistic classes, which can be viewed as hardness results against (uniform) probabilistic algorithms, with the hardness-to-randomness framework under uniform hardness assumptions [42, 70]. More specifically, for any oracle $\mathcal{O}$ and $k \geq 1$, we can construct a language $L_k \in \mathsf{BPP}^{\mathcal{O}}/_1$ that is hard against $\mathsf{BPTIME}^{\mathcal{O}}[n^k]/_1$. (See [54, Lemma 19].)

As in [26], assuming $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$, the language L_k is obtained by constructing a padded version of a certain PSPACE-completed language L_{hard} that has useful structural properties, such as *downward self-reducibility*, *random self-reducibility*, and *instance checkability* [70, 26], using the idea of an "optimal $\mathcal{O}$-*oracle* algorithm" for L_{hard}. (See [54, Lemma 17].)

The language L_k constructed in this way will have certain forms of downward self-reducibility and random self-reducibility. These properties enable us to plug L_k into the hardness-to-randomness construction of [70] (with additional "relativization properties" observed by [50]), and get a PRG that is infinitely-often secure against $\mathcal{O}$-*oracle* adversaries. Also, the fact that $L_k \in \mathsf{BPP}^{\mathcal{O}}/_1$ allows us to compute the PRG efficiently with oracle access to $\mathcal{O}$ and with one bit of advice, in a pseudodeterministic manner. ◀

Proof of Theorem 25. The theorem follows directly from Lemma 26 and Lemma 28, since in both the cases of $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$ and of $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$, we get a PRG that satisfies the conditions stated in the theorem. ◀

3.2 A Nearly Maximum Circuit Lower Bound for $\mathsf{BPE}^{\mathrm{MCSP}}/_{2^{\varepsilon n}}$

▶ **Theorem 29.** *For every $\varepsilon > 0$,*

$$\mathsf{BPE}^{\mathrm{MCSP}}/_{2^{\varepsilon n}} \not\subseteq \mathsf{SIZE}[2^n/n].$$

We first show the following two lemmas.

▶ **Lemma 30.** *For every $\varepsilon > 0$, there is a probabilistic polynomial-time algorithm A such that the following holds.*

- *For every $N \in \mathbb{N}$, on input 1^N, the algorithm A, with oracle access to* MCSP *and with an advice $\alpha(N) \in \{0,1\}^{N^{\varepsilon}}$, outputs with high probability a fixed truth table T_N of length $2^{n:=\lfloor \log N \rfloor}$ (which corresponds to some function $f\colon \{0,1\}^{n=\lfloor \log N \rfloor} \to \{0,1\}$), and*
- *for infinitely many $N \in \mathbb{N}$, T_N has circuit complexity greater than $2^n/n$.*

Proof. Let D be the following algorithm. On input $x \in \{0,1\}^N$, D lets $n := \lfloor \log N \rfloor$ and checks if $\mathrm{MCSP}\big(x_{[2^n]}, 2^n/n\big)$ is false. It is clear D can be determined in time $\mathsf{DTIME}^{\mathrm{MCSP}}[O(N)]$ and that D accepts x only if $x_{[2^n]}$ is a truth table with circuit complexity greater than $2^n/n$. Also, since most truth tables of length 2^n have circuit complexity greater than $2^n/n$ [67, 56], the acceptance probability of D is at least $1/2$.

Consider the generator $\{G_N\}_N$ from Theorem 25 that fools $\mathsf{DTIME}^{\mathrm{MCSP}}[O(N)]$ for infinitely many N. By the security of $\{G_N\}_N$, there exists some seed $z \in \{0,1\}^{N^\varepsilon}$ such that $D(G_N(z)) = 1$, and with oracle access to MCSP and with some bit $b \in \{0,1\}$, we can compute $G_N(z)$ pseudodeterministically in time $\mathrm{poly}(N)$. Therefore, given such z and b as advice (which is of $N^\varepsilon + 1$ bits but we can always scale the parameter ε), we can obtain with high probability some fixed truth table of length 2^n with circuit complexity greater than $2^n/n$. ◀

▶ **Lemma 31.** *For every $\varepsilon > 0$, there is a probabilistic algorithm B such that the following holds.*

- *For every $n \in \mathbb{N}$, on input 1^n, the algorithm B, with oracle access to* MCSP *and with an advice $\alpha(n) \in \{0,1\}^{2^{\varepsilon n}}$, runs in time $2^{O(n)}$ and outputs with high probability a fixed truth table T_n of length 2^n, and*
- *for infinitely many $n \in \mathbb{N}$, T_n has circuit complexity greater than $2^n/n$.*

Proof. Let $\varepsilon_0 := \varepsilon/2$. We first describe the advice used by the algorithm B. For input 1^n, the first part of the advice is a number N (if exists) such that

- $n = \lfloor \log N \rfloor$, and that
- the algorithm A from Lemma 30 (instantiated with parameter ε_0) on input 1^N, with oracle access to MCSP and with N^{ε_0} bits of advice, outputs with high probability a fixed truth table of length 2^n with circuit complexity greater than $2^n/n$.

Note that such a number N can be specified using $n+1$ bits. Also, we say that n is *good* if such an N exists. By Lemma 30, there are infinitely many good n.

The second part of the advice is the N^{ε_0} bits that are needed to compute $A(1^N)$. Note that the total number of bits for the advice is at most

$$n + 1 + N^{\varepsilon_0} < 2^{\varepsilon n}.$$

We let the output of $B(1^n)$ be the output of $A(1^N)$. Then whenever n is good, the canonical output of $A(1^N)$ (hence $B(1^n)$) will be a truth table of length 2^n with circuit complexity greater than $2^n/n$. ◀

Theorem 29 now follows easily from Lemma 31.

Proof of Theorem 29. Consider the language $L \in \mathsf{BPE}^{\mathrm{MCSP}}/_{2^{\varepsilon n}}$ that can be computed as follows. On input $x \in \{0,1\}^n$, we run $B(1^n)$, where B is the algorithm from Lemma 31, using MCSP as an oracle and using $2^{\varepsilon n}$ bits of advice. With high probability, we obtain the truth table of some fixed function $f_n \colon \{0,1\}^n \to \{0,1\}$. We then let $L(x) := f_n(x)$. By Lemma 31, for infinitely many n, f_n has circuit complexity greater than $2^n/n$, which implies that $L \notin \mathsf{SIZE}[2^n/n]$. ◀

3.3 Circuit Lower Bounds for Meta-Complexity Problems

In this subsection, we show that the problem of estimating $\mathrm{rK}^{\mathrm{poly},\mathrm{MCSP}}$ (the MCSP-oracle version of polynomial-time randomized Kolmogorov complexity) does not have fixed-polynomial-size circuits.

▶ **Theorem 32** (An Unconditional Circuit Lower Bound for Estimating $\mathrm{rK}^{\mathsf{poly},\mathrm{MCSP}}$). *For every $\varepsilon > 0$ and $c \geq 1$ there exists a constant $k \geq 1$ such that the following holds. Consider the following promise problem $\Pi^\varepsilon = (\mathcal{YES}_n, \mathcal{NO}_n)_{n\in\mathbb{N}}$, where*

$$\begin{aligned} \mathcal{YES}_n &:= \left\{ x \in \{0,1\}^n : \mathrm{rK}^{t,\mathrm{MCSP}}(x) \leq n^\varepsilon \right\}, \\ \mathcal{NO}_n &:= \left\{ x \in \{0,1\}^n : \mathrm{rK}^{t,\mathrm{MCSP}}(x) \geq n - 10 \right\}, \end{aligned}$$

and $t(n) = n^k$. Then $\Pi^\varepsilon \notin \mathsf{SIZE}[n^c]$.

To show Theorem 32, we first construct an efficient pseudodeterministic PRG, with oracle access to MCSP and an advice string of length $O(\log n)$, that can fool *circuits* (instead of uniform algorithms in [54]).

3.3.1 A Pseudodeterministic PRG Fooling Circuits

▶ **Theorem 33.** *For every $\varepsilon > 0$ and every $c, d \geq 1$, there is a generator $G = \{G_n\}_{n\in\mathbb{N}}$ with $G_n : \{0,1\}^{n^\varepsilon} \to \{0,1\}^n$ such that the following holds.*

Efficiency: *There is a probabilistic polynomial-time algorithm A such that for every $n \in \mathbb{N}$, on input 1^n and $z \in \{0,1\}^{n^\varepsilon}$, A, with oracle access to* MCSP *and an advice $\alpha(n) \in \{0,1\}^{O(\log n)}$ that is independent of z, outputs $G_n(z)$ with probability $\geq 2/3$.*

Pseudorandomness: *For infinitely many $n \in \mathbb{N}$, for every circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size at most n^c,*

$$\left| \Pr_{x\sim\{0,1\}^n}[C(x) = 1] - \Pr_{z\sim\{0,1\}^{n^\varepsilon}}[C(G_n(z)) = 1] \right| \leq \frac{1}{n^d}.$$

We need the following hardness-to-randomness construction.

▶ **Lemma 34** (Umans [71]). *There is a universal constant g and a function $G_{\mathrm{Umans}} : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ such that, for all s and Y where the circuit complexity of Y (viewed as a truth table) is at least s^g, and for all circuits C of size s,*

$$\left| \Pr_{x\sim\{0,1\}^{g\log|Y|}}[C(G_{\mathrm{Umans}}(Y,x)) = 1] - \Pr_{x\sim\{0,1\}^s}[C(x) = 1] \right| \leq \frac{1}{s}.$$

Moreover, G_{Umans} is computable in time $\mathrm{poly}(|Y|)$.

We also need the following variant of Lemma 30, whose proof is essentially the same as that of Lemma 30. We omit the details here.

▶ **Lemma 35.** *For every $\varepsilon > 0$ and $d \geq 1$, there is a probabilistic polynomial-time algorithm A such that the following holds.*

- *For every $N \in \mathbb{N}$, on input 1^N and $z \in \{0,1\}^{N^\varepsilon}$, the algorithm A, with oracle access to* MCSP *and with an advice $\alpha_A(N) \in \{0,1\}$, outputs with high probability a fixed truth table $T_{N,z}$ of length $2^{m:=\lfloor \log N \rfloor}$ (which corresponds to some function $f\colon \{0,1\}^m \to \{0,1\}$), and*
- *for infinitely many $N \in \mathbb{N}$, with probability at least $1 - 1/N^d$ over z, $T_{N,z}$ has circuit complexity at least $2^{m/2}$.*

We now present the PRG in Theorem 33.

Proof of Theorem 33. The idea is to combine Lemma 35 and Lemma 34.

Let $c' := \max\{c, 2d\}$ and $\varepsilon_0 := \varepsilon/(7c'g)$, where g is the constant from Lemma 34.

For $n \in \mathbb{N}$, we first specify the advice for computing the generator G_n. The first part of the advice is a number N (if exists) such that

- $n^{3c'g} \leq N < (n+1)^{3c'g}$, and that

- the algorithm A from Lemma 35 (instantiated with parameters ε_0 and $d := 1$), on input 1^N and $z \in \{0,1\}^{N^{\varepsilon_0}}$, with oracle access to MCSP and with one advice bit $\alpha_A(N)$, outputs (with high probability) a fixed truth table of length $2^{\lfloor \log N \rfloor}$ that has circuit complexity greater than $2^{\lfloor \log N \rfloor/2}$, for at least $1 - 1/N$ fraction of the z's.

Note that N can be encoded using $O(\log n)$ bits. We say that n is *good* if such an N exists. By Lemma 35, there are infinitely many good n. The second part of the advice is then the bit $\alpha_A(N)$ needed to run the algorithm A.

We compute G_n as follow. Given $z := (z^{\mathsf{first}}, z^{\mathsf{second}})$, where $z^{\mathsf{first}} \in \{0,1\}^{n^{\varepsilon/2}}$ and $z^{\mathsf{second}} \in \{0,1\}^{O(\log n)}$, we first invoke the algorithm A from Lemma 35 on 1^N (where N is the first part of the advice as described in the previous paragraph) and z^{first} to obtain a truth table $T_{n,z^{\mathsf{first}}}$. We can compute A using MCSP as oracle and the bit $\alpha_A(N)$, which is the second part of the advice. We will then use the hardness-to-randomness construction in Lemma 34 on $T_{n,z^{\mathsf{first}}}$. Note that since $|T_{n,z^{\mathsf{first}}}| \leq n^{O(1)}$, the input length of $G_{\mathrm{Umans}}(T_{n,z^{\mathsf{first}}}, -)$ can be $O(\log n)$. Finally, we output

$$G_{\mathrm{Umans}}\big(T_{n,z^{\mathsf{first}}}, z^{\mathsf{second}}\big)_{[n]}.$$

It is easy to verify that G_n satisfies the efficiency condition stated in Theorem 33, since A is a pseudodeterministic polynomial-time algorithm and $G_{\mathrm{Umans}}(T_{n,z^{\mathsf{first}}}, -)$ runs in time $\mathrm{poly}(n)$ deterministically.

Next, we show the pseudorandomness condition of G_n. Note that for any good n, with probability at least $1 - 1/n^{c'}$ over z^{first}, the algorithm A will output (with high probability) some fixed truth table $T_{n,z^{\mathsf{first}}}$ with circuit complexity at least $n^{c'g}$. Whenever $T_{n,z^{\mathsf{first}}}$ has such circuit complexity, $G_{\mathrm{Umans}}\big(T_{n,z^{\mathsf{first}}}, -\big)$, by Lemma 34, will be a PRG that $(1/n^{c'})$-fools size-$(n^{c'})$ circuits. In order words, with probability at least $1 - 1/n^{c'}$ over z^{first}, for every circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size at most $n^{c'}$,

$$\left| \Pr_{x \sim \{0,1\}^n}[C(x) = 1] - \Pr_{z^{\mathsf{second}}}[C(G_{\mathrm{Umans}}(T_{n,z^{\mathsf{first}}}, z^{\mathsf{second}})) = 1] \right| \leq \frac{1}{n^{c'}}.$$

Then by a union bound, we get

$$\left| \Pr_{x \sim \{0,1\}^n}[C(x) = 1] - \Pr_{z}[C(G_s(z)) = 1] \right| \leq \frac{1}{n^{c'}} + \frac{1}{n^{c'}} \leq \frac{1}{n^d},$$

as desired. ◀

3.3.2 Circuit Lower Bound for $\mathrm{rK}^{\mathrm{poly},\mathrm{MCSP}}$

We complete the proof of Theorem 32 using our PRG in Theorem 33.

Proof of Theorem 32. Let t be a polynomial specified later. For the sake of contradiction, suppose there exists a circuit $C\colon \{0,1\}^n \to \{0,1\}$ of size at most n^c such that for all $x \in \{0,1\}^n$,

- if $\mathrm{rK}^{t,\mathrm{MCSP}}(x) \leq n^{\varepsilon}$ then $C(x) = 1$, and
- if $\mathrm{rK}^{t,\mathrm{MCSP}}(x) \geq n - 10$ then $C(x) = 0$.

Note that by a simple counting argument, for most $x \in \{0,1\}^n$, we have $\mathrm{rK}^{t,\mathrm{MCSP}}(x) \geq n - 10$. This gives

$$\Pr_{x \sim \{0,1\}^n}[C(x) = 0] \geq \frac{1}{2}.$$

Consider the PRG in Theorem 32 instantiated with parameters $\varepsilon/2$, c and $d := 1$. We have that for infinitely many n, G_n $(1/n)$-fools circuits of size at most n^c. On the one hand, this implies

$$\Pr_{z\sim\{0,1\}^{n^{\varepsilon/2}}}[C(G_n(z)) = 0] \geq \Pr_{x\sim\{0,1\}^n}[C(x) = 0] - \frac{1}{n} \geq \frac{1}{3}. \tag{1}$$

On the other hand, since G_n can be computed, with oracle access to MCSP and with $O(\log n)$ bits of advice, pseudodeterministically in time $\mathrm{poly}(n)$, we have that there exists some polynomial t with $t(n) = \mathrm{poly}(n)$ such that for every $z \in \{0,1\}^{n^{\varepsilon/2}}$

$$\mathrm{rK}^{t,\mathrm{MCSP}}(G_s(z)) \leq n^{\varepsilon/2} + O(\log n) < n^{\varepsilon},$$

which by our assumption implies that

$$\Pr_{z\sim\{0,1\}^{\varepsilon/2}}[C(G_n(z)) = 1] = 1.$$

This contradicts Equation (1). ◀

3.4 Pseudodeterministic Construction for Range Avoidance

In this subsection, we show a pseudodeterministic algorithm, using an NP oracle and an advice string of sub-polynomial length, that solves the range avoidance problem infinitely often.

▶ **Theorem 36.** *For every $\varepsilon > 0$ and every $d \geq 1$, there is a probabilistic polynomial-time algorithm A such that the following holds.*

- *For every $m \in \mathbb{N}$, on any input circuit $C\colon \{0,1\}^m \to \{0,1\}^{m+1}$ of size m^d, the algorithm A, with access to an NP oracle and with an advice $\alpha(m) \in \{0,1\}^{m^{\varepsilon}}$,[8] outputs with high probability a string $x_C \in \{0,1\}^{m+1}$, and*
- *for infinitely many $m \in \mathbb{N}$, $x_C \notin \mathsf{Range}(C)$ for each input C.*

The following lemma is implicit in [51], attributed to [43].

▶ **Lemma 37** (Adaption of [51])**.** *There exists an algorithm R that runs in deterministic polynomial time with an NP oracle such that the following holds. For every $m \in \mathbb{N}$, given any circuit $C\colon \{0,1\}^m \to \{0,1\}^{m+1}$ and a truth table of length 2^n with circuit complexity at least $2^{n/2}$, where $n := 4\lceil \log(m \cdot |C|)\rceil + \lfloor \log m \rfloor$, the algorithm R outputs a string $x \in \{0,1\}^{m+1}$ such that $x \notin \mathsf{Range}(C)$.*

Proof Sketch. The proof is adapted from those of [51, Lemma 3 and Theorem 7].

First of all, it can be shown that for every $m \in \mathbb{N}$ and for every circuit $C\colon \{0,1\}^m \to \{0,1\}^{m+1}$, there exists a circuit $D\colon \{0,1\}^m \to \{0,1\}^{2m}$, which can be constructed efficiently from C, of size $m \cdot |C|$, such that given a string $y \in \{0,1\}^{2m}$ that is not in the range D and given access to an NP oracle, one can compute efficiently a string $x \in \{0,1\}^{m+1}$ that is not in the range of C. (See the proof of [51, Lemma 3].) Therefore, to show the lemma, it suffices to show how to find, for a given circuit D of size $m \cdot |C|$ mapping m bits to $2m$ bits, a string that is not in the range of D.

Given a circuit $D\colon \{0,1\}^m \to \{0,1\}^{2m}$, let $k := 4\lceil \log(|D|)\rceil$. It can be shown that one can efficiently construct a circuit $E\colon \{0,1\}^m \to \{0,1\}^{2^k m}$ with the following properties. Given a $(2^k m)$-bit string that is not in the range of E and given access to an NP oracle,

[8] We stress that the advice $\alpha(m)$ does *not* depend on the input circuit C.

we can compute in time $\mathrm{poly}(|D|)$ a string $x \in \{0,1\}^{2m}$ that is not in the range of D. Moreover, every string $y \in \{0,1\}^{2^k m}$ in the range of E has small circuit complexity, in the sense that there is a circuit $F_y \colon \{0,1\}^{k+\lceil \log m \rceil} \to \{0,1\}$ of size at most $O(|D| \log |D|)$ that computes a truth table (of length $2^{k+\lceil \log m \rceil}$) whose prefix is y. (See [51, Figure 1 and Proof of Theorem 7].)

Now if we have a truth table $T \in \{0,1\}^{2^n}$ with circuit complexity at least $2^{n/2}$, where $n = k + \lfloor \log m \rfloor$, we can construct a $(2^k m)$-bit string

$$y_T := T \circ 0^{2^k m - 2^n}.$$

For the sake of contradiction, suppose y_T is in the range of E. Then by the discussion in the previous paragraph, it is easy to construct a circuit from F_{y_T} of size at most $O(|D| \log |D|)$ that computes T, which has circuit complexity at least $2^{n/2} = 2^{(4\lceil \log(|D|) \rceil + \lfloor \log m \rfloor)/2} \geq |D|^2$. This gives a contradiction. ◀

We are now ready to show Theorem 36.

Proof of Theorem 36. The idea is to use Lemma 31 to obtain a hard truth table, and plug it in Lemma 37 to solve the range avoidance problem.

Let $\varepsilon_0 := \varepsilon/(10d)$, and let B be the pseudodeterministic algorithm from Lemma 31 instantiated with parameter ε_0.

We say that $m \in \mathbb{N}$ is *good* if for $n := 4\lceil (d+1) \log m \rceil + \lfloor \log m \rfloor$, the algorithm B on input 1^n, with oracle access to MCSP and with $2^{\varepsilon_0 n}$ bits of advice, outputs with high probability some fixed truth table of circuit complexity greater than $2^n/n$. Note that there are infinitely many good m, since there are infinitely many n such that $B(1^n)$ will successfully output a fixed hard truth table with high probability.

Consider the following algorithm A. Given a circuit $C \colon \{0,1\}^m \to \{0,1\}^{m+1}$ of size m^d, A first runs $B(1^n)$, where $n := 4\lceil (d+1) \log m \rceil + \lfloor \log m \rfloor$, using an NP-complete oracle (that can simulate MCSP) and using an advice string of length $2^{\varepsilon_0 n} \leq m^{\varepsilon}$, to obtain (with high probability) a fixed truth table of length 2^n. We then run the algorithm R from Lemma 37 using this truth table and using the NP-complete oracle again to get a string $x \in \{0,1\}^{m+1}$. Note that whenever m is good (and sufficiently large), the truth table output by $B(1^n)$ will have circuit complexity greater than $2^n/n \geq 2^{n/2}$, so using such a hard truth table the algorithm R will successfully output a string that is not in the range of C. ◀

4 Barriers for Derandomization under Uniform Assumptions

4.1 PSPACE-Relativizing Derandomization under Uniform Assumptions

We show that previous results on derandomization under uniform assumptions are PSPACE-relativizing. This includes the subexponential-time derandomization of Impagliazzo and Wigderson [42] (see also [70]) and the recent "unstructured hardness to average-case randomness" [19]. These results only achieve average-case derandomization, therefore it is helpful to introduce the following definition.[9]

[9] In the literature, it is more common to define heur-P as a family of *distributional* problems, which are pairs $(L, \mathcal{D})$ where L is a language and $\mathcal{D}$ is a distribution. In this paper, we only care about whether our derandomization succeeds on all polynomial-time samplable distributions. Thus, for simplicity, we define heur-P as simply a class of languages that admits polynomial-time heuristics over all polynomial-time samplable distributions.

▶ **Definition 38** (Average-Case Complexity Classes). *We say a language L is in* heur-P *if for every polynomial-time samplable distribution $\mathcal{D}$ and every polynomial p, there is a deterministic polynomial-time algorithm $\mathcal{A}$ such that*

$$\Pr_{x\sim\mathcal{D}}[\mathcal{A}(x) = L(x)] \geq 1 - 1/p(n).$$

Similarly, if the algorithm $\mathcal{A}$ runs in quasi-polynomial time, then we say $L \in$ heur-QuasiP; if for every $\varepsilon > 0$ there is such an algorithm $\mathcal{A}$ running in 2^{n^ε} time, then we say $L \in$ heur-SUBEXP.

We start by demonstrating that the techniques in [42] are PSPACE-relativizing:

▶ **Proposition 7** (Uniform Derandomization in [42] is PSPACE-Relativizing). *Let $\mathcal{O} \in$ PSPACE. If $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]} \neq \mathsf{BPP}^{\mathcal{O}}$, then $\mathsf{BPP}^{\mathcal{O}} \subseteq$ i. o. heur-$\mathsf{SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$.*

Proof Sketch. We first prove that if $\mathsf{PSPACE}^{\mathcal{O}} \neq \mathsf{BPP}^{\mathcal{O}}$, then $\mathsf{BPP}^{\mathcal{O}} \subseteq$ i. o. heur-$\mathsf{SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$. Observe that since $\mathcal{O} \in \mathsf{PSPACE}$, we have $\mathsf{PSPACE}^{\mathcal{O}} = \mathsf{PSPACE}$. Let L be a $\mathsf{PSPACE}^{\mathcal{O}}$-complete problem that is both downward self-reducible and random self-reducible (as constructed in [70]). One could define a family of pseudorandom generators $G_n : \{0,1\}^{n^{o(1)}} \to \{0,1\}^n$ based on the hardness of L such that the following holds: Let D be any oracle such that for every $n \in \mathbb{N}$,

$$|\Pr[D(\mathcal{U}_n) = 1] - \Pr[D(G_n(\mathcal{U}_{n^{o(1)}})) = 1]| > 1/n,$$

then L can be decided in randomized polynomial time with a D oracle. Moreover, G_n is computable in $2^{n^{o(1)}}$ time. If $\mathsf{BPP}^{\mathcal{O}} \not\subseteq$ i. o. heur-$\mathsf{SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$, then in particular G_n does not fool certain BPP algorithm (on average, on every input length). It follows from the reconstruction properties of $\{G_n\}$ that $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$.

We consider two cases. Suppose that $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]} \not\subseteq \mathsf{P}^{\mathcal{O}}/_{\mathrm{poly}}$, then it follows from standard relativizing hardness-randomness trade-offs [59, 41] that $\mathsf{BPP}^{\mathcal{O}} \subseteq$ i. o. $\mathsf{SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$. On the other hand, if $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]} \subseteq \mathsf{P}^{\mathcal{O}}/_{\mathrm{poly}}$, then by the relativizing Karp-Lipton theorem [48], $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]} \subseteq (\Sigma_2^p)^{\mathcal{O}} \subseteq \mathsf{PSPACE}^{\mathcal{O}}$. Since $\mathsf{PSPACE}^{\mathcal{O}} \neq \mathsf{BPP}^{\mathcal{O}}$, it follows (again) that $\mathsf{BPP}^{\mathcal{O}} \subseteq$ i. o. heur-$\mathsf{SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$. ◀

Before we consider [19], it is helpful to define the notion of *logspace-uniform* low-depth circuits:

▶ **Definition 39** (Logspace-Uniform Circuits, see [19, Definition 3.5]). *We say that a circuit family $\{C_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ of size $T(n)$ is* logspace-uniform *if there exists a machine M that on input 1^n runs in space $O(\log T(n))$ and prints C_n. For two functions $T(n)$ and $d(n)$, we denote the class of languages computable by logspace-uniform circuits of size $T(n)$ and depth $d(n)$ by* lu-$\mathsf{CKT}[T, d]$.

We show the following PSPACE-relativizing version of [19, Theorem 5.2]. Note that we can only achieve quasi-polynomial time derandomization if we want a PSPACE-relativizing result; this is because an oracle in PSPACE might require $2^{\mathrm{poly}(n)}$ time (instead of $2^{O(n)}$ time) to compute, and consequently, the HSG in [19] has seed length polylog(n) (instead of $O(\log n)$). It might be possible to obtain a $\mathsf{SPACE}[n]$-relativizing result that achieves polynomial-time derandomization, but we do not pursue this direction here.

▶ **Theorem 40** (High-End Uniform Derandomization in [19] is PSPACE-Relativizing)**.** *Let $\mathcal{O} \in \mathsf{PSPACE}$ and $\varepsilon > 0$ be a constant. Assume there is a function L computable by logspace-uniform circuits of size $2^{\mathrm{poly}(n)}$ and depth $2^{n^{o(1)}}$, making $\mathcal{O}$-oracle queries of $\mathrm{poly}(n)$ length, such that $L \notin \mathsf{i.o.BPTIME}\left[2^{n^{\varepsilon}}\right]^{\mathcal{O}}$. (That is, $\mathsf{lu\text{-}CKT}^{\mathcal{O}[\mathrm{poly}]}\left[2^{\mathrm{poly}(n)}, 2^{n^{o(1)}}\right] \not\subseteq \mathsf{i.o.BPTIME}^{\mathcal{O}}\left[2^{n^{\varepsilon}}\right]$.) Then*

$$\mathsf{RP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}QuasiP}^{\mathcal{O}} \quad \textit{and} \quad \mathsf{BPP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}QuasiP}^{\mathcal{O}}/_{\log^{o(1)} n}.$$

Proof Sketch. The key observation is that since $\mathcal{O} \in \mathsf{PSPACE}$, $\mathcal{O}$ itself admits a logspace-uniform circuit of size $2^{\mathrm{poly}(n)}$ and depth $\mathrm{poly}(n)$. It follows that the hard language L in our assumption can actually be computed by a logspace-uniform circuit of size $2^{\mathrm{poly}(n)}$ and depth $2^{n^{o(1)}}$ *without any oracles*. That is:

$$L \in \mathsf{lu\text{-}CKT}^{\mathcal{O}[\mathrm{poly}]}\left[2^{\mathrm{poly}(n)}, 2^{n^{o(1)}}\right] \subseteq \mathsf{lu\text{-}CKT}\left[2^{\mathrm{poly}(n)}, 2^{n^{o(1)}}\right].$$

The rest of the proof follows [19, Theorem 5.2] closely. In what follows, the input length of the hard problems with low-depth circuits will be denoted as ℓ instead of n. Let $L^{(1)} \in \mathsf{lu\text{-}CKT}\left[2^{O(\ell)}, 2^{\ell^{o(1)}}\right]$ be an instance-checkable problem from [19, Proposition 4.4] such that L polynomial-time reduces to $L^{(1)}$.[10] The instance checker for $L^{(1)}$ runs in $2^{\ell^{o(1)}}$ time. By [19, Lemma A.1], there exists a constant $\varepsilon' > 0$ and a language $L' \in \mathsf{lu\text{-}CKT}\left[2^{O(\ell)}, 2^{\ell^{o(1)}}\right]$ that is not in $\mathsf{i.o.BPTIME}^{\mathcal{O}}\left[2^{\ell^{\varepsilon'}}\right]$ and has an instance checker running in $2^{\ell^{o(1)}}$ time.

The next step is to create an HSG from the hard function L'. Let $\mathcal{A}$ be a probabilistic algorithm which we want to derandomize, n be its input length, and $M := n^k$ be the amount of randomness it uses. Let $\ell := \lceil \log^{2/\varepsilon'} n \rceil$, $f : \{0,1\}^n \to \{0,1\}^{2^\ell}$ be the multi-output function that discards the input and outputs the truth table of $L'_\ell := L' \cap \{0,1\}^\ell$. We use [19, Theorem 4.5] with parameter $\delta := \ell^{\varepsilon'/4-1}$ to obtain a generator G_f and a reconstruction algorithm R.

- The generator G_f runs in time $T^{O(1/\delta)} \leq 2^{\ell^2} \leq 2^{\mathrm{polylog}(n)}$ and outputs a list of M-bit strings.
- The reconstruction algorithm R gets oracle access to a function $D : \{0,1\}^M \to \{0,1\}$, and runs in time $2^{\ell^{o(1)}} \cdot T^\delta \leq 2^{\ell^{\varepsilon'/3}}$. Assume that D $(1/M)$-avoids the generator G_f, then w.p. $\geq 1 - \frac{O(\log^2 T)}{T}$, R^D prints an oracle circuit C_f such that the truth table of C_f^D is exactly L'_ℓ.

Given an $\mathsf{RP}^{\mathcal{O}}$ algorithm $\mathcal{A}$, we use the HSG to fool $\mathcal{A}$, making it run in deterministic $T^{O(1/\delta)} \leq 2^{\mathrm{polylog}(n)}$ time. Now suppose there is a distribution $\mathcal{D}$ samplable in polynomial time with $\mathcal{O}$ oracles, as well as an infinite set of input lengths $\mathcal{I} \subseteq \mathbb{N}$ such that our derandomization fails on $\mathcal{I}$ w.p. $1/\mathrm{poly}(n)$. This means that w.p. at least $1/\mathrm{poly}(n)$ over $x \sim \mathcal{D}$, we have $\Pr_r[A(x,r) = 1] \geq 1/2$ but $A(x,w) = 0$ for every w in the HSG.

We will use a reconstruction argument to compute L'_ℓ in probabilistic $2^{\ell^{\varepsilon'}}$ time with an $\mathcal{O}$ oracle, reaching a contradiction. We say an input length ℓ is nice if there is some $n \in \mathcal{I}$ such that $\ell = \lceil \log^{2/\varepsilon'} n \rceil$; there are infinitely many nice input lengths. On input $x \in \{0,1\}^\ell$,

[10] Proposition 4.4 of [19] only claimed to hold for $L \in \mathsf{lu\text{-}CKT}\left[2^{O(\ell)}, 2^{\ell^{o(1)}}\right]$ instead of $\mathsf{lu\text{-}CKT}\left[2^{\mathrm{poly}(\ell)}, 2^{\ell^{o(1)}}\right]$. We can use a padding argument to reduce L to some problem in $\mathsf{lu\text{-}CKT}\left[2^{O(\ell)}, 2^{\ell^{o(1)}}\right]$ and then invoke this proposition; the only difference is that the reduction runs in polynomial time instead of linear time.

where ℓ is a nice input length, we sample uniformly at random an input length n such that $\ell = \lceil \log^{2/\varepsilon'} n \rceil$, and sample $x \sim \mathcal{D}_n$. With probability $1/\mathrm{poly}(n)$, the following good event (which we denote $\mathcal{G}$) happens: n is good and $A(x,-)$ is a distinguisher for our hitting set. Then we invoke the instance checker for L'_ℓ on input x. Whenever the instance checker asks a query q, we use $R^{A(x,-)}$ to find the answer of $L'(q)$. If $\mathcal{G}$ happens then w.h.p. the instance checker outputs the correct answer $L'(x)$; even if $\mathcal{G}$ does not happen, w.h.p. the instance checker outputs either $L'(x)$ or $\perp$. By sampling n and x for $\mathrm{poly}(\ell)$ times, w.h.p. $\mathcal{G}$ will happen at least once, and we compute $L'_\ell(x)$ successfully. Since there are $\mathrm{poly}(n)$ samples, the instance checker runs in $2^{\ell^{o(1)}}$ time, the oracle $A(x,-)$ runs in $\mathrm{poly}(n)$ time, and the reconstruction algorithm runs in $2^{\ell^{\varepsilon'/3}}$ time, the total running time is

$$\mathrm{poly}(n) \cdot 2^{\ell^{o(1)}} \cdot \mathrm{poly}(n) \cdot 2^{\ell^{\varepsilon'/3}} \ll 2^{\ell^{\varepsilon'}}.$$

The proof for $\mathsf{BPP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}QuasiP}^{\mathcal{O}}/_{\log^{o(1)} n}$ is similar. The only difference is to notice that the targeted HSG constructed in [19, Theorem 4.5] is also a *targeted somewhere-PRG*, consisting of $2^{\ell^{o(1)}}$ sub-PRGs one of which is guaranteed to be secure. The advice for the derandomized algorithm is the index of the sub-PRG that is secure, thus has length $\log(2^{\ell^{o(1)}}) \le \log^{o(1)} n$. ◀

▶ **Proposition 8** (Uniform Derandomization in [19] is PSPACE-Relativizing)**.** *Let* $\mathcal{O} \in \mathsf{PSPACE}$ *and suppose that* $\mathsf{PSPACE}^{\mathcal{O}} \not\subseteq \mathsf{i.o.}\,\mathsf{BPP}^{\mathcal{O}}$*. Then*

$$\mathsf{RP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]} \quad \textit{and} \quad \mathsf{BPP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]}/_{O(\log n)}.$$

Proof Sketch. The proof outline is the same as Theorem 40, but the parameters are a bit different. In particular, the hard language L' is in $\mathsf{lu\text{-}CKT}\left[2^{O(\ell)}, \mathrm{poly}(\ell)\right] \setminus \mathsf{BPP}^{\mathcal{O}}$. Fix any constant $\varepsilon > 0$, the HSG has the following parameters:

$$\ell := n^{\varepsilon/3}, \quad M := n^k, \quad \text{and } \delta := (3/\varepsilon)\log \ell/\ell.$$

The generator runs in $T^{O(1/\delta)} \le 2^{O(\ell^2)} \le 2^{n^\varepsilon}$ time and outputs a list of M-bit strings. The reconstruction algorithm runs in $O(nT^\delta) \le \mathrm{poly}(\ell)$ time. If the generator fails to fool a certain $\mathsf{RP}^{\mathcal{O}}$ algorithm (on average and infinitely often), then we can use the reconstruction algorithm to compute L' in $\mathsf{BPP}^{\mathcal{O}}$, contradicting our hypothesis. Therefore the generator successfully fools every $\mathsf{RP}^{\mathcal{O}}$ algorithm (on average and infinitely often), and thus $\mathsf{RP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]}$.

Moreover, the generator is a somewhere PRG: It can be partitioned into $d' := \mathrm{poly}(\ell)$ "sub-PRGs" such that for every $\mathsf{BPP}^{\mathcal{O}}$ algorithm, one of the "sub-PRG" fools this algorithm successfully (on average and infinitely often). We embed the index of the correct "sub-PRG" as advice, taking $\log d' = O(\log n)$ bits. Therefore $\mathsf{BPP}^{\mathcal{O}} \subseteq \mathsf{heur\text{-}SUBEXP}^{\mathcal{O}[\mathrm{poly}]}/_{O(\log n)}$. ◀

4.2 Bounded-Relativization Barriers for Uniform Derandomization

4.2.1 Barrier for Worst-Case Derandomization under Uniform Assumptions

Recall that [42] showed that if $\mathsf{EXP} \neq \mathsf{BPP}$ then $\mathsf{BPP} \subseteq \mathsf{i.o.}\,\mathsf{heur\text{-}SUBEXP}$. We present some evidence that current techniques cannot improve this result to worst-case derandomization:

▶ **Theorem 10.** *There is an oracle* $\mathcal{O} \in \mathsf{EXPH}$ *such that*

$$\mathsf{RP}^{\mathcal{O}} \not\subseteq \mathsf{i.o.}\,\mathsf{DTIME}^{\mathcal{O}}[2^n] \quad \textit{and} \quad \mathsf{UP}^{\mathcal{O}} \not\subseteq \mathsf{BPTIME}^{\mathcal{O}}[2^n].$$

Note that $\mathsf{UP} \not\subseteq \mathsf{BPTIME}[2^n]$ is much stronger than $\mathsf{EXP} \neq \mathsf{BPP}$, yet it still does not imply fixed exponential time worst-case derandomization of RP by EXPH-relativizing techniques.

▶ **Corollary 41.** *If we can use* PSPACE*-relativizing techniques to show*

$$\mathsf{EXP} \neq \mathsf{BPP} \implies \mathsf{BPP} \subseteq \mathsf{i.o.SUBEXP},$$

then $\mathsf{L} \neq \mathsf{NP}$ *follows.*

Proof of Theorem 10. Our oracle world will consist of two oracles $\mathcal{P}$ and $\mathcal{Q}$ (it is easy to combine them into a single oracle). The oracle $\mathcal{P}$ creates a hard problem in RP against deterministic algorithms, and the oracle $\mathcal{Q}$ creates a hard problem in UP against probabilistic algorithms. In particular, every input to the oracle $\mathcal{P}$ will be of the form (x, r) where $|r| = 100|x|$, and let

$$L_{\mathcal{P}} := \left\{ x : \exists r \in \{0,1\}^{100|x|}, \mathcal{P}(x, r) = 1 \right\}.$$

We will guarantee that for every x, $\Pr_r[\mathcal{P}(x, r) = 1]$ is either 0 or greater than $1/2$, thus $L_{\mathcal{P}} \in \mathsf{RP}$. We also define

$$L_{\mathcal{Q}} := \left\{ 1^n : \exists r \in \{0,1\}^{100n}, \mathcal{Q}(r) = 1 \right\}.$$

We will guarantee that for every input length $100n$, $|\mathcal{Q} \cap \{0,1\}^{100n}| \leq 1$, thus $L_{\mathcal{Q}} \in \mathsf{UP}$.

Let $M_1, M_2, \ldots$ be an enumeration of Turing machines running in deterministic 2^n time, and $N_1, N_2, \ldots$ be an enumeration of probabilistic Turing machines running in 2^n time (that does not necessarily satisfy the BPP promise).[11] We want every M_i to fail to solve $L_{\mathcal{P}}$ on all but finitely many input lengths, and every N_i to fail to solve $L_{\mathcal{Q}}$ on infinitely many input lengths. In particular, let n_1 be a large enough constant, and $n_i = 2^{500 n_{i-1}}$ for each $i \geq 2$. Assuming that every machine appears in the sequence $\{N_i\}$ infinitely often, we want that for each $i \in \mathbb{N}$, N_i fails to solve $L_{\mathcal{Q}}$ on input 1^{n_i}.

We also need the following terminologies. The *n-th slice* of $\mathcal{P}$ is the set of inputs of the form (x, r) where $|x| = n$ and $|r| = 100n$. The *ℓ-th slice* of $\mathcal{Q}$ is simply the set of inputs with length ℓ. During our construction, there will be some entries $\mathcal{P}(x, r)$ and $\mathcal{Q}(r)$ that are *fixed*, meaning that their values will never change in the subsequent construction. There will also be some inputs x that are *frozen*, meaning that for every $r \in \{0,1\}^{100|x|}$, the values of $\mathcal{P}(x, r)$ are fixed and will never change in the subsequent construction. If an entry $\mathcal{P}(x, r)$ or $\mathcal{Q}(r)$ is not fixed and, in the case of $\mathcal{P}(x, r)$, x is not frozen, then we say this entry is *free*.

Our construction proceeds in stages. During stage n, we will fix the entire n-th slices of $\mathcal{P}$ and $\mathcal{Q}$; we will possibly also fix or freeze some other entries. It will be guaranteed that before the n-th stage:

1. the number of frozen strings beyond the n-th slice (including the n-th slice) is at most $2^{n/2}$;
2. the number of fixed strings (except the frozen ones) beyond the n-th slice (including the n-th slice) is at most 2^{5n}; and
3. each fixed or frozen entry beyond the n-th slice (including the n-th slice) is set to be 0.

[11] In this paper, whenever we enumerate a list of Turing machines, we assume every machine occurs in the list *infinitely many times*.

Diagonalization against M_i. Let $x_1, x_2, \ldots, x_n$ be n inputs of length n that are not frozen. Since there are at most $2^{n/2}$ frozen inputs of length n, it is always possible to select n unfrozen inputs. For each $i \in [n]$, we simulate the machine M_i on input x_i. Whenever M_i asks a query $\mathcal{P}(x, r)$ or $\mathcal{Q}(r)$, if this query is not free, then we return the corresponding value fixed before; if it is free, then we return 0 as the answer and also *fix* this query to be 0. Then M_i will output an answer $ans_i \in \{0, 1\}$. Note that there are at most $2^{5n} + n \cdot 2^n \ll 2^{100n}/3$ entries in $\mathcal{P}$ that are fixed, thus the vast majority of the entries $\mathcal{P}(x_i, r)$ are free. We fix all these entries $\mathcal{P}(x_i, r)$ to be $(1 - ans_i)$. It is easy to see that M_i fails to compute $L_{\mathcal{P}}$ on input x_i.

Diagonalization against N_i. If $n = 10n_i$ for some $i \in \mathbb{N}$, then we also need to fix the $100n_i$-th slice of $\mathcal{Q}$ so that N_i fails to compute $L_{\mathcal{Q}}$ on input 1^{n_i}. (In stage n, this happens after the above diagonalization against M_i.) We simulate N_i on input 1^{n_i}. Note that N_i is a probabilistic machine running in time 2^{n_i}, thus it has $B := 2^{2^{n_i}}$ computational branches, where each branch has $1/B$ probability mass. On each branch, whenever N_i asks a query $\mathcal{P}(x, r)$ or $\mathcal{Q}(r)$, if this query is not free, then we return the corresponding value fixed before; if it is free, then we return 0 as the answer to the query. Note that we cannot fix the query as there are potentially as many as B queries to fix. Instead, after simulating all the computational branches, we pick out the "heavy" queries that occur in a lot of branches and fix them:

- A string x is *heavy* if at least $\frac{1}{10^4|x|^3}$ fraction of branches queried some free entry of the form $\mathcal{P}(x, r)$.
- An entry $\mathcal{Q}(r)$ is *heavy* if at least $\frac{1}{10^4}$ fraction of branches queried $\mathcal{Q}(r)$.

We *freeze* every heavy x by setting every unfixed entry $\mathcal{P}(x, r)$ to be 0. We also *fix* every heavy $\mathcal{Q}(r)$ to be 0. Note that each computational branch of N_i only makes at most 2^{n_i} queries. Therefore, for each input length m where $n \leq m \leq 2^{n_i}$, the number of heavy strings $x \in \{0, 1\}^m$ is at most $10^4 2^{n_i} m^3$. The number of entries $\mathcal{Q}(r)$ fixed before this stage is at most 2^{5n}, and the number of heavy entries $\mathcal{Q}(r)$ is at most $10^4 2^{n_i}$, therefore the total number of fixed entries in $\mathcal{Q}$ after this stage is at most $2^{6n} \ll 2^{100n_i}$.

After simulating every computational branch of N_i, let p be the probability that N_i outputs 1 on input 1^n. If $p < 0.5$, then we pick a free $r \in \{0, 1\}^{100n_i}$ and set $\mathcal{Q}(r) = 1$; since the number of fixed entries $\mathcal{Q}(r)$ is $\ll 2^{100n_i}$, such r always exists. If $p \geq 0.5$, then we do nothing.

Now we have that N_i solves $L_{\mathcal{Q}}$ on input 1^{n_i} with probability less than 2/3 (actually, at most 1/2). Is this always the case in the future? Consider how the future changes to the oracles might affect the behavior of N_i. Let m be an input length where $n < m \leq 2^n$. During stage m (which has not happened yet), we will pick m inputs $x_1, x_2, \ldots, x_m \in \{0, 1\}^m$, then some entries of the form $\mathcal{P}(x_i, r)$ will be set to be 1. Also note that the probability mass of computational branches that each x_i affects is at most $\frac{1}{10^4 m^3}$. Therefore, the total probability mass of computational branches affected is at most

$$\sum_{m \in \mathbb{N}} \frac{m}{10^4 m^3} < \frac{1}{10^3}.$$

Note that we also set at most one entry $\mathcal{Q}(r)$ to be 1, where $|r| = 100n_i$. (The next time we diagonalize against N_{i+1} happens in stage $n_{i+1} \gg 2^n$.) Since this entry $\mathcal{Q}(r)$ is not heavy, the probability mass of computational branches that it affects is at most $\frac{1}{10^4}$.

It follows that only an $\frac{11}{10^4}$ fraction of computational branches of N_i will be affected in the future. That is, let p' be the probability that N_i outputs 1 on input 1^{n_i} at the end of our construction (i.e., after stage $\gg 2^n$), then $|p' - p| < \frac{11}{10^4}$. Therefore, it cannot be the case that

$\Pr[N_i(1^{n_i}) = L_{\mathcal{Q}}(1^{n_i})] \geq 2/3$.

Clean-up. At the end of stage n, we fix every unfixed input $\mathcal{P}(x, r)$ in the n-th slice to be 0. If $n = 10n_i$ then we also fix every unfixed input $\mathcal{Q}(r)$ in the $10n$-th slice to be 0. The number of frozen strings is at most $10^4 2^{n/10} n^3 \leq 2^{n/2}$ (note that at most one N_i contributes to the frozen strings since the input lengths $\{n_i\}$ are very far apart). The number of fixed entries above the $(n+1)$-st slice is at most

$$2^{5n} + n \cdot 2^n + \sum_{m=n}^{2^{n/10}} 10^4 2^{n/10} m^3 < 2^{5(n+1)}.$$

The complexity of $\mathcal{O}$. We show that the above oracles $\mathcal{P}$ and $\mathcal{Q}$ are computable in EXPH. It suffices to present a deterministic algorithm with access to a Σ_3^p oracle that on input 1^n, outputs the truth tables of the n-th slices of $\mathcal{P}$ and $\mathcal{Q}$ in $2^{O(n)}$ time.

We simulate each stage $n' \leq n$. Before stage n', we maintain the set of fixed entries $\mathcal{P}(x, r)$, $\mathcal{Q}(r)$, as well as the set of frozen strings x. There are at most $2^{O(n')}$ many such strings. In what follows, we use $\langle \mathcal{P}, \mathcal{Q} \rangle$ to denote (the length-$2^{O(n')}$ encoding of) the list of fixed entries and frozen strings.

It is easy to diagonalize against each M_i: we simply choose the inputs $x_1, x_2, \ldots, x_{n'}$, simulate each M_i on input x_i, and fix the oracles accordingly. This takes $2^{O(n')}$ time.

To simulate N_i, we need to define the following language capturing the weight of queries. Consider simulating the probabilistic machine N_i on the current oracle $\langle \mathcal{P}, \mathcal{Q} \rangle$. Whenever N_i makes a query, if this query is not free, then we return the answer of the query; otherwise we return 0. The following two claims are immediate corollaries of Theorem 23:

▷ Claim 42. There is a language $\textsc{Heavy-String}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, x)$ computable in Σ_2^p such that:

- if at least $\frac{1.1}{10^4 |x|^3}$ fraction of computational branches of N_i queried some free entry of the form $\mathcal{P}(x, -)$, then $\textsc{Heavy-String}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, x) = 1$;
- if at most $\frac{0.9}{10^4 |x|^3}$ fraction of computational branches of N_i queried some free entry of the form $\mathcal{P}(x, -)$, then $\textsc{Heavy-String}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, x) = 0$.

▷ Claim 43. There is a language $\textsc{Heavy-Entry}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, r)$ computable in Σ_2^p such that:

- if at least $\frac{1.1}{10^4}$ fraction of branches of N_i queried $\mathcal{Q}(r)$, then $\textsc{Heavy-Entry}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, r) = 1$;
- if at most $\frac{0.9}{10^4}$ fraction of branches of N_i queried $\mathcal{Q}(r)$, then $\textsc{Heavy-Entry}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, r) = 0$.

(Note that both $\textsc{Heavy-String}$ and $\textsc{Heavy-Entry}$ accepts inputs of length $2^{O(n')}$.)

With the aid of these two languages, it is possible to simulate stage n' now. We say a string x is *heavy* if $\textsc{Heavy-String}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, x) = 1$, and an entry $\mathcal{Q}(r)$ is *heavy* if $\textsc{Heavy-Entry}(\langle \mathcal{P}, \mathcal{Q} \rangle, N_i, r) = 1$. Since there are at most $2^{O(n')}$ heavy strings, we can find all of them in $2^{O(n')}$ time with access to an $\mathsf{NP}^{\textsc{Heavy-String}}$ oracle. Similarly, since there are $2^{O(n')}$ heavy entries, we can find all of them in $2^{O(n')}$ time with access to an $\mathsf{NP}^{\textsc{Heavy-Entry}}$ oracle. We freeze these heavy strings and fix these heavy entries accordingly.

Finally, we estimate the probability that N_i outputs 1 (given current versions of $\mathcal{P}$ and $\mathcal{Q}$) in Σ_2^p, within additive error 0.1. If this estimation is less than $1/2$, then we pick a free $r \in \{0,1\}^{10n'}$ and set $\mathcal{Q}(r) = 1$; otherwise we do nothing.

At the end of stage n', we fix every unfixed entry in the n'-th slice of $\mathcal{P}(x,r)$ to be 0. If $n = 10n_i$ then we also fix every unfixed input $\mathcal{Q}(r)$ in the $10n$-th slice to be 0. This takes $2^{O(n')}$ time.

It is easy to see that simulating the n stages takes $2^{O(n)}$ time with access to a Σ_3^p oracle. ◀

▶ Remark 44. It is instructive to see why this construction only rules out worst-case derandomization instead of average-case derandomization. (If we could rule out average-case derandomization, then PSPACE $\neq$ EXPH follows from Proposition 7!)

The reason is that we do not want to affect the acceptance probability of $N_i(1^{n_i})$ by too much. Let $m \in [10n_i + 1, 2^{n_i}]$ be a "future" input length, we say $x \in \{0,1\}^m$ is "used for diagonalization" if there is some $\mathcal{P}(x,r)$ that is fixed to 1. Each string x used for diagonalization influences the accept probability of N_i by a small amount ($\frac{1}{10^4 m^3}$ in our proof), therefore we cannot afford to have too many strings used for diagonalization.

If we only want each M_i to fail in the worst case, we only need to use one input $x_i \in \{0,1\}^m$ to diagonalize against each M_i, hence we only need to use $\omega(1)$ strings x for diagonalization. On the other hand, if we want each M_i to fail on average (say, under the uniform distribution over $\{0,1\}^m$), then there has to be a lot of strings (e.g., $2^m/\mathrm{poly}(m)$) used for diagonalization; our previous guarantees on the acceptance probability of N_i will be completely destroyed.

4.2.2 Barrier for Almost-Everywhere Derandomization without Advice

It is natural to ask whether the results of [42] can be improved to *almost-everywhere* derandomization. Recently, substantial progress was made by Chen, Rothblum, and Tell [19], who showed average-case derandomization on *almost every input length* based on uniform assumptions. In particular, assuming $\mathsf{lu\text{-}CKT}[2^{O(n)}, 2^{o(n)}] \not\subseteq \mathsf{i.o.BPTIME}[2^{\Omega(n)}]$, they showed that $\mathsf{RP} \subseteq \mathsf{heur\text{-}P}$ and $\mathsf{BPP} \subseteq \mathsf{heur\text{-}P}/_{O(\log n)}$.

One open question is whether the $O(\log n)$-bit advice in the derandomization of BPP can be eliminated. In Theorem 11, we present an oracle world where this improvement is not possible. Unfortunately, we do not know how to implement this oracle in EXPH; nevertheless, we still show in Theorem 12 that eliminating the $O(\log n)$-bit advice by PSPACE-relativizing techniques would require proving new lower bounds for SAT.

▶ **Theorem 11.** *There is an oracle $\mathcal{O}$ such that*

$$\mathsf{BPP}^{\mathcal{O}} \not\subseteq \mathsf{heur\text{-}DTIME}^{\mathcal{O}}[2^n] \quad \textit{and} \quad \mathsf{UP}^{\mathcal{O}} \not\subseteq \mathsf{i.o.BPTIME}^{\mathcal{O}}[2^n].$$

Proof. We use a similar construction as in Theorem 10. The difference is that now we want a problem in BPP that is infinitely-often hard, and a problem in UP that is almost-everywhere hard, therefore we need to define the oracles differently. In particular, the oracle $\mathcal{P}$ only receives one input r, and the oracle $\mathcal{Q}$ receives two inputs (x,r), where $|r| = 10|x|$. The hard language in BPP is:

$$L_{\mathcal{P}} = \left\{ 1^n : \Pr_{r \sim \{0,1\}^{n^4}}[\mathcal{P}(r) = 1] \geq 1/2 \right\}.$$

It is guaranteed that for every n, the fraction of length-n^4 strings that are in $\mathcal{P}$ is either at most $1/3$ or at least $2/3$, therefore $L_{\mathcal{P}} \in \mathsf{BPP}$. On the other hand, the hard language in UP is:

$$L_{\mathcal{Q}} = \left\{ x : \exists r \in \{0,1\}^{10|x|}, \mathcal{Q}(x,r) = 1 \right\}.$$

It is guaranteed that for every input x, there is at most one string $r \in \{0,1\}^{10|x|}$ such that $\mathcal{Q}(x,r) = 1$, therefore $L_{\mathcal{Q}} \in \mathsf{UP}$. We will construct the oracles $\mathcal{P}$ and $\mathcal{Q}$ such that $L_{\mathcal{P}} \notin \mathsf{DTIME}[2^n]$ and $L_{\mathcal{Q}} \notin \mathsf{i.o.BPTIME}[2^n]$. Note that $L_{\mathcal{P}}$ is a unary language, therefore we also have $\mathsf{BPP} \not\subseteq \mathsf{heur\text{-}DTIME}[2^n]$.

We use the same terminologies as before: $M_1, M_2, \ldots$ is an enumeration of deterministic Turing machines running in 2^n time, and $N_1, N_2, \ldots$ is an enumeration of probabilistic Turing machines running in 2^n time. The ℓ-th *slice* of $\mathcal{P}$ is the set of inputs with length ℓ, and the n-th *slice* of $\mathcal{Q}$ is the set of inputs of the form (x,r) where $|x| = n$ and $|r| = 10n$. An entry is *fixed* if its value will never change in the subsequent construction. We choose n_1 to be a large enough constant and define a sequence $\{n_i\}$ where $n_i = 2^{10n_{i-1}}$ for each $i \geq 2$. We want every N_i to fail to solve $L_{\mathcal{Q}}$ on all but finitely many input lengths, and every M_i to fail to solve $L_{\mathcal{P}}$ on input 1^{n_i}.

Our construction proceeds in stages. We guarantee that:

- For every input length n that is not in the sequence $\{n_i\}$, the n-th slice of $\mathcal{P}$ is identically 0.
- For every input length n, there are at most $2n$ entries in the entire n-th slice of $\mathcal{Q}$ that returns 1.
- Consider only the oracle $\mathcal{Q}$. Before the n-th stage, the number of fixed entries beyond the n-th slice of $\mathcal{Q}$ (including the n-th one) is at most 2^{6n}, and all of them are fixed to be 0.

We start from the $n^\star$-th stage for some large enough constant $n^\star$. For each $n \geq n^\star$, in the n-th stage, we diagonalize against the machines N_i and M_i, and fix the n-th slice of $\mathcal{Q}$ as well as the n^4-th slice of $\mathcal{P}$ (when n equals some n_i). Details follow.

Diagonalization against N_i. Let n' be the smallest integer in the sequence $\{n_i\}$ such that $n' \geq n$, and i' be the index such that $n' = n_{i'}$. The $(n')^4$-th slice of $\mathcal{P}$ may contain either few or most strings, and we do not know which is the case yet. Therefore, we will diagonalize each N_i twice, first assuming the n'-th slice of $\mathcal{P}$ is nearly empty and then assuming the n'-th slice of $\mathcal{P}$ is nearly full. For this reason, we will need $2n$ distinct strings $x_{i,b} \in \{0,1\}^n$, one for each pair of $i \in [n]$ and $b \in \{0,1\}$. Note that at the beginning of the n-th stage, there are at most $2^{6n} \ll 2^{10n}$ strings fixed in the n-th slice of $\mathcal{Q}$, therefore we can always choose $2n$ strings $x_{i,b}$ where no entry of the form $\mathcal{Q}(x_{i,b}, r)$ is fixed.

For each $i \in [n]$ and $b \in \{0,1\}$, we simulate $N_i(x_{i,b})$, assuming that most entries in the n'-th slice of $\mathcal{P}$ returns b. Note that N_i is a probabilistic machine running in time 2^n, thus it has $B := 2^{2^n}$ computational branches, where each branch has $1/B$ probability mass. On each branch, whenever N_i asks a query $\mathcal{P}(r)$ or $\mathcal{Q}(x,r)$, if this query is already fixed, then we return the corresponding value fixed before; otherwise:

- Suppose the query is $\mathcal{P}(r)$. If $|r|$ is in the sequence $\{n_i^4\}$, then since $\mathcal{P}(r)$ is not fixed and $|r| \leq 2^n$, we know that $|r| = (n')^4$ and we return b as the answer. Otherwise ($|r| \neq (n')^4$) we return 0.
- Suppose the query is $\mathcal{Q}(x,r)$, then we return 0 as the answer.

Again, we can only afford to fix the *heavy* entries, defined as follows. Let K be the number of machine-input pairs that we still need to simulate in the future before fixing the $(n')^4$-th slice of $\mathcal{P}$; note that this includes both the current $N_i(x_{i,b})$ and the final $M_{i'}(1^{n'})$. That is:

$$K = 2(n-i) + (2-b) + \sum_{j=n+1}^{n'} (2j) + 1.$$

- An entry $\mathcal{P}(r)$ is *heavy* if $|r| = (n')^4$ and at least $2^{-(n'+4)K}$ fraction of branches queried $\mathcal{P}(r)$.
- An entry $\mathcal{Q}(x,r)$ is *heavy* if at least $\frac{1}{10^4|x|^3}$ fraction of branches queried $\mathcal{Q}(x,r)$.

We fix every heavy entry according to the way we answer this entry before. That is, every heavy entry $\mathcal{P}(r)$ is fixed to be b and every heavy entry $\mathcal{Q}(x,r)$ is fixed to be 0. The number of heavy entries in $\mathcal{P}$ is at most $2^{(n'+4)K} \cdot 2^n$; for each input length $m \geq n$, the number of heavy entries in the m-th slice of $\mathcal{Q}$ is at most $2^n \cdot 10^4 \cdot m^3$.

Let p be the probability (over the B computational branches) that $N_i(x_{i,b})$ outputs 1. If $p < 1/2$ then we find a string r such that $\mathcal{Q}(x_{i,b}, r)$ is unfixed yet and set $\mathcal{Q}(x_{i,b}, r) = 1$, making $x_{i,b} \in L_{\mathcal{Q}}$. Such a string r exists since we have fixed strictly less than 2^{10n} strings in the n-th slice of $\mathcal{Q}$. If $p \geq 1/2$ then we do nothing.

Assuming that most entries in the $(n')^4$-th slice of $\mathcal{P}$ indeed returns b, at this point the probability that N_i outputs the correct answer on input $x_{i,b}$ is at most $1/2$. However, the oracles $\mathcal{P}$ and $\mathcal{Q}$ might change in the future, thus the behavior of N_i might also change. What fraction of computational branches might be affected in the future?

- Consider the $K-1$ machine-input pairs that we will simulate in the future before the $(n')^4$-th slice of $\mathcal{P}$ is completely fixed. The number of heavy entries in the $(n')^4$-th slice of $\mathcal{P}$ that these machines fix is at most

  $$\sum_{k=1}^{K-1} 2^{(n'+4)k} \cdot 2^{n'} \leq 2 \cdot 2^{(n'+4)(K-1)} \cdot 2^{n'}.$$

  Therefore, the probability mass of computational branches of $N_i(x_{i,b})$ influenced by the future changes to the $(n')^4$-th slice of $\mathcal{P}$ is at most

  $$2 \cdot 2^{(n'+4)(K-1)} \cdot 2^{n'} \cdot 2^{-(n'+4)K} \leq 1/8.$$

- For each $m \leq 2^n$, there are at most $2m$ entries in the m-th slice of $\mathcal{Q}$ that we will fix to be 1. The probability mass of computational branches of $N_i(x_{i,b})$ influenced by these entries is at most

  $$\sum_{m=n}^{2^n} \frac{2m}{10^4 m^3} < \frac{1}{10^3}.$$

It follows that the future modifications to the oracles $\mathcal{P}$ and $\mathcal{Q}$ will only affect the accept probability of $N_i(x_{i,b})$ by $\frac{1}{8} + \frac{1}{10^3} < \frac{1}{6}$. Therefore, it cannot be the case that

$$\Pr[N_i(x_{i,b}) = L_{\mathcal{Q}}(x_{i,b})] \geq 2/3.$$

Diagonalization against M_i. If $n = n_i$ for some $i \in \mathbb{N}$, then we want that M_i does not compute $L_{\mathcal{P}}$ on input 1^n correctly. We simulate M_i on the input 1^n. When M_i asks a query $\mathcal{P}(r)$ or $\mathcal{Q}(x,r)$, if this query is already fixed, then we return the corresponding value fixed before; otherwise we return 0 and fix this queried entry.

After this simulation, we argue that only a small fraction of entries in the n^4-th slice of $\mathcal{P}$ are fixed. First, the simulation of $M_i(1^n)$ fixes at most 2^n entries. Second, there are at most $K = \sum_{j=1}^{n}(2j) = O(n^2)$ machine-input pairs $N_i(x_{i,b})$ simulated so far, and the k-th such machine fixes at most $2^{(n+4)k} \cdot 2^n$ entries. Therefore, the number of entries fixed by some previous $N_i(x_{i,b})$ is at most

$$\sum_{k=1}^{K} 2^{(n+4)k} \cdot 2^n \leq 2^{O(n^3)}.$$

It follows that all but $2^{O(n^3)}$ entries in the n^4-th slice of $\mathcal{P}$ are not fixed yet. Let $b \in \{0,1\}$ be the output bit of $M_i(1^n)$, then we set $L_{\mathcal{P}}(1^n) = 1 - b$ by fixing every unfixed entry in the n^4-th slice of $\mathcal{P}$ to be $1-b$. We have that on input 1^n, $L_{\mathcal{P}}$ satisfies the BPP promise, and M_i fails to solve $L_{\mathcal{P}}$.

Clear-up. At the end of stage n, we fix every unfixed input $\mathcal{Q}(x, r)$ on the n-th slice to be 0. It is easy to see that there are at most $2n$ entries in the n-th slice of $\mathcal{Q}(x, r)$ that returns 1.

During the n-th stage, the number of entries we fixed beyond the $(n+1)$-st slice of $\mathcal{Q}$ is at most

$$2^n + \sum_{m=n+1}^{2^n} 2^n \cdot 10^4 \cdot m^3 \leq 2^{5n}.$$

Thus the total number of entries we fixed beyond the $(n+1)$-st slice of $\mathcal{Q}$ is at most $2^{6n} + 2^{5n} < 2^{6(n+1)}$. ◀

Unfortunately, we do not know how to compute the above oracles in EXPH. The reason is that when we diagonalize N_i on input length n, the number of heavy entries $\mathcal{P}(r)$ we need to fix is $2^{(n'+4)K} \cdot 2^n = 2^{\mathrm{poly}(n')}$. Since n' might be exponentially large compared to n, this upper bound might be doubly exponential. Nevertheless, we show that under the assumption that

$$\mathsf{SAT} \in \mathsf{DTIME}[n \cdot \mathrm{polylog}(n)] \cap \mathsf{NC}, \tag{*}$$

the oracles $\mathcal{P}, \mathcal{Q}$ in Theorem 11 can be computed in PSPACE. It follows that

▶ **Corollary 45.** *Any* PSPACE*-relativizing proof of*

$$\mathsf{PSPACE} \not\subseteq \mathsf{i.o.BPP} \implies \mathsf{BPP} \subseteq \mathsf{heur\text{-}SUBEXP}$$

would also imply a breakthrough lower bound for SAT*, i.e., refuting Equation* (*).

▶ **Theorem 12.** *Suppose that* $\mathsf{SAT} \in \mathsf{DTIME}[n \cdot \mathrm{polylog}(n)] \cap \mathsf{NC}$*. Then there is an oracle* $\mathcal{O}$ *satisfying the conclusions of Theorem 11 that can be computed in polynomial space.*

We sketch some intuition before presenting the full proof of Theorem 12. The bottleneck of putting $\mathcal{O}$ into EXPH is that we do not have enough space to store all the heavy entries $\mathcal{P}(r)$ fixed by a probabilistic machine N_i, as there might be doubly-exponentially many of them. Therefore, whenever we simulate a machine N_i and it asks a query $\mathcal{P}(r)$, we have to compute from scratch whether $\mathcal{P}(r)$ was fixed by a previous machine. If we have an oracle Heavy that given r, decides whether $\mathcal{P}(r)$ was already fixed, then we can simulate N_i in exponential time with a constant number of alternations (using Theorem 24). However, there are $\mathrm{poly}(n)$ machines N_j simulated before N_i, and each time we invoke Heavy, we need to simulate these machines to see if any of them has already fixed $\mathcal{P}(r)$. It follows that the simulation of N_i actually requires $\mathrm{poly}(n)$ alternations and $2^{\mathrm{poly}(n)}$ time. Still, under a strong enough assumption such as (*), we can simulate these $\mathrm{poly}(n)$ alternations in $2^{\mathrm{poly}(n)}$ time.

Proof of Theorem 12. Consider the construction in Theorem 11, with the only difference that we do not calculate the precise probabilities; instead, we compute their approximations in PH. Since $\mathsf{SAT} \in \mathsf{NC}$, we have $\mathsf{EXP} = \mathsf{PSPACE}$ by padding, thus it suffices to construct the oracles in $2^{\mathrm{poly}(n)}$ time.

The most involved part of this proof is to compute the following function $\textsc{Heavy}(n, \langle \mathcal{P}, \mathcal{Q} \rangle, i, b, x_{i,b}, r)$, which indicates whether $\mathcal{P}(r)$ is a heavy query of the probabilistic machine N_i on input $x_{i,b}$. Here, the input of Heavy consists of the description of oracles $\mathcal{P}$ and $\mathcal{Q}$, an integer $i \leq n$, an input $x_{i,b} \in \{0,1\}^n$, and a query $r \in \{0,1\}^{(n_{i'})^4}$ (where $n_{i'}$ is the smallest element in the sequence $\{n_i\}$ such that $n_{i'} \geq n$). We may assume $|r| \leq 2^n$ as otherwise N_i could never query $\mathcal{P}(r)$. The oracle description $\langle \mathcal{P}, \mathcal{Q} \rangle$ will contain the following information:

- We record a table of all entries in $\mathcal{Q}$ that are fixed; there are at most $\sum_{m \le n} 2^m + 2^{6n} \le 2^{O(n)}$ such entries. For each entry, we also record a timestamp indicating when this entry is fixed.
- We record all entries of $\mathcal{P}$ up to the $n_{i'-1}$-th slice (but we do *not* include any information of the $n_{i'}$-th slice of $\mathcal{P}$); the description length of $\mathcal{P}$ is also at most $2^{O(n)}$.

Note that the input length of HEAVY is at most $2^{O(n)}$. Instead of requiring HEAVY to decide exactly whether $\mathcal{P}(r)$ is a heavy query, we only require a 2-approximation: if $\mathcal{P}(r)$ is queried by $N_i(x_{i,b})$ w.p. at least $2^{-(n'+4)K}$ then HEAVY returns 1, while if $\mathcal{P}(r)$ is queried w.p. at most $2^{-(n'+4)K}/2$ then HEAVY returns 0.

Let k be the number of machine-input pairs we have simulated before $N_i(x_{i,b})$ (inclusive). That is,

$$k := \sum_{m<n} (2m) + 2i + b + 1 \le O(n^2).$$

We will show that:

▷ Claim 46. If $\mathsf{SAT} \in \mathsf{DTIME}[n \cdot \mathrm{polylog}(n)]$, then we can compute HEAVY in deterministic $O(2^{k^2})$ time.

Proof. Consider simulating (a random computational branch of) $N_i(x_{i,b})$ while answering the oracle queries of N_i accordingly.[12] Note that we have not recorded the $n_{i'}$-th slice of $\mathcal{P}$ in our description $\langle \mathcal{P}, \mathcal{Q} \rangle$. As a consequence, whenever $N_i(x_{i,b})$ asks a query $\mathcal{P}(r_q)$ where $|r_q| = n_{i'}$, we need to recursively call HEAVY to decide whether this query was already fixed and which value it was fixed to. We enumerate every $n_{i'-1} < \tilde{n} \le n$, every machine N_j ($j \le \tilde{n}$), and every bit b'. Suppose that we feed the machine N_j with the input $x'_{j,b'} \in \{0,1\}^{n_{i'}}$.[13] Using the timestamps recorded in the table $\mathcal{Q}$, we can recover the state of the oracle $\mathcal{Q}$ before the simulation of $N_j(x'_{j,b'})$; we call this oracle $\mathcal{Q}'$. (The portion of oracle $\mathcal{P}$ up to the $n_{i'-1}$-th slice remains the same.) Then, we call $\textsc{Heavy}(\tilde{n}, \langle \mathcal{P}, \mathcal{Q}' \rangle, j, b', x'_{j,b'}, r_q)$ to see if this query is a heavy query fixed by N'_j. If it is, then we return $\mathcal{P}(r_q) := b'$; otherwise we search through the next machine. If the query $\mathcal{P}(r)$ was not fixed before and was asked in this simulation. then we return 1; otherwise we return 0.

The above argument implies that HEAVY can be computed recursively in the following sense. There is a machine $V_{\textsc{Heavy}}$ that gets $\mathsf{input} := (n, \langle \mathcal{P}, \mathcal{Q} \rangle, i, b, x_{i,b}, r)$ and some randomness $z \in \{0,1\}^{2^{O(n)}}$ (denoting a random computational branch of $N_i(x_{i,b})$), runs in $2^{O(n)}$ time with $\mathrm{poly}(n)$ invocations of the HEAVY oracle, and outputs 0 or 1. Let $\varepsilon := 2^{-(n'+4)/K}$, we want that $\textsc{Heavy}(\mathsf{input}) = 1$ if $\Pr_z[V_{\textsc{Heavy}}(\mathsf{input}; z) = 1] \ge \varepsilon$, and $\textsc{Heavy}(\mathsf{input}) = 0$ if $\Pr_z[V_{\textsc{Heavy}}(\mathsf{input}; z) = 1] \le \varepsilon/2$.

We can use induction to show that HEAVY can be computed in (deterministic) $O(2^{k^2})$ time. In particular, by the induction hypothesis, each call of the HEAVY oracle made by $V_{\textsc{Heavy}}$ on input can be computed in $O(2^{(k-1)^2})$ time. Therefore, $V_{\textsc{Heavy}}$ runs in at most $2^{(k-1)^2} \cdot \mathrm{poly}(k)$ time. By Theorem 24, there is a $\Sigma_5\mathsf{TIME}[2^{(k-1)^2} \cdot \mathrm{poly}(k)]$ time machine $\widetilde{\textsc{Heavy}}$ such that $\widetilde{\textsc{Heavy}}(\mathsf{input}) = 1$ if $\Pr_z[V_{\textsc{Heavy}}(\mathsf{input}; z) = 1] \ge \varepsilon$ and $\widetilde{\textsc{Heavy}}(\mathsf{input}) = 0$ if $\Pr_z[V_{\textsc{Heavy}}(\mathsf{input}; z) = 1] \le \varepsilon/2$. By our hypothesis that $\mathsf{SAT} \in \mathsf{DTIME}[n \cdot \mathrm{polylog}(n)]$, there is a deterministic machine HEAVY that runs in $2^{(k-1)^2} \cdot \mathrm{poly}(k) < 2^{k^2}$ time and decides the same language as $\widetilde{\textsc{Heavy}}$. ◁

[12] That is, whenever N_i asks a query $\mathcal{P}(r)$ or $\mathcal{Q}(x, r)$ that is fixed, we answer accordingly. Whenever N_i asks a query $\mathcal{P}(r)$ that is not fixed, if $|r| = (n')^4$ then we return b, otherwise we return 0. Whenever N_i asks a query $\mathcal{Q}(x, r)$ that is not fixed, we return 0.

[13] We assume that there is an easy and deterministic way of assigning the inputs $x_{j,b'}$ for each pair (j, b').

Now we show that given an integer N, it is possible to print the N-th slice of $\mathcal{P}$ and $\mathcal{Q}$ in deterministic $2^{\mathrm{poly}(N)}$ time. We simulate the stages n for $n = 1, 2, \cdots$, where during each stage we need to diagonalize against probabilistic machines $N_1, N_2, \cdots$. If $n = n_i$ for some $i \in \mathbb{N}$, then we also need to diagonalize against M_i.

- For each N_i, we can list the set of its heavy queries of the form $\mathcal{Q}(x, r)$ in deterministic $2^{\mathrm{poly}(n)}$ time with a $\mathsf{PH}^{\textsc{Heavy}}$ oracle. Note that Heavy can be decided in time quasi-polynomial in its input length; also recall we assumed that $\mathsf{SAT} \in \mathsf{P}$. Thus we can enumerate the heavy queries of the form $\mathcal{Q}(x, r)$ in deterministic $2^{\mathrm{poly}(n)}$ time (without additional oracles). We fix all these queries. (Note that we do not fix the heavy queries of the form $\mathcal{P}(r)$; instead, we use the oracle Heavy to decide whether a query $\mathcal{P}(r)$ is fixed.) Then we use the PH oracle to estimate the probability that $N_i(x_{i,b})$ outputs 1. If the probability is small then we pick some r such that $\mathcal{Q}(x_{i,b}, r)$ is unfixed, and fix this entry to 1; otherwise we do nothing.
- Before we diagonalize against M_i, we spend $2^{\mathrm{poly}(n)}$ time to retrieve the list of fixed entries in the n^4-th slice of $\mathcal{P}$ from the oracle Heavy. Then we simulate M_i, and for every unfixed query, we fix it to be 0. Let $b \in \{0, 1\}$ be the output of M_i, then we set every unfixed entry in the n^4-th slice of $\mathcal{P}$ to be $1 - b$.

It is easy to see that the above process runs in $2^{\mathrm{poly}(N)}$ time. ◀

▶ Remark 47. A beautiful line of work [24, 53, 25, 22, 76, 77, 15, 58] investigated time-space trade-off lower bounds for SAT. Lower bounds proved in these works come in two flavors: "SAT cannot be solved by a machine with certain time and space bounds *simultaneously*", or "SAT either cannot be solved in some time bound, or cannot be solved in some space bound (even by two different machines)". The state-of-the-art lower bounds of the first flavor is that SAT cannot be solved in n^c time and $n^{o(1)}$ space *simultaneously*, for every $c < 2\cos(\pi/7) \approx 1.801$ [77]; note that such lower bounds do not contradict (*). The state-of-the-art lower bounds of the second flavor is that SAT either requires more than $n \cdot \mathrm{polylog}(n)$ time or requires $\log^{2-o(1)} n$ depth [58]. To the best of our knowledge, it is still an open question to disprove (*).

5 Barriers for Explicit Constructions

5.1 PSPACE-Relativizing Pseudodeterministic Constructions

In this section, we show that the previous results on pseudodeterministic constructions are PSPACE-relativizing. A property Q is *dense* if for every $n \in \mathbb{N}$, $|Q \cap \{0,1\}^n| \geq 2^n/\mathrm{poly}(n)$. For every dense property Q computable in polynomial time, Oliveira and Santhanam [62] presented a pseudodeterministic algorithm that on input 1^n, outputs a canonical element in Q_n with high probability. Their pseudodeterministic algorithm runs in subexponential time, is zero-error (i.e., the algorithm never outputs any element besides $\perp$ ("failure") and the canonical one), and is correct on infinitely many input lengths n. We verify that their argument holds relative to every oracle $O \in \mathsf{PSPACE}$.

▶ **Proposition 48** (Formal Version of Proposition 13)**.** *Let $\mathcal{O} \in \mathsf{PSPACE}$. Then for every $\varepsilon > 0$ and every dense property $Q \in \mathsf{P}^{\mathcal{O}}$, there exist a zero-error pseudodeterministic $\mathcal{O}$-oracle algorithm A with running time 2^{n^ε} and an infinite sequence $\{x_{n_i}\}_{i \in \mathbb{N}}$ such that $x_{n_i} \in Q \cap \{0,1\}^{n_i}$ for each $i \in \mathbb{N}$ and that*

$$\Pr_A\left[A^{\mathcal{O}}(1^{n_i}) = x_{n_i}\right] \geq 3/4.$$

Proof Sketch. We follow the analysis from [62].

Let $c \geq 1$ and $Q \in \mathsf{DTIME}^{\mathcal{O}}[n^c]$ be a dense property. It suffices to show an HSG $\{H_n\}_n$ of seed length n^ε such that H_n can be computed pseudodeterministically with zero error in time 2^{n^ε} with oracle access to $\mathcal{O}$, and that for infinitely many n, H_n hits Q on input length n. This is because we can then enumerate all $z \in \{0,1\}^{n^\varepsilon}$ and find the first z such that $Q(H_n(z)) = 1$. Such a z exists since Q is a dense property and H_n hits Q.

First of all, using the "easy witness" method [45], one can show that there exists a family of sets $\{H_n^{\mathsf{easy}} \subseteq \{0,1\}^n\}_n$, such that each H_n^{easy} is computable deterministically in time 2^{n^ε} with oracle access to $\mathcal{O}$, and that if Q avoids $\{H_n^{\mathsf{easy}}\}_n$ on all but finitely many input length n, then $\mathsf{BPP}^{\mathcal{O}} = \mathsf{ZPP}^{\mathcal{O}}$. Roughly speaking, H_n^{easy} contains the truth tables of all $\mathcal{O}$-circuits $C\colon \{0,1\}^{\log n} \to \{0,1\}$ of size at most $n^{\varepsilon/10}$. If Q avoids H_n^{easy}, then we can obtain, with high probability and without error, an n-bit truth table that has $\mathcal{O}$-circuit complexity at least $n^{\varepsilon/10}$, by randomly picking an n-bit string that is accepted by Q. Such hard truth tables can then be used to derandomize $\mathsf{BPP}^{\mathcal{O}}$. (See [62, Proof of Lemma 2] for details.)

If $\mathsf{BPP}^{\mathcal{O}} \neq \mathsf{ZPP}^{\mathcal{O}}$, $\{H_n^{\mathsf{easy}}\}_n$ will give a valid HSG. Now assume $\mathsf{BPP}^{\mathcal{O}} = \mathsf{ZPP}^{\mathcal{O}}$. It suffices to construct an infinitely often HSG that can be computed pseudodeterministically with oracle access to $\mathcal{O}$, since it can be made zero-error using the assumption $\mathsf{BPP}^{\mathcal{O}} = \mathsf{ZPP}^{\mathcal{O}}$.

We consider two cases. Suppose $\mathsf{PSPACE} \subseteq \mathsf{BPP}^{\mathcal{O}}$. Then the existence of a valid HSG follows easily from Lemma 26. Now suppose $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$. We can also use the following hardness-to-randomness construction to obtain a valid HSG.

▶ **Theorem 49** ([42, 70, 50]). *Let $\mathcal{O}$ be any oracle. If $\mathsf{PSPACE} \not\subseteq \mathsf{BPP}^{\mathcal{O}}$, then for every $b, c \geq 1$, there is a sequence $\{G_\ell\}_{\ell \geq 1}$, where $G_\ell\colon \{0,1\}^\ell \to \{0,1\}^{\ell^b}$ is computable in time $2^{O(\ell)}$, such that for every language $L \in \mathsf{DTIME}^{\mathcal{O}}[n^c]$, there are infinitely many ℓ such that*

$$\left| \Pr_{x \sim \{0,1\}^{\ell^b}}[L(x) = 1] - \Pr_{z \sim \{0,1\}^{\ell}}[L(G_\ell(z)) = 1] \right| \leq \frac{1}{10}.$$

This completes the proof of Proposition 48. ◀

5.2 Bounded-Relativization Barriers for Explicit Constructions

5.2.1 Barriers for Almost-Everywhere Pseudodeterministic Constructions

We show that EXPH-relativizing techniques cannot prove *almost-everywhere* pseudodeterministic constructions, even if the construction algorithms are allowed $2^{o(n)}$ time. The underlying oracle uses the query complexity lower bounds proved by Goldwasser, Impagliazzo, Pitassi, and Santhanam [31] *in a black-box fashion.* (We thank Rahul Santhanam for pointing out that the query complexity lower bounds imply an EXPH-computable oracle without almost-everywhere pseudodeterministic constructions in a black-box fashion.)

▶ **Definition 50.** *A* pseudodeterministic *decision tree for a search problem S is a distribution $\mathcal{T}$ over decision trees with the following property: For every input x, there is a* canonical *value o such that with probability at least $2/3$, $\mathcal{T}(x) = o$. Let $\mathsf{psP}^{\mathsf{dt}}(S)$ denote the minimum depth of any pseudodeterministic decision tree for S.*

Consider the following search problem denoted as FIND1. The input is a string $x \in \{0,1\}^N$ where it is guaranteed that there are at least $N/2$ bits in x that are equal to 1. The problem is to find some $i \in [N]$ such that $x_i = 1$. The following lower bound on the pseudodeterministic query complexity of FIND1 was proved in [31]:

▶ **Theorem 51** ([31]). $\mathsf{psP}^{\mathsf{dt}}(\textsc{Find1}) = \Omega(\sqrt{N})$.

We use this result as a black box and construct an oracle $\mathcal{O} \in \mathsf{EXPH}$ without almost-everywhere pseudodeterministic constructions running in $2^{o(n)}$ time:

▶ **Theorem 52.** *There is an oracle $\mathcal{O} \in \mathsf{EXPH}$ and a dense property $Q \in \mathsf{P}^{\mathcal{O}}$ such that the following holds. For every randomized (and purportedly pseudodeterministic) algorithm $\mathcal{A}$ that runs in $2^{o(n)}$ time with oracle access to $\mathcal{O}$ and every infinite sequence of outputs $\{x_n\}_{n\in\mathbb{N}}$ where each $x_n \in Q \cap \{0,1\}^n$, there are infinitely many input lengths $n \in \mathbb{N}$ such that*

$$\Pr[\mathcal{A}(1^n) = x_n] \leq 3/4.$$

Proof. Let $M_1, M_2, \cdots$ be an enumeration of probabilistic machines running in $2^{0.1n}$ time. Let n_1 be a large enough constant and define the sequence $\{n_i\}$ where $n_i = 2^{n_{i-1}}$ for each $i \geq 2$. We let $Q = \mathcal{O}$ itself be the dense property without pseudodeterministic construction algorithms. In particular, we want that for every $i \in \mathbb{N}$, $M_i(1^{n_i})$ fails to generate a canonical string $x \in \mathcal{O} \cap \{0,1\}^{n_i}$. Naturally, our oracle construction proceeds in stages, where for each $i \in \mathbb{N}$, the n_i-th slice of $\mathcal{O}$ is constructed by carefully diagonalizing against M_i; on the other hand, if $n \in \mathbb{N}$ is not in the sequence $\{n_i\}$, then we simply let $\mathcal{O}$ accept every string of length n.

Now we show how to construct the n_i-th slice of $\mathcal{O}$ in the i-th stage. The idea is simple: construct the (purportedly pseudodeterministic) decision tree $\mathcal{T}$ corresponding to M_i, find an exponential-length input tt of FIND1 on which $\mathcal{T}$ fails, and let the truth table of the n_i-th slice of $\mathcal{O}$ be tt. More precisely:

- **Converting M_i into a distribution of decision trees.** We define the distribution of decision trees $\mathcal{T}$. The input to FIND1 has length 2^{n_i} and is considered as the truth table of the n_i-th slice of $\mathcal{O}$. Note that since $2^{0.1n_i} \ll n_{i+1}$, for every $n' \leq 2^{0.1n_i}$ such that $n' \neq n_i$, the n'-th slice of $\mathcal{O}$ are fixed. To sample a (deterministic) decision tree T from $\mathcal{T}$, sample a sequence of random coins fed to M_i and simulate M_i on the oracle $\mathcal{O}$. Whenever M_i makes a query $\mathcal{O}(x)$, if $|x| = n_i$, then the decision tree asks the x-th bit of our input; otherwise we return the already-fixed value of $\mathcal{O}(x)$.
- **Invoking the lower bound.** Note that $\mathcal{T}$ only makes $2^{0.1n_i} < o(\sqrt{2^{n_i}})$ queries. By Theorem 51, there exists an input $tt \in \{0,1\}^{2^{n_i}}$ such that $\mathcal{T}$ fails to solve FIND1 on input tt pseudodeterministically. That is, for every valid output x such that $tt_x = 1$, $\Pr[\mathcal{T}(tt) = x] < 2/3$. Let the truth table of the n_i-th slice of $\mathcal{O}$ to be such a tt.
  It is easy to see that for every probabilistic machine $\mathcal{A}$ that runs in $2^{o(n)}$ time with oracle access to $\mathcal{O}$, and every infinite sequence of outputs $\{x_n\}_{n\in\mathbb{N}}$ where each $x_n \in \mathcal{O} \cap \{0,1\}^n$, for every i such that $\mathcal{A} = M_i$, the probability that $\mathcal{A}(1^{n_i})$ outputs x_{n_i} is at most $3/4$.
- **Complexity of $\mathcal{O}$.** Let $\mathcal{O}_{\leq n_{i-1}}$ denote the description of the oracle $\mathcal{O}$ up to the n_{i-1}-th slice (note that $\mathcal{O}_{\leq n_{i-1}}$ can be described in $\mathrm{poly}(2^{n_{i-1}}) \leq \mathrm{poly}(n_i)$ bits), and $x \in \{0,1\}^{n_i}$. Note that $\mathcal{O}_{\leq n_{i-1}}$ and tt together defines the oracle $\mathcal{O}$ up to input length $n_{i+1} - 1$, so the behavior of $M_i^{\mathcal{O}}$ is completely determined by $M_i, \mathcal{O}_{\leq n_{i-1}}$, and tt. Let $\mathsf{ProbEst}(\mathcal{O}_{\leq n_{i-1}}, tt, M_i, x)$ be an oracle such that

$$\Pr[M_i^{\mathcal{O}}(1^{n_i}) = x] > 3/4 \implies \mathsf{ProbEst}(\mathcal{O}_{\leq n_{i-1}}, tt, M_i, x) = 1,$$
$$\Pr[M_i^{\mathcal{O}}(1^{n_i}) = x] < 2/3 \implies \mathsf{ProbEst}(\mathcal{O}_{\leq n_{i-1}}, tt, M_i, x) = 0.$$

  By Theorem 23, the oracle $\mathsf{ProbEst}$ can be implemented in $\mathsf{DTIME}[2^{O(n_i)}]^{\mathsf{PH}}$. Consider the following algorithm with oracle access to $\mathsf{ProbEst}$ that prints a truth table tt. The algorithm maintains the prefix of a truth table which is initially the empty string, and extends this prefix bit by bit. Suppose that we have a prefix of length ℓ, denoted as $tt' \in \{0,1\}^{\ell}$. To fix the $(\ell+1)$-st bit of tt', we check if there exists a truth table tt

such that (1) for every $x \in \{0,1\}^{n_i}$, $\mathsf{ProbEst}(\mathcal{O}_{\leq n_{i-1}}, tt, M_i, x) = 0$, and (2) $tt' \circ 0$ (tt' concatenated with a bit 0) is a prefix of tt. If there is such a tt, then we append 0 to the end of tt'; otherwise we append 1 to the end of tt'.
It is clear that the algorithm always outputs a truth table tt on which M_i fails. Since $\mathsf{ProbEst}$ runs in $\mathsf{DTIME}[2^{O(n_i)}]^{\mathsf{PH}}$, the whole algorithm also runs in $\mathsf{DTIME}[2^{O(n_i)}]^{\mathsf{PH}}$. Using this algorithm, it is easy to construct the oracle $\mathcal{O}$ in EXPH. ◀

▶ **Corollary 53.** *If there is a* PSPACE*-relativizing proof that for every dense property Q computable in polynomial time, there is a pseudodeterministic construction for Q running in $2^{o(n)}$ time that is correct on* almost every input length*, then* $\mathsf{L} \neq \mathsf{NP}$*.*

5.2.2 Barriers for Deterministic Constructions and Lower Bounds for MKtP

In this section, we construct an oracle in EXP relative to which there is no deterministic construction in $2^n/n^{\omega(1)}$ time (even infinitely often), showing that any non-trivial deterministic constructions using PSPACE-relativizing techniques would separate PSPACE from EXP.

As an interesting corollary, it is easy to approximate the Kt complexity to an $(1+\varepsilon)$ factor in this oracle world in deterministic $n^{O(\log n)}$ time. Therefore, although it is EXP-hard to approximate the Kt complexity under $(\mathsf{P}/_{\mathrm{poly}})$-truth-table reductions and NP-Turing reductions [5], any PSPACE-relativizing proof that the Kt complexity requires deterministic $n^{\omega(\log n)}$ time to approximate would separate PSPACE from EXP.

▶ **Definition 54.** *For a constant $\varepsilon > 0$,* $\mathsf{Gap}_\varepsilon\mathsf{MKtP}$ *is defined to be the promise problem $(\mathcal{YES}_n, \mathcal{NO}_n)_{n\in\mathbb{N}}$, where*

$$\begin{aligned} \mathcal{YES}_n &:= \{(x,s) \in \{0,1\}^n \times \mathbb{N} : \mathrm{Kt}(x) \leq s\}, \\ \mathcal{NO}_n &:= \{(x,s) \in \{0,1\}^n \times \mathbb{N} : \mathrm{Kt}(x) > (1+\varepsilon)\cdot s\}. \end{aligned}$$

Our proof relies heavily on the equivalence between non-trivial derandomization and the hardness of $\mathsf{Gap}_\varepsilon\mathsf{MKtP}$ [34]. It is not hard to construct an oracle under which non-trivial derandomization is impossible, which means that a dense subset of the complement of MKtP can be accepted by an efficient algorithm A. Then, we use the worst-case to average-case reduction of [33] to transform A into a worst-case approximation algorithm for $\mathsf{Gap}_\varepsilon\mathsf{MKtP}$.

▶ **Theorem 55.** *There exists an oracle $\mathcal{O} \in \mathsf{EXP}$ such that*

1. *there is a dense property $Q \in \mathsf{P}^{\mathcal{O}}$ such that every deterministic algorithm that runs in time $2^n/n^{\omega(n)}$ fails to find a string in $Q \cap \{0,1\}^n$ on almost every input length n, and*
2. $\mathsf{Gap}_\varepsilon\mathsf{MKtP}^{\mathcal{O}} \in \mathsf{DTIME}^{\mathcal{O}}[n^{O(\log n)}]$ *for every constant $\varepsilon > 0$.*

To show the worst-case to average-case reduction, we use the following pseudorandom generator construction.

▶ **Lemma 56** (cf. [33])**.** *For any $d, m \leq 2n, \varepsilon > 0$, there exists a "pseudorandom generator construction"*

$$G\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$$

such that for any distinguisher $D : \{0,1\}^m \to \{0,1\}$, if

$$\Pr_{w\sim\{0,1\}^m}[D(w)=1] - \Pr_{z\sim\{0,1\}^d}[D(G(x,z))=1] \geq \frac{1}{2},$$

then

$$\mathrm{K}^{\mathrm{poly}(n),D}(x) \leq \exp(\ell^2/d) \cdot m + d + O(\log n),$$

where $\ell = O(\log n)$. *Moreover,* $G(x,z)$ *can be computed in time* $\mathrm{poly}(n)$.

Proof of Theorem 55. We construct an oracle $\mathcal{O} \in \mathsf{EXP}$ under which a dense subset of MKtP is in P. We start with $n := 1$, $\mathcal{O} := \emptyset$, and $F := \emptyset$, where F is a set of "frozen" strings. The construction of $\mathcal{O}$ in Stage n is as follows. Let $\mathcal{O}_n$ denote the state of the oracle $\mathcal{O}$ at the beginning of stage n. Let E_n be the set of all strings $x \in \{0,1\}^n$ such that $\mathrm{Kt}^{\mathcal{O}_n}(x) \leq n - c\log n$, where c is a sufficiently large constant (e.g., $c := 3$). Let F_n be the set of all strings $q \in \{0,1\}^*$ such that there exist $k \in [n - c\log n]$ and a description $d \in \{0,1\}^k$ of a Turing machine such that the universal Turing machine $U^{\mathcal{O}_n}$ on input d makes the query q in time $2^{n-k-c\log n}$. Then, we update $F := F \cup F_n$ and $\mathcal{O}_{n+1} := \mathcal{O}_n \cup (\{0,1\}^n \setminus (E_n \cup F))$ and move on to the next stage $n+1$. This completes the description of the oracle $\mathcal{O} := \bigcup_{n\in\mathbb{N}} \mathcal{O}_n$. It is easy to observe that $\mathcal{O} \in \mathsf{EXP}$.

The oracle $\mathcal{O}$ is a dense subset of $\left\{x : \mathrm{Kt}^{\mathcal{O}}(x) > n - c\log n\right\}$ in the following sense:

1. $x \notin \mathcal{O}$ for every string $x \in \{0,1\}^n$ such that $\mathrm{Kt}^{\mathcal{O}}(x) \leq n - c\log n$.
2. $x \in \mathcal{O}$ for at least half of the strings $x \in \{0,1\}^n$.

To see the first property, it suffices to show that if $x \in \{0,1\}^n$ satisfies $\mathrm{Kt}^{\mathcal{O}}(x) \leq n - c\log n$, then $\mathrm{Kt}^{\mathcal{O}_n}(x) \leq n - c\log n$ (which implies $x \in E_n$ and thus $x \notin \mathcal{O}$). Assuming that there exists a string d such that $U^{\mathcal{O}}(d)$ outputs x in time $2^{n-c\log n-|d|}$, we claim that $U^{\mathcal{O}_n}(d)$ also outputs x. If not, there exists a query in $\mathcal{O} \setminus \mathcal{O}_n$ made by $U^{\mathcal{O}}(d)$. Let q be the first query in $\mathcal{O} \setminus \mathcal{O}_n$. The same query is also queried during the computation of $U^{\mathcal{O}_n}(d)$. Thus, $q \in F_n$, which implies $q \notin \mathcal{O}$, which is a contradiction.

To see the second property, we bound the size of $E_n \cup F$ at Stage n. The size of E_n is at most $\sum_{k\in[n-c\log n]} 2^k \leq 2^n/n^{c-1}$. The number $|F_n|$ of frozen strings in Stage n is at most

$$\sum_{k\in[n-c\log n]} 2^k \cdot 2^{n-k-c\log n} \leq 2^n/n^{c-1}.$$

Thus, the number $|F|$ of frozen strings until Stage n is at most $\sum_{n'=1}^{n} |F_{n'}| \leq 2^n/n^{c-2}$. Overall, we obtain

$$|\mathcal{O} \cap \{0,1\}^n| \geq 2^n - |E_n| - |F| \geq \frac{1}{2} \cdot 2^n.$$

Let $Q := \mathcal{O}$. We prove that $Q \in \mathsf{P}^{\mathcal{O}}$ is a dense property such that every $2^n/n^{\omega(1)}$-time deterministic algorithm A fails to find a string in $Q \cap \{0,1\}^n$ on input 1^n. By the definition of Kt, we have $\mathrm{Kt}^{\mathcal{O}}(A(1^n)) \leq O(\log n) + \log(2^n/n^{\omega(1)}) \leq n - \omega(\log n)$. Thus, the output $A(1^n)$ of A is not in $\mathcal{O}$.

We now use the worst-case to average-case reduction of [34] to obtain an approximation algorithm $A^{\mathcal{O}}$ for $\mathsf{Gap}_\varepsilon\mathsf{MKtP}$. Let $x \in \{0,1\}^n$ and $s \in \mathbb{N}$ be an instance of $\mathsf{Gap}_\varepsilon\mathsf{MKtP}$. Let $d = k \cdot O(\log^2 n)$, where $k = k(\varepsilon)$ is a sufficiently large constant that will be chosen depending on $\varepsilon > 0$. We may assume without loss of generality that $O(k^2 \cdot \log^2 n) \leq s \leq n + O(\log n)$ because whether $\mathrm{Kt}(x) \leq O(k^2 \cdot \log^2 n)$ or not can be decided in time $n^{O(k^2 \cdot \log n)}$ by an exhaustive search. Let m be a parameter chosen later. Let G be the pseudorandom generator construction of Lemma 56. The algorithm $A^{\mathcal{O}}$ accepts (x,s) if and only if $G(x,z) \notin \mathcal{O}$ for every $z \in \{0,1\}^d$. The running time of $A^{\mathcal{O}}$ is $2^d\mathrm{poly}(n) = n^{O(\log n)}$.

We prove the correctness of $A^{\mathcal{O}}$. Assume that $\mathrm{Kt}^{\mathcal{O}}(x) \leq s$. Then, for every $z \in \{0,1\}^d$, we have

$$\mathrm{Kt}^{\mathcal{O}}(G(x,z)) \leq \mathrm{Kt}^{\mathcal{O}}(x) + O(d + \log n) \leq s + O(d + \log n) \leq m - c\log m,$$

where the last inequality holds by choosing a sufficiently large $m = s + O(d + \log n)$. It follows that $G(x,z) \notin \mathcal{O}$ and that $A^{\mathcal{O}}$ accepts (x,s). Conversely, assume that $A^{\mathcal{O}}$ accepts (x,s), which means that $\Pr_{z\sim\{0,1\}^d}[\mathcal{O}(G(x,z)) = 1] = 0$. We claim that $\mathrm{Kt}^{\mathcal{O}}(x) \leq (1+\varepsilon)\cdot s$. Since $\mathcal{O}$ is dense, we have

$$\Pr_{w\sim\{0,1\}^m}[\mathcal{O}(w) = 1] - \Pr_{z\sim\{0,1\}^d}[\mathcal{O}(G(x,z)) = 1] \geq \frac{1}{2}.$$

By Lemma 56, we obtain

$$\mathsf{K}^{\mathrm{poly}(n),\mathcal{O}}(x) \leq \exp(O(\log^2 n)/d)\cdot m + d + O(\log n) \leq \exp(1/k)\cdot m + O(k\cdot\log^2 n).$$

In particular,

$$\mathrm{Kt}^{\mathcal{O}}(x) \leq \exp(1/k)\cdot m + O(k\cdot\log^2 n) \leq (1+O(1/k))\cdot m + O(k\cdot\log^2 n) \leq (1+\varepsilon)\cdot m,$$

where we choose a sufficiently large $k := O(1/\varepsilon)$ and use that $m \geq O(k^2\cdot\log^2 n)$ in the last inequality. ◀

6 Barriers for Circuit Lower Bounds for Merlin–Arthur Classes

Buhrman, Fortnow, and Thierauf [13] showed that $\mathsf{MA\text{-}EXP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ and Santhanam [65] proved that $\mathsf{MA}/_1 \not\subseteq \mathsf{SIZE}[n^k]$ for every constant k. Their techniques rely heavily on win-win analysis and thus only yield circuit lower bounds that hold infinitely often.

This section presents an EXPH-relativizing barrier for proving almost-everywhere versions of these lower bounds. In Section 6.1, we show that Santhanam's circuit lower bound is PSPACE-relativizing. In Section 6.2, we construct an EXPH oracle under which the almost-everywhere lower bound fails.

6.1 PSPACE-Relativizing Circuit Lower Bounds for Merlin–Arthur Classes

▶ **Theorem 57.** *Let $\mathcal{O} \in \mathsf{PSPACE}$. For every $k \in \mathbb{N}$, $\mathsf{MA}^{\mathcal{O}}/_1 \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[n^k]$.*

We need the notion of an instance checker.

▶ **Definition 58** (Instance-Checkable Languages)**.** *A language L is said to be same-length instance-checkable if there is a probabilistic polynomial-time oracle machine $I^{(-)}$ with output in $\{0,1,\bot\}$ such that for any input x:*

1. *$I^{(-)}$ only makes oracle queries of length $|x|$.*
2. *$I^L(x) = L(x)$ with probability 1.*
3. *$I^A(x) \in \{L(x),\bot\}$ with probability at least $2/3$ for any oracle A.*

▶ **Lemma 59** ([70, 26])**.** *There is a PSPACE-complete language L_{hard} that is same-length instance-checkable.*

We now show Theorem 57. The proof follows closely to that of [65], which employs a win-win argument. Let L_{hard} be the language from Lemma 59. We will consider two cases: $L_{\mathsf{hard}} \in \mathsf{SIZE}^{\mathcal{O}}[\mathrm{poly}]$ and $L_{\mathsf{hard}} \notin \mathsf{SIZE}^{\mathcal{O}}[\mathrm{poly}]$.

▶ **Lemma 60.** *Let $\mathcal{O} \in \mathsf{PSPACE}$. If $L_{\mathsf{hard}} \in \mathsf{SIZE}^{\mathcal{O}}[\mathrm{poly}]$, then for every $k \in \mathbb{N}$, $\mathsf{MA}^{\mathcal{O}} \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[n^k]$.*

Proof. We first note that $L_{\mathsf{hard}} \in \mathsf{SIZE}^{\mathcal{O}}[\mathrm{poly}]$ implies $L_{\mathsf{hard}} \in \mathsf{MA}^{\mathcal{O}}$. More specifically, the $\mathsf{MA}^{\mathcal{O}}$ protocol for L_{hard} on input $x \in \{0,1\}^n$ is as follows. Merlin sends to Arthur the polynomial-size $\mathcal{O}$-oracle circuit C_n that computes L_{hard} on inputs of length n. Then Arthur, which has access to the oracle $\mathcal{O}$, runs the (same-length) instance checker I for L_{hard} on x while answering its queries using C_n and outputs 1 if and only if $I^{C_n}(x) = 1$. The correctness follows from the property of the instance checker. Since L_{hard} is PSPACE-complete, we have $\mathsf{PSPACE} \subseteq \mathsf{MA}^{\mathcal{O}}$.

Following the (folklore) diagonalization argument showing that PSPACE does not have fixed-polynomial-size circuits, one can show that for every $k \geq 1$, there is a language $L \in \mathsf{PSPACE}$ that does not have $\mathcal{O}$-oracle circuits of size n^k. This argument works because $\mathcal{O} \in \mathsf{PSPACE}$ and we can simulate any size-(n^k) $\mathcal{O}$-oracle in polynomial space. Then by the previous paragraph, we have $L \in \mathsf{PSPACE} \subseteq \mathsf{MA}^{\mathcal{O}}$. ◀

▶ **Lemma 61.** *Let $\mathcal{O}$ be any oracle. If $L_{\mathsf{hard}} \notin \mathsf{SIZE}^{\mathcal{O}}[\mathrm{poly}]$, then for every $k \in \mathbb{N}$, $\mathsf{MA}^{\mathcal{O}}/_1 \not\subseteq \mathsf{SIZE}^{\mathcal{O}}[n^k]$.*

The proof of Lemma 61 is essentially the same as the original proof in [65]. We present the detail here for completeness.

Proof of Lemma 61. We define the following padded version of L_{hard}, called L_k, as follows.

> $x \in L_k$ iff $x = yz$, where $y \in L_{\mathsf{hard}}$, $|z| > |y|$, $|z| = 2^\ell$ for some integer ℓ and $(|y|+|z|)^{k+1} \leq s(|x|) < (|y|+2|z|)^{k+1}$, where $s(m)$ is the minimum size of an $\mathcal{O}$-oracle circuit that computes L_{hard} on inputs of length m.

We will show that L_k is in $\mathsf{MA}^{\mathcal{O}}/_1$, but does not have $\mathcal{O}$-oracle circuits of size n^k.

For the upper bound, we first specify the sequence of advice bits. We say that input length $n \in \mathbb{N}$ is *good* for L_k if $n = m + 2^\ell$ for non-negative integers m and ℓ, $n > 2m$ and $(m+2^\ell)^{k+1} \leq s(m) \leq (m+2\cdot 2^\ell)^{k+1}$, where $s(m)$ is the minimum size of an $\mathcal{O}$-oracle circuit that computes L_{hard} on inputs of length m. Note that if n is good for L_k, $m = m(n)$ and $\ell = \ell(n)$ are well-defined, since in this case, we can obtain ℓ by looking at the most significant bit of the binary representation of n. For input length n, we let the corresponding advice bit $b_n = 1$ iff n is good.

Consider the following procedure for deciding L_k. On input $x \in \{0,1\}^n$, Arthur first checks if the advice bit b_n is 1. If not, reject immediately, Otherwise, we have $n = m + 2^\ell$ for some $m, \ell \in \mathbb{N}$ and Arthur can obtain $yz := x$, where $|y| = m, |z| = 2^\ell$. Note that in this case, $x \in L_k$ if and only if $y \in L_{\mathsf{hard}}$. Then Merlin sends to Arthur the minimum $\mathcal{O}$-oracle circuit C_m that computes L_{hard} on inputs of length m. Note that the size of this circuit is $s(m) < (m+2\cdot 2^\ell)^{k+1} \leq n^{O(1)}$ since n is good. Then Arthur, which has oracle access to $\mathcal{O}$, runs the (same-length) instance checker I for L_{hard} on y while answering its queries using C_m, and he outputs 1 if and only if $I^{C_m}(y) = 1$. The correctness follows from the property of the instance checker.

For the lower bounds, suppose for the sake of contradiction that $L_k \in \mathsf{SIZE}^{\mathcal{O}}[n^k]$. Let $\{D_n\}_n$ be a circuit family that computes L_k, where each D_n is an $\mathcal{O}$-oracle circuit of size n^k.

For each $m \in \mathbb{N}$, let $s(m)$ be the minimum size of an $\mathcal{O}$-oracle circuit that computes L_{hard} on inputs of length m. By assumption, $L_{\mathsf{hard}} \notin \mathsf{SIZE}^{\mathcal{O}}[\mathrm{poly}]$, so there is an infinite set $\mathcal{I} \subseteq \mathbb{N}$ such that for each $m \in N$,

$$s(m) > (m+1)^{k+1}.$$

Consider the following sequence of circuits $\{C_m\}_{m\in\mathcal{I}}$ that computes L_{hard} on the input lengths in $\mathcal{I}$. For each $m \in \mathcal{I}$, consider the the unique integer ℓ such that

$$(m + 2^\ell)^{k+1} \leq s(m) < (m + 2 \cdot 2^\ell)^{k+1}. \tag{2}$$

Such an ℓ exists since $(m+1)^{k+1} < s(m) \leq 2^m$ for $m \in \mathcal{I}$. Then on input $x \in \{0,1\}^m$, C_m simulates D_{m+2^ℓ} on $x1^{2^\ell}$. Since $m + 2^\ell$ is a good length and D_{m+2^ℓ} computes L_k correctly on inputs of length $m + 2^\ell$, $C_{m+2^\ell}\left(x1^{2^\ell}\right) = 1$ if and only if $x \in L_k$. Note that the size of C_m is at most the size of D_{m+2^ℓ}, which is at most $(m + 2^\ell)^{k+1} < s(m)$. This contradicts Equation (2). ◀

Proof of Theorem 57. Theorem 57 follows directly from Lemma 60 and Lemma 61. ◀

6.2 Bounded-Relativization Barriers for Circuit Lower Bounds

We present an EXPH-relativization barrier for proving an almost-everywhere version of Santhanam's lower bound. The oracle construction is based on [13]. Although the construction in [13] does not seem to be in EXPH, by modifying the construction using approximation counting in PH, we obtain an EXPH-computable oracle under which $\mathsf{MA}_{/1}$ is computable by linear-sized circuits infinitely often.

▶ **Lemma 62.** *There exists an oracle $\mathcal{O} \in \mathsf{EXPH}$ such that*

$$\mathsf{MATIME}^{\mathcal{O}}[2^n]/_1 \subseteq \mathsf{i.o.SIZE}^{\mathcal{O}}[O(n)].$$

Proof. We prove $\text{pr-}\mathsf{MATIME}^{\mathcal{O}}[2^n] \subseteq \mathsf{i.o.SIZE}^{\mathcal{O}}[O(n)]$ for some oracle $\mathcal{O}$. We enumerate all the 2^n-time randomized machine $M_1, M_2, \cdots$, where each M_i takes an input x of length n and a certificate $y \in \{0,1\}^{2^n}$ and runs in time 2^n. Note that any MA-type algorithm that runs in time 2^n can be simulated by M_i for some i.

Let c be some universal constant ($c := 5$ suffices). Let $R_n := \{0,1\}^{cn}$. The input to $\mathcal{O}$ is of the form (r, i, x), where $r \in R_n$, $i \in [n]$, and $x \in \{0,1\}^n$ for some $n \in \mathbb{N}$. For any $r \in \{0,1\}^{cn}$, let S_r be the set of (r, i, x) such that $i \in [n]$ and $x \in \{0,1\}^n$.

We will construct an oracle $\mathcal{O}$ and a family of "advice strings" $r_n^* \in R_n$ for infinitely many n such that

1. if $\Pr_{M_i}[M_i^{\mathcal{O}}(x, y) = 1] \geq \frac{3}{4}$ for some y, then $(r_n^*, i, x) \in \mathcal{O}$, and
2. if $\Pr_{M_i}[M_i^{\mathcal{O}}(x, y) = 1] \leq \frac{1}{4}$ for any y, then $(r_n^*, i, x) \notin \mathcal{O}$.

Assuming this, it is easy to construct an $\mathcal{O}$-oracle linear-size circuit that simulates M_i as follows. The circuit takes r_n^* as hard-wired input and accepts an input $x \in \{0,1\}^n$ if and only if $(r_n, i, x) \in \mathcal{O}$.

Here is the construction of $\mathcal{O}$. We start with $\mathcal{O} := \emptyset$. Some pairs (i, x) will be marked "forced", meaning that M_i accepts on input x. Some strings r will be marked "frozen", meaning that the behavior of $\mathcal{O}$ on inputs in S_r will not be changed after r is frozen. Initially, no pair is forced and no advice string is frozen. We start with Stage $n := 1$. In Stage n, we construct an oracle as follows.

Stage n. Consider the following condition, which we call $(*)_\theta$ for a threshold $\theta \in (0, 1)$.

> There exist an unfrozen string $r \in R_n$, an unforced pair $(i, x) \in [n] \times \{0,1\}^n$, an oracle $B \subseteq S_r$, and a certificate $y \in \{0,1\}^{2^n}$ such that
>
> $$\Pr_{M_i}\left[M_i^{\mathcal{O}\cup B}(x, y) = 1\right] \geq \theta,$$
>
> where the probability is taken over the coin flip of the randomized machine M_i.

We need to argue that the final oracle $\mathcal{O}$ can be computed in EXPH. It may not be possible to check $(*)_{3/4}$ exactly in EXPH, but using approximate counting in PH (Theorem 23), we can check whether a promise variant of $(*)_{3/4}$ is satisfied or not in EXPH. Specifically, by Theorem 23, there exists an algorithm in PH that, given as input n and $\mathcal{O}$ (which can be encoded as a binary string of length $2^{O(n)}$), accepts if $(*)_{3/4}$ holds, and rejects if $(*)_{1/2}$ does not hold. By the standard search-to-decision reduction, we obtain a PH-oracle polynomial-time algorithm S that outputs $r \in R_n$, $(i,x) \in [n] \times \{0,1\}^n$, $B \subseteq S_r$, and $y \in \{0,1\}^{2^n}$ that satisfy $(*)_{1/2}$ if $(*)_{3/4}$ holds, and outputs $\perp$ if $(*)_{1/2}$ does not hold. (Note that S runs in time $2^{O(n)}$ on inputs of length $2^{O(n)}$ with a PH oracle.)

While the search algorithm S outputs a certificate (r,i,x,B) for $(*)_{1/2}$ on input $(n,\mathcal{O})$ (instead of $\perp$), we do the following. Update $\mathcal{O} := \mathcal{O} \cup B$. Let r be frozen and let (i,x) be forced. Let $\mathcal{R}_\theta$ be the set of $r' \in R_n$ such that (r',i',x') is queried during the computation of $M_i^{\mathcal{O}}(x,y)$ for some (i',x') with probability at least θ over the internal randomness of M_i. By Theorem 24, some set A such that $\mathcal{R}_{2^{-2n}} \subseteq A \subseteq \{0,1\}^* \setminus \mathcal{R}_{2^{-2n-1}}$ can be enumerated in EXPH. Note that $|A| \leq 2^{2n+1} \cdot 2^n$ because M_i can make at most 2^n queries on each computation path. Let r' be frozen for every $r' \in A$. If S outputs $\perp$ (and thus $(*)_{3/4}$ does not hold), then let $r_n^* \in R_n$ be the first string that is not frozen. (We will later claim that such a string r_n^* exists.) Then, we add to $\mathcal{O}$ all the tuples (r_n^*,i,x) such that $(i,x) \in [n] \times \{0,1\}^n$ is forced. This completes the description of Stage n. Then we move on to the next stage $n' := 2^n + 1$, so that the construction in each stage is independent.

It is evident from the construction that $\mathcal{O} \in \mathsf{EXPH}$.

Fix any n and consider Stage n. We claim that there exists a string $r_n^* \in R_n$ that is not frozen at the end of Stage n. We say that $(*)$ is satisfied if S does not output $\perp$. Observe that the number of times that $(*)$ is satisfied is at most $n2^n$. The reason is that there are at most $n2^n$ pairs $(i,x) \in [n] \times \{0,1\}^n$, and each time $(*)$ is satisfied, (i,x) becomes forced. Each time $(*)$ is satisfied, at most $1 + 2^{2n+1} \cdot 2^n \leq 2^{3n+2}$ strings $r \in R_n$ can be frozen. Thus, the number of frozen strings in R_n in Stage n is at most $n2^n \cdot 2^{3n+2}$. Since there are at most n stages before Stage n, in total, there are at most $n \cdot n2^n \cdot 2^{3n+2} < 2^{5n} \leq |R_n|$ frozen strings in R_n. (Here, we used that $c := 5$.) Thus, there exists some unfrozen string $r_n^* \in R_n$, which shows that the construction of $\mathcal{O}$ is well defined.

It remains to show that for every pair $(i,x) \in [n] \times \{0,1\}^n$,

1. if $\Pr_{M_i}[M_i^{\mathcal{O}}(x,y) = 1] \geq \frac{3}{4}$ for some y, then (i,x) is forced (and thus $(r_n,i,x) \in \mathcal{O}$), and
2. if $\Pr_{M_i}[M_i^{\mathcal{O}}(x,y) = 1] \leq \frac{1}{4}$ for any y, then (i,x) is not forced (and thus $(r_n,i,x) \notin \mathcal{O}$).

Fix any n and consider any unforced pair $(i,x) \in [n] \times \{0,1\}^n$. We claim that for every $y \in \{0,1\}^{2^n}$, it holds that

$$\Pr_{M_i}\left[M_i^{\mathcal{O}}(x,y) = 1\right] < \frac{3}{4}.$$

Fix any $y \in \{0,1\}^{2^n}$. Let $\mathcal{O}_n$ be the oracle $\mathcal{O}$ right after the while loop of Stage n. Since $(*)_{3/4}$ is not satisfied at that point, for every $r \in R_n$ and for every oracle $B \subseteq S_r$,

$$\Pr_{M_i}\left[M_i^{\mathcal{O}_n \cup B}(x,y) = 1\right] < \frac{3}{4}.$$

Let $B \subseteq S_{r_n^*}$ be the set of strings added to $\mathcal{O}$ at the end of Stage n. Then, the final oracle $\mathcal{O}$ coincides with $\mathcal{O}_n \cup B$ on any inputs of length at most 2^n; thus, we obtain

$$\Pr_{M_i}\left[M_i^{\mathcal{O}}(x,y) = 1\right] = \Pr_{M_i}\left[M_i^{\mathcal{O}_n \cup B}(x,y) = 1\right] < \frac{3}{4}.$$

Next, fix any n and consider any forced pair $(i,x) \in [n] \times \{0,1\}^n$. Let $\mathcal{O}_{i,x}$ be the oracle right after (i,x) is forced and y be the certificate that satisfies $(*)$. Then, we have

$$\Pr\left[M_i^{\mathcal{O}_{i,x}}(x,y)=1\right] \geq \frac{1}{2}.$$

We claim that this probability does not decrease by $\frac{1}{4}$ even if we replace $\mathcal{O}_{i,x}$ with the final oracle $\mathcal{O}$. After (i,x) is forced, $(*)$ can be satisfied at most $n2^n$ times, and each time $(*)$ is satisfied, at most one S_r whose subset is added to $\mathcal{O}$. Thus, there are at most $n2^n$ unfrozen strings $r \in R_n$ such that some strings in S_r are added to $\mathcal{O}$. Note that any string r that is queried by M_i with probability 2^{-2n} is frozen right after (i,x) is forced. Thus, each unfrozen string can decrease the probability that $M_i^{\mathcal{O}}(x,y)$ accepts by 2^{-2n}. In total, the unfrozen strings can affect the probability of acceptance by $n2^n \cdot 2^{-2n} < \frac{1}{4}$. Thus, we obtain

$$\Pr\left[M_i^{\mathcal{O}}(x,y)=1\right] > \frac{1}{2} - \frac{1}{4} \geq \frac{1}{4}.$$ ◀

▶ **Corollary 63.** *If there is a* PSPACE*-relativizing proof that for every constant* $k \geq 1$*,* $\mathsf{MA}/_1 \not\subseteq \mathsf{i.o.SIZE}[n^k]$*, then* $\mathsf{L} \neq \mathsf{NP}$ *follows.*

7 Open Problems

We believe that the perspective of bounded relativization will lead to a better understanding of current proof techniques, in particular interactive proofs. There are many interesting questions left open:

1. Find some reason that "current techniques" have not been able to separate PSPACE from EXPH. We interpret the results in this paper as: If "current techniques" are PSPACE-relativizing and cannot separate PSPACE from EXPH, then they also cannot prove "slight improvements of known results" for which an EXPH-relativization barrier exists.
   The "conventional wisdom" seems to indicate that it is difficult for "current techniques" to separate PSPACE from EXPH, but we have not been able to find any formal justification. Instead, an optimist might treat the oracles in this paper as first steps for attacking the PSPACE vs. EXPH problem – IP = PSPACE indicates some *weakness* of PSPACE, thus making it vulnerable to separate from EXPH.
2. Find new oracles in EXPH that rules out "slight" improvements of known results. For example, can we show that EXPH-relativizing techniques cannot prove $\mathsf{MA} \not\subseteq \mathsf{SIZE}[n^k]$? (Santhanam's lower bound [65] is for the class $\mathsf{MA}/_1$, and it has been an open problem since then to eliminate the one-bit advice in the lower bound.) Can we show that EXPH-relativizing techniques cannot yield fast derandomization from strong uniform lower bounds for EXP, such as $\mathsf{EXP} \not\subseteq \mathsf{BPSUBEXP} \implies \mathsf{BPP} \subseteq \mathsf{i.o.heur\text{-}QuasiP}$?[14]
3. What can bounded relativization say about the Algorithmic Method for proving circuit lower bounds? Williams [78, 79] showed that non-trivial circuit-analysis (e.g., satisfiability or derandomization) algorithms for polynomial-size circuits imply that $\mathsf{NEXP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$. The general implication from non-trivial algorithms to circuit lower bounds does not relativize as shown in [75, Theorem 11]; however, it is unclear whether the corresponding oracle is in EXPH.

[14] Note that we might need to assume $\mathsf{P} \neq \mathsf{L}$ for this problem, since if $\mathsf{P} = \mathsf{L}$ then $\mathsf{EXP}^{\mathcal{O}[\mathrm{poly}]} = \mathsf{PSPACE}^{\mathcal{O}}$ for every oracle $\mathcal{O}$. It is known that strong uniform lower bound for PSPACE implies fast derandomization of BPP [70, 20], and the proofs are likely PSPACE-relativizing.

4. Finally, is there an oracle world under which Σ_2^p has linear-size circuits on infinitely many input lengths? Our construction showed that if we replace Σ_2^p by pr-MA then such an oracle world exists and can be computed in EXPH. This problem is connected to the Missing-String problem studied in [75].

References

1 Scott Aaronson. Oracles are subtle but not malicious. In *Conference on Computational Complexity* (CCC), pages 340–354. IEEE Computer Society, 2006. doi:10.1109/CCC.2006.32.

2 Scott Aaronson. The teaser. https://scottaaronson.blog/?p=3054, 2017. Accessed: Feb 6, 2023.

3 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1):2:1–2:54, 2009. doi:10.1145/1490270.1490272.

4 Eric Allender. Oracles versus proof techniques that do not relativize. In *SIGAL International Symposium on Algorithms*, volume 450 of *Lecture Notes in Computer Science*, pages 39–52. Springer, 1990. doi:10.1007/3-540-52921-7_54.

5 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.

6 Sanjeev Arora and Boaz Barak. *Computational Complexity – A Modern Approach*. Cambridge University Press, 2009. URL: http://www.cambridge.org/catalogue/catalogue.asp?isbn=9780521424264.

7 Sanjeev Arora, Russell Impagliazzo, and Umesh Vazirani. Relativizing versus nonrelativizing techniques: the role of local checkability. *Manuscript*, 1992. URL: https://people.eecs.berkeley.edu/~vazirani/pubs/relativizing.pdf.

8 Barış Aydınlıoğlu and Eric Bach. Affine relativization: Unifying the algebrization and relativization barriers. *ACM Trans. Comput. Theory*, 10(1):1:1–1:67, 2018. doi:10.1145/3170704.

9 László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Comput. Complex.*, 1:3–40, 1991. doi:10.1007/BF01200056.

10 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. doi:10.1007/BF01275486.

11 Theodore P. Baker, John Gill, and Robert Solovay. Relativizations of the P =?NP question. *SIAM J. Comput.*, 4(4):431–442, 1975. doi:10.1137/0204037.

12 Harry Buhrman, Lance Fortnow, and Rahul Santhanam. Unconditional lower bounds against advice. In *International Colloquium on Automata, Languages and Programming* (ICALP), pages 195–209, 2009. doi:10.1007/978-3-642-02927-1_18.

13 Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *Conference on Computational Complexity* (CCC), pages 8–12, 1998. doi:10.1109/CCC.1998.694585.

14 Harry Buhrman and Leen Torenvliet. Randomness is hard. *SIAM J. Comput.*, 30(5):1485–1501, 2000. doi:10.1137/S0097539799360148.

15 Samuel R. Buss and R. Ryan Williams. Limits on alternation trading proofs for time-space lower bounds. *Comput. Complex.*, 24(3):533–600, 2015. doi:10.1007/s00037-015-0104-9.

16 Richard Chang, Benny Chor, Oded Goldreich, Juris Hartmanis, Johan Håstad, Desh Ranjan, and Pankaj Rohatgi. The random oracle hypothesis is false. *J. Comput. Syst. Sci.*, 49(1):24–39, 1994. doi:10.1016/S0022-0000(05)80084-4.

17 Lijie Chen, Xin Lyu, and R. Ryan Williams. Almost-everywhere circuit lower bounds from non-trivial derandomization. In *Symposium on Foundations of Computer Science* (FOCS), pages 1–12. IEEE, 2020. doi:10.1109/FOCS46700.2020.00009.

18 Lijie Chen, Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. Relations and equivalences between circuit lower bounds and Karp–Lipton theorems. In *Computational Complexity Conference* (CCC), volume 137 of *LIPIcs*, pages 30:1–30:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CCC.2019.30.

19 Lijie Chen, Ron D. Rothblum, and Roei Tell. Unstructured hardness to average-case randomness. In *Symposium on Foundations of Computer Science* (FOCS), pages 429–437. IEEE, 2022. doi:10.1109/FOCS54457.2022.00048.

20 Lijie Chen, Ron D. Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds. In *Symposium on Foundations of Computer Science* (FOCS), pages 13–23. IEEE, 2020. doi:10.1109/FOCS46700.2020.00010.

21 Lijie Chen and Roei Tell. Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise. In *Symposium on Foundations of Computer Science* (FOCS), pages 125–136. IEEE, 2021. doi:10.1109/FOCS52979.2021.00021.

22 Scott Diehl and Dieter van Melkebeek. Time-space lower bounds for the polynomial-time hierarchy on randomized machines. *SIAM J. Comput.*, 36(3):563–594, 2006. doi:10.1137/050642228.

23 Lance Fortnow. The role of relativization in complexity theory. *Bull. EATCS*, 52:229–243, 1994.

24 Lance Fortnow. Time-space tradeoffs for satisfiability. *J. Comput. Syst. Sci.*, 60(2):337–353, 2000. doi:10.1006/jcss.1999.1671.

25 Lance Fortnow, Richard J. Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):835–865, 2005. doi:10.1145/1101821.1101822.

26 Lance Fortnow and Rahul Santhanam. Hierarchy theorems for probabilistic polynomial time. In *Symposium on Foundations of Computer Science* (FOCS), pages 316–324, 2004. doi:10.1109/FOCS.2004.33.

27 Lance Fortnow, Rahul Santhanam, and R. Ryan Williams. Fixed-polynomial size circuit bounds. In *Computational Complexity Conference* (CCC), pages 19–26. IEEE Computer Society, 2009. doi:10.1109/CCC.2009.21.

28 Lance Fortnow and Michael Sipser. Are there interactive protocols for coNP languages? *Inf. Process. Lett.*, 28(5):249–251, 1988. doi:10.1016/0020-0190(88)90199-8.

29 Gudmund Skovbjerg Frandsen and Peter Bro Miltersen. Reviewing bounds on the circuit size of the hardest functions. *Inf. Process. Lett.*, 95(2):354–357, 2005. doi:10.1016/j.ipl.2005.03.009.

30 Oded Goldreich. *Computational complexity – A conceptual perspective*. Cambridge University Press, 2008.

31 Shafi Goldwasser, Russell Impagliazzo, Toniann Pitassi, and Rahul Santhanam. On the pseudo-deterministic query complexity of NP search problems. In *Computational Complexity Conference* (CCC), volume 200 of *LIPIcs*, pages 36:1–36:22. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.36.

32 Hans Heller. On relativized exponential and probabilistic complexity classes. *Inf. Control.*, 71(3):231–243, 1986. doi:10.1016/S0019-9958(86)80012-2.

33 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In *Symposium on Foundations of Computer Science* (FOCS), pages 247–258, 2018. doi:10.1109/FOCS.2018.00032.

34 Shuichi Hirahara. Non-disjoint promise problems from meta-computational view of pseudorandom generator constructions. In *Computational Complexity Conference* (CCC), volume 169 of *LIPIcs*, pages 20:1–20:47. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.20.

35 Shuichi Hirahara. Unexpected hardness results for Kolmogorov complexity under uniform reductions. In *Symposium on Theory of Computing* (STOC), pages 1038–1051, 2020. doi:10.1145/3357713.3384251.

36 Shuichi Hirahara. Symmetry of information from meta-complexity. In *Computational Complexity Conference* (CCC), volume 234 of *LIPIcs*, pages 26:1–26:41. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CCC.2022.26.

37 Shuichi Hirahara, Zhenjian Lu, and Hanlin Ren. Bounded relativization. *Electron. Colloquium Comput. Complex.*, 30:70, 2023. URL: https://eccc.weizmann.ac.il/report/2023/070.

38 Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. An axiomatic approach to algebrization. In *Symposium on Theory of Computing* (STOC), pages 695–704. ACM, 2009. doi:10.1145/1536414.1536509.

39 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The power of natural properties as oracles. In *Computational Complexity Conference* (CCC), volume 102 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.7.

40 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002. doi:10.1016/S0022-0000(02)00024-7.

41 Russell Impagliazzo and Avi Wigderson. $\mathsf{P} = \mathsf{BPP}$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Symposium on Theory of Computing* (STOC), pages 220–229. ACM, 1997. doi:10.1145/258533.258590.

42 Russell Impagliazzo and Avi Wigderson. Randomness vs time: Derandomization under a uniform assumption. *J. Comput. Syst. Sci.*, 63(4):672–688, 2001. doi:10.1006/jcss.2001.1780.

43 Emil Jerábek. Dual weak pigeonhole principle, Boolean complexity, and derandomization. *Ann. Pure Appl. Log.*, 129(1-3):1–37, 2004. doi:10.1016/j.apal.2003.12.003.

44 Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. $\mathsf{MIP}^* = \mathsf{RE}$. *CoRR*, abs/2001.04383, 2020. doi:10.48550/arXiv.2001.04383.

45 Valentine Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. *J. Comput. Syst. Sci.*, 63(2):236–252, 2001. doi:10.1006/jcss.2001.1763.

46 Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *Symposium on Theory of Computing* (STOC), pages 73–79. ACM, 2000. doi:10.1145/335305.335314.

47 Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Inf. Control.*, 55(1-3):40–56, 1982. doi:10.1016/S0019-9958(82)90382-5.

48 Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Symposium on Theory of Computing* (STOC), pages 302–309. ACM, 1980. doi:10.1145/800141.804678.

49 Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos H. Papadimitriou. Total functions in the polynomial hierarchy. In *Innovations in Theoretical Computer Science* (ITCS), volume 185 of *LIPIcs*, pages 44:1–44:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ITCS.2021.44.

50 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

51 Oliver Korten. The hardest explicit construction. In *Symposium on Foundations of Computer Science* (FOCS), pages 433–444. IEEE, 2021. doi:10.1109/FOCS52979.2021.00051.

52 Leonid A. Levin. Randomness conservation inequalities; information and independence in mathematical theories. *Information and Control*, 61(1):15–37, 1984. doi:10.1016/S0019-9958(84)80060-1.

53 Richard J. Lipton and Anastasios Viglas. On the complexity of SAT. In *Symposium on Foundations of Computer Science* (FOCS), pages 459–464. IEEE Computer Society, 1999. doi:10.1109/SFFCS.1999.814618.

54 Zhenjian Lu, Igor Carboni Oliveira, and Rahul Santhanam. Pseudodeterministic algorithms and the structure of probabilistic time. In *Symposium on Theory of Computing* (STOC), pages 303–316. ACM, 2021. doi:10.1145/3406325.3451085.

55 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992. doi:10.1145/146585.146605.

56 Oleg B Lupanov. On the synthesis of switching circuits. *Doklady Akademii Nauk SSSR*, 119(1):23–26, 1958.

57 Peter Bro Miltersen, N. V. Vinodchandran, and Osamu Watanabe. Super-polynomial versus half-exponential circuit size in the exponential hierarchy. In *International Computing and Combinatorics Conference* (COCOON), pages 210–220, 1999. doi:10.1007/3-540-48686-0_21.

58 Cody D. Murray and R. Ryan Williams. Easiness amplification and uniform circuit lower bounds. In *Computational Complexity Conference* (CCC), volume 79 of *LIPIcs*, pages 8:1–8:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.CCC.2017.8.

59 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

60 Igor C. Oliveira. Randomness and intractability in Kolmogorov complexity. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 32:1–32:14, 2019. doi:10.4230/LIPIcs.ICALP.2019.32.

61 Igor C. Oliveira and Rahul Santhanam. Conspiracies between learning algorithms, circuit lower bounds, and pseudorandomness. In *Computational Complexity Conference* (CCC), pages 18:1–18:49, 2017. doi:10.4230/LIPIcs.CCC.2017.18.

62 Igor C. Oliveira and Rahul Santhanam. Pseudodeterministic constructions in subexponential time. In *Symposium on Theory of Computing* (STOC), pages 665–677, 2017. doi:10.1145/3055399.3055500.

63 Hanlin Ren and Rahul Santhanam. A relativization perspective on meta-complexity. In *International Symposium on Theoretical Aspects of Computer Science* (STACS), volume 219 of *LIPIcs*, pages 54:1–54:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.STACS.2022.54.

64 Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *FOCS*, pages 640–650. IEEE, 2022. doi:10.1109/FOCS54457.2022.00067.

65 Rahul Santhanam. Circuit lower bounds for Merlin–Arthur classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009. doi:10.1137/070702680.

66 Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992. doi:10.1145/146585.146609.

67 Claude E. Shannon. The synthesis of two-terminal switching circuits. *Bell Syst. Tech. J.*, 28(1):59–98, 1949. doi:10.1002/j.1538-7305.1949.tb03624.x.

68 Michael Sipser. A complexity theoretic approach to randomness. In *Symposium on Theory of Computing* (STOC), pages 330–335, 1983. doi:10.1145/800061.808762.

69 Larry J. Stockmeyer. The complexity of approximate counting (preliminary version). In *Symposium on Theory of Computing* (STOC), pages 118–126. ACM, 1983. doi:10.1145/800061.808740.

70 Luca Trevisan and Salil P. Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity*, 16(4):331–364, 2007. doi:10.1007/s00037-007-0233-x.

71 Christopher Umans. Pseudo-random generators for all hardnesses. *J. Comput. Syst. Sci.*, 67(2):419–440, 2003. doi:10.1016/S0022-0000(03)00046-1.

72 Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

73 N. V. Vinodchandran. A note on the circuit complexity of PP. *Theor. Comput. Sci.*, 347(1-2):415–418, 2005. doi:10.1016/j.tcs.2005.07.032.

74 Emanuele Viola. On approximate majority and probabilistic time. In *Conference on Computational Complexity* (CCC), pages 155–168. IEEE Computer Society, 2007. doi:10.1109/CCC.2007.16.

75 Nikhil Vyas and R. Ryan Williams. On oracles and algorithmic methods for proving lower bounds. In *Innovations in Theoretical Computer Science Conference* (ITCS), volume 251 of *LIPIcs*, pages 99:1–99:26. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ITCS.2023.99.

76 R. Ryan Williams. Inductive time-space lower bounds for SAT and related problems. *Comput. Complex.*, 15(4):433–470, 2006. doi:10.1007/s00037-007-0221-1.

77 R. Ryan Williams. Time-space tradeoffs for counting NP solutions modulo integers. *Comput. Complex.*, 17(2):179–219, 2008. doi:10.1007/s00037-008-0248-y.

78 R. Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

79 R. Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014. doi:10.1145/2559903.

80 Christopher B. Wilson. Relativized circuit complexity. *J. Comput. Syst. Sci.*, 31(2):169–181, 1985. doi:10.1016/0022-0000(85)90040-6.

Lower Bounds for Polynomial Calculus with Extension Variables over Finite Fields

Russell Impagliazzo
University of California San Diego, CA, USA

Sasank Mouli
Hyderabad, India

Toniann Pitassi
Columbia University, New York, NY, USA

Abstract

For every prime $p > 0$, every $n > 0$ and $\kappa = O(\log n)$, we show the existence of an unsatisfiable system of polynomial equations over $O(n \log n)$ variables of degree $O(\log n)$ such that any Polynomial Calculus refutation over $\mathbb{F}_p$ with M extension variables, each depending on at most κ original variables requires size $\exp\left(\Omega(n^2)/10^{\kappa}(M + n \log n)\right)$

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof complexity, Algebraic proof systems, Polynomial Calculus, Extension variables, $\mathsf{AC}^0[p]$-Frege

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.7

Funding *Russell Impagliazzo*: Supported by the Simons Foundation and NSF grant CCF-1909634.
Sasank Mouli: Supported by the Simons Foundation, NSF grant CCF-1909634 and the Swiss National Science Foundation project n. 200021_207429 / 1 "Ideal Membership Problems and the Bit Complexity of Sum of Squares Proofs".
Toniann Pitassi: Supported by NSF grant CCF-1900460, and by the IAS School of Mathematics.

Acknowledgements The authors would like to thank Paul Beame and Dmitry Sokolov for helpful discussions.

1 Introduction

A major goal of proof complexity is to show limits on the types of reasoning formalizable with concepts of small computational complexity, usually formalized as circuits from small circuit classes. This makes results in proof complexity analogous to (and often building on) results in circuit complexity. However, despite having strong lower bounds for the class $\mathsf{AC}^0[p]$ since the 1980's, ([18, 19]) it is still an open problem in proof complexity to establish superpolynomial (or even quadratic) lower bounds for the corresponding proof system $\mathsf{AC}^0[p]$-Frege.

Motivated by the lack of progress towards proving $AC^0[p]$-Frege lower bounds, [4] defined the Nullstellensatz (Nullsatz) proof system for refuting systems of unsolvable polynomial equations. Given a system of polynomial equations $\mathcal{P} = \{P_1 = 0, \ldots, P_m = 0\}$ in Boolean variables $x_1, \ldots, x_n$ (where we enforce the Boolean condition by adding the equations $x_i^2 - x_i = 0$ to $\mathcal{P}$), a Nullsatz refutation of $\mathcal{P}$ over a field $\mathbb{F}$ is a set of polynomials $\mathcal{Q} = \{Q_1, \ldots, Q_m\}$ such that $\sum_i P_i Q_i = 1$. The degree of the refutation is the maximum degree of the $P_i Q_i$'s, and the size is the sum of the sizes of the polynomials in $\mathcal{P}, \mathcal{Q}$. A dynamic version of Nullsatz, called the Polynomial Calculus (PC) was later defined in [10].

While these and later papers showed strong lower bounds for these proof systems, often these lower bounds were brittle in that the tautologies where lower bounds were proved also had small upper bounds under changes of variables. Our work is intended to address

the issue of proving algebraic proof lower bounds that are more robust under changes of variables. This can be viewed as a small but significant step towards proving lower bounds for $\mathsf{AC}^0[p]$-Frege, since the latter can simulate such changes of variables.

One reason for the brittleness of many of the earlier lower bounds is that these lower bounds were highly sensitive to the initial encoding. The known PC lower bounds hold for unsatisfiable CNF formulas which are converted to a corresponding system of unsolvable polynomial equations. Previous works established exponential PC lower bounds assuming a Boolean encoding, where the variables are Boolean, enforced by the initial equations $x_i^2 - x_i = 0$. Another natural encoding is the "Fourier" encoding which represents the constraints by polynomials over $\{-1, 1\}$-valued variables (by applying the linear transformation $x_i = 1 - 2x_i$ to the Boolean encoding). However under this second encoding, the size lower bounds all break down. This is due to the proof method, where size lower bounds were obtained from *degree* lower bounds. Over $\{0, 1\}$-valued variables, this can be accomplished by applying known size-degree tradeoffs for PC or by a random restriction argument to kill off all large monomials. But over $\{-1, 1\}$-valued variables, these methods no longer work: a generic size-degree tradeoff no longer holds (there are polynomial sized proofs of the Tseitin tautologies, although they require linear degree [8]), and since the monomials now correspond to parity equations, they are resilient to random restrictions.

However, recently, Sokolov [20] broke this barrier, and managed to prove exponential size lower bounds for PC refutations over the $\{-1, 1\}$ encoding. We note that while this may seem like a minor improvement over the known lower bounds which held for the $\{0, 1\}$-encoding, Sokolov had to invent a new and ingenious technique for proving size lower bounds. In this work, we generalize the methods of Sokolov to prove exponential PC lower bounds with up to $M = N^{2-\epsilon}$ extension variables which can depend on up to $\kappa = O(\log N)$ *original* variables (where N is the number of variables in the tautology). This shows that the Sokolov method can be used to prove highly robust lower bounds, that are not sensitive to local changes of variables. We state our result more precisely for two different choices of parameters, one that maximizes the size lower bound, and the other that maximizes the number of allowable extension variables.

▶ **Theorem 1** (high-end). *For n sufficiently large, there is a family of CNF tautologies F^{SEL} on $O(n \log n)$ variables with $poly(n)$ clauses of width $O(\log n)$ such that for any $M = npolylog(n)$ and $\kappa = O(\log \log n)$, any PC refutation over $\mathbb{F}_p$ of F^{SEL}, together with M κ-local extension axioms, requires size $2^{\Omega(n/polylog(n))}$.*

▶ **Theorem 2** (low-end). *For the same family of tautologies as above, there are $0 < \alpha, \beta, \gamma < 1$ so that, for $M = n^{1+\alpha}, \kappa = \beta log n$, any PC refutation of F^{SEL} together with any M κ-local extensions over $\mathbb{F}_p$ requires size $2^{\Omega(n^\gamma)}$.*

We remark that our extension variables are only allowed to depend on the original variables, and not on previously defined extension variables. (In the more general case where extension variables are defined recursively, the proof system corresponds to $AC^0[p]$-Frege, where the level of recursion corresponds to the $AC^0[p]$ circuit depth.) Thus our lower bound can be (roughly) seen as proving exponential lower bounds for the following restricted class of depth-2.5 PC refutations. First, the refutation is given a *new* set of M variables, $z_1, \ldots, z_M$, and is allowed to define a corresponding set of M κ-local polynomials $Q_1, \ldots, Q_M$ (where each Q_i can only depend on κ original variables). Lines in the refutation are polynomials over the original variables, plus the new *extension* variables (which are placeholders for the Q_i's). Substituting the Q_i's for the new variables gives a set of depth 2.5 algebraic circuits using a pre-specified set of κ-local functions at the bottom layer of the circuit.

1.1 Related Work

The work that inspired us and that is most related to our result is the recent paper by Sokolov [20], proving exponential lower bounds on the size of PC refutations of CNF formulas, where the variables take on values in $\{1, -1\}$. We generalize Sokolov's result to hold over any finite field, even with the addition of superlinear many extension variables, each depending arbitrarily on a small number of original variables. Thus our result can be alternatively viewed as making progress towards proving exponential lower bounds for depth-3 $AC^0[p]$-Frege, for a family of CNF formulas.

We note that for systems of polynomial equations over the rationals, a body of recent work establishes much stronger lower bounds. First, [13] proved lower bounds for subsystems of IPS over the rationals by restricted classes of circuits, including low-depth formulas, multilinear formulas and read-once oblivious branching programs. Secondly, Alekseev [2] proved exponential lower bounds on the bit complexity of PC proofs with an arbitrary number of extension variables of unbounded depth over the rationals. Andrews and Forbes [3] prove quasipolynomial lower bounds on the circuit size of constant-depth IPS proofs for a different family of polynomials over the rationals; however, their hard instances do not have small-size constant-depth circuits. Finally, [14] establish a similar lower bound as [3], but for hard instances that have small constant-depth circuits.

We remark that these lower bounds are incomparable to ours for several reasons. First, they do not hold for finite fields, and secondly, the choice of hard polynomials are inherently nonboolean: [13, 2, 14] use the subset sum principle which when translated to a propositional statement is no longer hard, and the hard polynomials in [3] have logarithmic depth. Thus on the one hand they establish superpolynomial lower bounds for *much* stronger subsystems of IPS, but on the other hand, they do not translate to lower bounds for propositional proofs in the sense of Cook-Reckhow [11]. In particular, they don't imply lower bounds for proof systems dealing with Boolean formulae.

1.2 Our Result: Proof Overview

The standard way of proving size lower bounds for PC for an unsatisfiable formula F for Boolean-valued variables dates back to the celebrated superpolynomial lower bounds for Resolution [15, 7], where the basic tool is to reduce size lower bounds to degree lower bounds (or in the case of Resolution, size to clause-width) by way of either a general size-depth tradeoff, or by a more general random restriction argument. At a high level, both methods iteratively select a variable that occurs in a lot of high-degree terms, set this variable to zero (to kill off all high-degree terms containing it), while also ensuring (possibly by setting additional variables) that F remains hard to refute after applying the partial restriction. After applying this size-to-degree reduction, the main technical part is to prove degree lower bounds for the restricted version of F.

As mentioned in the Introduction, over the $\{-1, 1\}$ basis, the size to degree reduction breaks down. In fact, no generic reduction to degree can exist since random XOR instances over this basis require linear degree but have polynomial size PC refutations. Moreover, we lacked *any* method for proving PC lower bounds for unsatisfiable CNFs over the basis $\{-1, 1\}$, and more generally over an arbitrary linear transformation of the variables. In [16], we highlighted this as an open problem, noting that it is a necessary step toward proving superpolynomial $AC^0[2]$-Frege lower bounds, a major open problem in proof complexity.

Recently, Sokolov [20] made significant progress by proving exponential lower bounds for PC (as well as for SOS) for random CNF formulas over the domain $\{-1, 1\}$, by developing new formula-specific techniques to reduce size to degree over this domain. As this is the starting point for our work, we begin by describing the main method in [20] for reducing size to degree for certain families of formulas over $\{-1, 1\}$.

Let Π be an alleged PC refutation of F of small size which includes the axioms $w^2 = 1$ for all variables w. The first step in Sokolov's argument is to show how to remove all high degree terms containing a particular variable w, provided that w is *irrelevant* – meaning that it does not occur in any of the initial polynomials other than the equation $w^2 = 1$. Intuitively, we want to show that if our unsatisfiable system of polynomial equations doesn't contain w, then we should be able to eliminate high degree terms containing w altogether from the refutation. To show this, Sokolov introduced a new operation termed *Split* where he writes each line q in the refutation as $q_0 + q_1 w$, and proves by induction that if we replace each line q by the pair of lines q_0, q_1, then it is still a valid refutation of F (and no longer contains w). While the Split operation removes w from the proof, it doesn't kill off high degree terms. The crucial insight is that although this doesn't directly kill off high degree terms, a slightly different measure of degree (called Quadratic degree) can be used instead, since removing w via the Split operation removes all high Quadratic degree terms that w contributed to, and secondly low Quadratic degree implies low ordinary degree. The second and easier step in Sokolov's argument uses specific expansion properties of F to show that for any variable w, there exists a small restriction ρ (to some of the other variables) such that w becomes irrelevant under ρ.

Our main theorem significantly generalizes Sokolov's lower bound by proving exponential lower bounds for an unsatisfiable CNF formulas F, even when we allow the axioms $\mathcal{P}$ to contain superlinear many extension axioms, provided that each extension axiom depends on a small number of original variables. Note that the variables of F are Boolean, but the extension variables are not restricted to being Boolean. In particular, it may be the case that zero is not in the support of an extension variable (i.e. the set of all possible values that can be assigned to it without violating any Boolean axioms), for example if extension variable z is defined by the equation $z = x - 2$, then z cannot be set to zero without falsifying the Boolean axiom $x^2 - x = 0$ for x. Intuitively we will handle extension variables z that cannot be set to zero in a similar manner to Sokolov, by first isolating z, and then generalizing the Split operation in order to kill off all large Quadratic degree terms that contain z. However, dealing with a general set of extension axioms presents new technical challenges that we address next.

Our first idea is to design the unsatisfiable formula F carefully so that we can force variables to be irrelevant in a more modular way. Specifically, let $F(x_1, \ldots, x_n)$ be an expanding unsatisfiable k-CSP formula with $m = O(n)$ constraints, such that any subset of $m' = \epsilon m$ constraints is unsatisfiable and requires proofs of large PC degree. We define an unsatisfiable formula F^{SEL} (based on F) that intuitively states that there is a subset S of $m' = \epsilon m$ constraints of F (as chosen by new selector variables $\mathbf{y}$) that is satisfiable. We will prove lower bounds on the set of constraints F^{SEL} even with the addition of an arbitrary set of extension axioms satisfying the conditions mentioned earlier. In order to make a variable of F^{SEL} irrelevant, we will simply make sure that our eventual assignment to the selector variables ($\mathbf{y}$) avoids constraints of F that contain this variable (we can also make a selector variable irrelevant in a slightly more complicated way, details are left to the relevant section).

A second challenge that we face (that doesn't come up in Sokolov's proof) is that extension variables may be defined so that originally they can be consistently set to zero, but can change status after applying a restriction. For example, suppose the proof uses the extension

axiom $z = x_1x_2 + x_1$. Then zero is in the support of z (since we can set $x_1 = x_2 = 0$), but if we set $x_1 = 1$, then zero is no longer in the support of z. In order to deal with this dynamically changing status of variables, our notion of Quadratic degree must pay attention to which category each of the extension variables is in at any particular time, and make sure that we do not lose progress that was made earlier due to variables changing from initially containing zero to disallowing zero in their support. Fortunately we observe that variables can only change unidirectionally, (since the support of a variable cannot increase under a restriction) and this is crucial for arguing that our measure of Quadratic degree always decreases so that we continually make progress.

Finally, we also have to generalize Sokolov's Split operation, which was previously defined only for $\{-1, 1\}$ variables. We give a generalization of how to do the Split for arbitrary valued variables.

2 Preliminaries

▶ **Definition 3** (Polynomial Calculus/Polynomial Calculus Resolution). *Let $\Gamma = \{P_1 \dots P_m\}$ be an unsolvable system of polynomials in variables $\{x_1 \dots x_n\}$ over $\mathbb{F}$. A* PC *(Polynomial Calculus) refutation of Γ is a sequence of polynomials $\{R_1 \dots R_s\}$ such that $R_s = 1$ and for every $\ell \in [s]$, $R_\ell \in \Gamma$, R_ℓ is either a polynomial from Γ, or is obtained from two previous polynomials R_j, R_k, $j, k < \ell$ by one of the following derivation rules:*

$R_\ell = \alpha R_j + \beta R_k$ for $\alpha, \beta \in \mathbb{F}$

$R_\ell = x_i R_k$ for some $i \in [n]\}$

The size of the refutation is $\sum_{\ell=1}^{s} |R_\ell|$, where $|R_\ell|$ is the number of monomials in the polynomial R_ℓ. The degree of the refutation is $\max_\ell deg(R_\ell)$.

A PCR (Polynomial Calculus Resolution) refutation is a PC refutation over the set of Boolean variables $\{x_1 \dots x_n, \bar{x}_1 \dots \bar{x}_n\}$ where $\{\bar{x}_1 \dots \bar{x}_n\}$ are twin variables of $\{x_1 \dots x_n\}$ i.e. the equations $x_i^2 - x_i = 0$, $\bar{x_i}^2 - \bar{x_i} = 0$ and $x_i\bar{x}_i = 0$ are treated as axioms.

▶ **Definition 4** (PC plus Extension Axioms). *Let $\Gamma = \{P_1 \dots P_m\}$ be a set of polynomials in variables $\{x_1 \dots x_n\}$ over a field $\mathbb{F}$. We will refer to the polynomials in Γ as (initial) axioms. Let $\mathbf{z} = z_1 \dots z_M$ be new extension variables with corresponding extension axioms $z_j - Q_j(x_1 \dots x_n)$. A* PC + Ext *(*PC *plus extension) refutation of Γ with M extension axioms $Ext = \{z_j - Q_j(x_1 \dots x_n)\}$ is a* PC *refutation of the set of polynomials $\Gamma' = \{P_1 \dots P_m, z_1 - Q_1 \dots z_M - Q_M\}$. An extension axiom $z_j = Q_j(x_1 \dots x_n)$ is κ-local if Q_j is a κ-junta; that is, if the polynomial Q_j defining z_j involves at most κ of the $\mathbf{x}$-variables. We say that Π is a (M, κ) − PC + Ext refutation of Γ if it is a* PC + Ext *refutation of Γ with M extension axioms, each of which are κ-local. The size of the refutation is total size of all lines in the refutation, including the polynomials in Γ plus the extension axioms (where the size of a line $P \in \Pi$ is the number of monomials in P).*

We note that our definition of extension axioms is more limited than the general notion of extension axioms. Here we only allow the extension variables to depend on the *original* variables from Γ; the more general definition allows the extension variables to depend on the original $\mathbf{x}$-variables, and also on other extension variables.

▶ **Definition 5** (k-local CSPs). *A constraint C_i over Boolean variables $\{x_1, \dots, x_n\}$ is simply a Boolean formula over these variables. C_i is a k-local constraint if C_i depends on at most k variables. A k-CSP $\mathcal{C} = C_1 \wedge \dots \wedge C_m$ over $\{x_1, \dots, x_N\}$ is the conjunction of a set of k-local constraints.*

We translate a k-CSP formula into a system of polynomial equations using the standard PCR translation which we define next.

▶ **Definition 6** (Converting k-CSPs into Polynomial Equations). *Let C be a k-local constraint over variables $x_{i_1}, \ldots, x_{i_k}$. We convert C to a polynomial equation, $p(C)$, using the translations $p(x_{i_j}) = 1 - x_{i_j}$, $p(\neg A) = 1 - p(A)$, $p(A \vee B) = p(A) \cdot p(B)$. It is easy to check that for any Boolean assignment α to the underlying variables, $C(\alpha) = 1 \leftrightarrow p(\alpha) = 0$, and $C(\alpha) = 0 \leftrightarrow p(\alpha) = 1$.*

A k-CSP $\mathcal{C} = C_1 \wedge \ldots \wedge C_m$ over $\{x_1, \ldots, x_n\}$ converts to a set of polynomial equations $\{E_j \mid j \in [m]\} \cup \{B_i \mid i \in [n]\}$ over $\{x_1, \ldots, x_n\} \cup \{\bar{x}_1, \ldots, \bar{x}_n\}$ where E_j is the polynomial equation $p(C_j)$ In addition, we add the Boolean axioms $\{B_i \mid i \in [n]\}$, where $B_i = \{x_i^2 - x_i = 0,\ \bar{x_i}^2 - \bar{x_i} = 0,\ x_i\bar{x_i} = 0\}$ which force $x_i, \bar{x_i}$ to be zero-one valued, and force exactly one of $x_i, \bar{x_i}$ to be one.

3 The Hard Formulas

We distinguish between the case $p = 2$ and the case $p > 2$, and concentrate on the latter. This is because the case $p = 2$ does not require any new technical ideas, and we can pick from a large number of known hard tautologies for this case, such as random CNF's. Over $\mathbb{F}_2$, every extension variable is zero-one valued, and so standard size-degree tradeoffs pertain even with respect to extension variables. Also, κ-local extension variables can change the degree by at most a factor of κ, therefore a degree lower bound of $\Omega(n)$ for the original tautology over n variables implies a degree lower bound of $\Omega(n/\kappa)$ after adding κ-local extension variables. Known size-degree tradeoffs imply that the degree must be at least square root of the number of variables in order to obtain exponential size lower bounds, this immediately gives a lower bound tolerating close to n^2/κ^2 many κ-local extension variables [10, 6, 17].

Over any field, there are unsatisfiable families of k-CNF formulas (e.g. the Tseitin tautologies as well as random parity equations) that require linear degree but have polynomial sized proofs with a linear number of extension variables [8, 6]. Therefore formulas that require high PC degree are not sufficient. Instead we will create our hard examples by taking a hard instance and then using selector variables to pick out a subset of the constraints. Similar ideas were used earlier (e.g., [12]). In more detail, our underlying hard unsatisfiable formulas, $\{F_{n,k}^{SEL}\}$, will be constructed from a family of k-CSP formulas, $F_{n,k}$, that have the property that any sufficiently large subset of the constraints of $F_{n,k}$ is unsatisfiable and still requires large PC degree.

▶ **Definition 7.** *Let $F_{n,k} = \{E_j \mid j \in [m]\} \cup \{B_i \mid i \in [n]\}$ be the system of degree-k polynomial equations over $\mathbf{x} = \{x_i, \bar{x}_i \mid i \in [n]\}$, obtained by converting a size-m k-CSP as given by Definition 6. For convenience, we will index the polynomial equations E_j in binary notation, so for example if $b_1 \ldots b_{\log m} \in \{0,1\}^{\log m}$ is the binary notation for $j \in [m]$, we will write E_j as $E_{b_1 \ldots b_{\log m}}$. We define a new set of polynomial equations $F_{n,k}^{SEL}$ with parameters m, m' as follows. The variables are $\mathbf{x} \cup \mathbf{y}$, where $\mathbf{x}$ are the original variables of $F_{n,k}$ and $\mathbf{y} = \{y_{i,j},\ \overline{y_{i,j}} \mid i \in [m'],\ j \in [\log m]\}$ are new "pigeon" variables. Let E^{SEL} be the following set of equations, where $y_i \neq b_1 \ldots b_{\log m}$ abbreviates the monomial $\prod_{b_j=1} y_{i,j} \prod_{b_j=0} \overline{y_{i,j}}$:*

- **(i)** $\forall i \in [m'], \forall b_1 \ldots b_{\log m} \in \{0,1\}^{\log m}$, $(y_i \neq b_1 \ldots b_{\log m}) \cdot E_{b_1 \ldots b_{\log m}} = 0$;
- **(ii)** $\forall i, i' \in [m']$, $i \neq i'$, $\forall b_1 \ldots b_{\log m} \in \{0,1\}^{\log m}$, $(y_i \neq b_1 \ldots b_{\log m}) \cdot (y_{i'} \neq b_1 \ldots b_{\log m}) = 0$.

$F_{n,k}^{SEL}$ consists of the polynomial equations E^{SEL} together with the Boolean axioms $B_{i,j} = \{y_{i,j}^2 - y_{i,j} = 0,\ \overline{y_{i,j}}^2 - \overline{y_{i,j}} = 0,\ y_{i,j}\overline{y_{i,j}} = 0\}$ for all $i \in [m'], j \in [m]$.

Intuitively we think of the $\mathbf{y}$ variables as a mapping from m' pigeons to m holes, where the holes correspond to the m axioms/constraints from E. For $i \in [m']$, the i^{th} "pigeon" y_i selects a hole (an equation from E).

The first set of polynomial equations in E^{SEL} states that if pigeon y_i selects the equation $E_{b_1 \dots b_{\log m}}$, then this equation must be satisfied; the second set of equations in E^{SEL} states that the mapping is one-to-one and thus altogether the $\mathbf{y}$ selector variables choose a subset E' of exactly m' equations from E. Thus $F^{SEL}_{n,k}$ asserts that there exists a subset of m' constraints of $F_{n,k}$ (chosen by the $\mathbf{y}$-variables) that are satisfiable.

Throughout this paper, the $\mathbf{x}$-variables are the variables that underly $F_{n,k}$; the $\mathbf{y}$-variables are the selector/pigeon variables described above that choose a subset of m' constraints from $F_{n,k}$, and the extension variables used in the PC + Ext refutation will be the $\mathbf{z}$-variables.

Our hard instances will be $F^{SEL}_{n,k}$, with $m = 10n$, $m' = (1-\epsilon)m$, where $F_{n,k}$ is (the polynomial translation of) an unsatisfiable k-CSP formula with $m = 10n$ k-local constraints over variables $\mathbf{x} = x_1 \dots x_n$, satisfying the follow property:

▶ **Property 8.** *Every subset of $(1-\epsilon)m'$ constraints is unsatisfiable and requires linear* PC *degree.*

The following Theorem shows that for sufficiently large n, such formulas exist. Similar proofs have appeared in several papers (e.g., [5]) but we give a proof in the Appendix for completeness.

▶ **Theorem 9.** *Let $m = 10n$. Then there exists constants $k > 0$, $0 < \epsilon < 1$ such that for sufficiently large n, there exists k-CSP formulas $\{F_{n,k}\}$ with m constraints such that Property 8 holds with $m' = (1-\epsilon)m$.*

4 The Lower Bound

4.1 Technical Proof Overview

Conventionally, proof size lower bounds are reduced to degree lower bounds, a single step of which involves finding a *heavy* variable that occurs in a large fraction of high degree terms of the proof and setting it to zero. In our setting, if the heavy variable turns out to be an extension variable, z with extension axiom $z = Q(\mathbf{x}, \mathbf{y})$, it may be *Nonsingular* meaning that we cannot set $z = 0$ (without falsifying the extension axiom or a Boolean axiom), as opposed to *Singular* variables which can be set to zero in a consistent way[1]. In this case, we cannot simply eliminate the high degree terms containing z by setting $z = 0$. Sokolov [20] focused on the case where variables are over the ± 1 basis instead of the usual Boolean one, which is the simplest case where all variables are Nonsingular. Sokolov introduced *Quadratic degree* as a measure to be used instead of degree. Quadratic degree essentially measures the maximal degree of the *square* of each polynomial P occurring in the proof. For a ± 1 variable z, $z^2 = 1$, so squaring a polynomial P on ± 1 variables removes the contribution of a term $t \in P$ as it gets squared out, and what remain are the terms $t_1 t_2$ for t_1, $t_2 \in P$. Since any variable that appears in both terms gets squared out, the degree of these terms measures the symmetric difference between such terms, and this turns out to be a key complexity measure while dealing with Nonsingular variables. Sokolov showed that a refutation of low Quadratic degree can be turned into one of low degree. Thus the presence of Nonsingular variables

[1] This terminology is taken from singular and nonsingular matrices, since the key property we use is that a variable z is Nonsingular if and only if z^{p-2} is a "multiplicative inverse" of z, i.e. $z^{p-1} = 1$

is not necessarily a problem as long as the Quadratic degree of each line is low. Sokolov also introduced an operation *Split* that acts on a proof line by line in order to remove the contribution to Quadratic degree of any particularly *heavy* Nonsingular variable z, in the special case where they always take on values in ± 1, by replacing a line $P = P_1 z + P_0$ in the refutation with the lines P_1 and P_0. Sokolov managed to show that for some well chosen tautologies, the new Split lines still form a valid refutation of a hard subset of axioms. The crucial observation here is that this splitting of lines has eliminated from the square of the proof all pairs of terms whose product contained z. Thus, repeated application of Split would lead to contradiction of known degree lower bounds.

The first step for us was to generalize the notions of Quadratic degree and *Split* to any finite field. Motivated by the above definition of Quadratic degree, we generalize it as follows. Given two terms t_1 and t_2, a Nonsingular variable z contributes to the Quadratic degree between t_1 and t_2 if and only if it appears with different exponents in them, i.e. $z^i \in t_1$ and $z^j \in t_2$, for $i \neq j$. A Singular variable z contributes if and only if it appears in one of the terms with a nonzero exponent. The Quadratic degree of t_1 and t_2 is the total number of such variables z that contribute. Generalizing the *Split* operation proved a bit more difficult. We first focus on the case over $\mathbb{F}_p$ analogous to Sokolov's, where we have a variable z such that the identity $(z-a)(z-b) = 0$ holds for some constants $a, b \neq 0$ in the field. Note that a line $P(z)$ of the proof is of the form $P_{p-2}z^{p-2} + \cdots + P_1 z + P_0$. In the case of ± 1 variables, $p = 3$ and thus the contribution by z to Quadratic degree comes just from the interaction between two polynomials P_1 and P_0. Therefore separating P_1 and P_0 into different lines removes this contribution entirely. In the general case, however, the contribution by z to Quadratic degree is the sum total of interactions between polynomials P_i and P_j for every pair $i, j < p-1$ such that i and j are distinct. We show how to separate P into two lines R_1, R_0 such that the interaction between P_i and P_j is completely removed, for any i, j satisfying $a^{i-j} \neq b^{i-j}$, or in other words, z^i and z^j are linearly independent over the two values that z takes. Let $R(z) = R_1 z^i + R_0 z^j$ be a polynomial such that R agrees with P for each possible value of z, i.e. $R(a) = P(a)$ and $R(b) = P(b)$. Since z^i and z^j are linearly independent over values $\{a, b\}$, these two equations can be solved for their coefficients R_1, R_0, expressed in terms of $P_{p-2} \ldots P_0$. On closer observation, we find that P_i does not occur in the expression for R_1 and similarly P_j does not occur in R_0, and therefore we have successfully broken P into lines R_1 and R_0 while separating P_i and P_j. It is straightforward to show that this new set of lines forms a valid refutation, but an essential assumption we make here is that the initial axioms are free of z, except for $(z-a)(z-b) = 0$.

We now move to dealing with the case of a more general extension variable z with the extension axiom $z - Q$, where $Q(\mathbf{x}, \mathbf{y})$ is a polynomial that can depend on at most κ variables. Let H be the set of all pairs of terms (t_1, t_2) in a line of a given refutation that have high Quadratic degree between them. We would like to emulate Sokolov's strategy of eliminating this set of pairs from the refutation to drop its Quadratic degree. If an extension variable z which is Singular appears heavily in H, we apply the restriction that sets it to zero (which exists by the definition of Singular). In the case that z is Nonsingular, our goal is to reduce it to the above case in order to apply Split. But first, we will have to choose a "good" pair of indices ℓ_1, ℓ_0 such that Splitting them is effective in reducing H. We observe that for any pair of indices i, j, the set of pairs (t_1, t_2) in H such $z^i \in t_1$ and $z^j \in t_2$ is disjoint from the similar set defined for a distinct pair i', j'. Therefore by averaging we can pick a good pair ℓ_1, ℓ_0 that covers at least a $1/p^2$ fraction of z's appearances in H. We now have to reduce z to take on two distinct values a, b in order to apply Split, but these values need to be such that $a^{\ell_1 - \ell_0} \neq b^{\ell_1 - \ell_0}$. We show that there is a decision tree process (Lemma 22) that

queries the variables underlying Q such that it is always possible to reduce z to the form $(b-a)w^* + a$, where a, b are useful to separate the indices ℓ_1, ℓ_0. It is fairly easy to see as a result of the discussion so far that if we are able to apply Split on z with indices ℓ_1, ℓ_0 at this stage, it causes a sizable reduction in H.

We are now almost ready to apply Split, but we still have to meet the requirement that the axioms are free of z. Since z is an extension variable it appears only in the extension axiom which has now been reduced to the form $(b-a)w^* + a$, and so the only way to remove this axiom is to make a substitution for $w^* = (z-a)/(b-a)$ in terms of z. This would get rid of this extension axiom and take the Boolean axiom for w^* to $(z-a)(z-b) = 0$ just like we need, but if w^* appears in any of the other axioms this substitution just creates new copies of z. Therefore we need to remove w^* from all the other axioms before we try to make this substitution. Here is where we make use of the structure of our tautology $F^{SEL}_{n,k}$ by defining an operation **Cleanup** which can remove any Boolean variable w^* from the axioms without actually setting it to a constant value. **Cleanup** also restores the structure of our tautology so that we are always working with a subset of equations and pigeons from $F^{SEL}_{n,k}$ that are untouched by previous restrictions. We describe this operation in detail in Section 4.5.1.

Once we perform the above cleanup operations we are ready to make the substitution for $w^* = (z-a)/(b-a)$ in terms of z to satisfy the requirements for Split. We are met with a final hurdle here: this substitution can potentially increase the number of pairs of terms in H. Fortunately it can be resolved by a simple case analysis: if the blowup is too large it must have been the case that w^* appeared frequently in H, and so setting it to zero will reduce H without the need for Split. Otherwise, Split is able to offset this blowup.

Therefore we have demonstrated above how to reduce the size of the high Quadratic degree set H by a constant fraction. Performing this for sufficiently many iterations would remove H entirely and lower the Quadratic degree of any refutation. We then use a generalized version of Sokolov's argument that low Quadratic degree implies low degree in order to switch to a low degree refutation. For a small sized refutation, the number of iterations needed is bounded and thus we are able to keep most of the pigeons and equations alive at the end. We then select a hard subset of equations by assigning all remaining pigeons, and expand any remaining extension variables in order to obtain a low degree refutation of these equations, towards a contradiction.

4.2 Singular and Nonsingular variables

Let us fix the finite field $\mathbb{F}_p$, $p > 2$ for the rest of the article. We also fix a set of unsatisfiable polynomials F over Boolean variables $\mathbf{x} \cup \mathbf{y}$, and a set of extension axioms Ext of the form $z - Q$ over variables $\mathbf{z}$. Whenever we refer to a refutation Π, we assume that it is a $\mathsf{PC} + \mathsf{Ext}$ refutation of $F \cup Ext$.

▶ **Definition 10** (Support of a variable). *Let $z - Q(w_{i_1}, \ldots, w_{i_\kappa}) = 0$ be a κ-local extension axiom associated with z. We define the set $vars(Q) = \{w_{i_1}, \ldots, w_{i_\kappa}\}$ and sometimes write $vars(z)$ to denote $vars(Q)$, the set of variables that z depends on. The support of z, $supp(z) \subseteq [0, p-1]$, is equal to the set of all values $a \in [0, p-1]$ such that there exists a Boolean assignment α to the variables of Q such that $Q(\alpha) = a$. Sometimes we also indicate this by $supp(Q)$.*

We extend the definition of support also to Boolean variables. For a Boolean variable w, $supp(w) = \{0, 1\}$ as enforced by the Boolean axiom $w^2 = w$.

▶ **Definition 11** (Singular and Nonsingular variables w.r.t. Ext). *Let Ext be a set of extension axioms and let z be an extension variable with an axiom in Ext. We say that z is Singular*

w.r.t. Ext iff $0 \in supp(z)$*; otherwise we say that* z *is Nonsingular w.r.t. Ext. Any Boolean variable is considered Singular by default, independent of the set Ext, since zero always belongs to its support. For a term* t*, let* $sing(t)$ *be the subterm of* t *containing the Singular variables in* t*, and let* $nsing(t)$ *be the subterm of* t *containing the Nonsingular variables.*

Note that for a Singular extension variable z, it is possible to set z to zero, However, we note that this may falsify other polynomial equations in F. For example, if $xy = 0$ is a polynomial equation in F, and the extension axiom for z is $z - 1 + xy = 0$, then setting $x = y = 1$ forces $z = 0$, but this falsifies $xy = 0$.

▶ **Definition 12.** *Let* $A \subseteq [1, \ldots, p-1]$*,* $A \neq \emptyset$*. Define* $\ell(A)$ *to be the least* $\ell \in [1, p-1]$ *such that the set* $\{a^\ell \mid a \in A\}$ *is singleton. For a Nonsingular* z*, define* $\ell(z) = \ell(supp(z))$*.*

▶ **Lemma 13.** *Let* z *be a Nonsingular extension variable with extension axiom* $z - Q = 0$*. Then the following polynomial equations are implied by (and therefore derivable from) the extension axiom for* z *plus the Boolean axioms for all variables in* $vars(Q)$*, in degree at most* $|vars(Q)|$*.*

1. $z - Q' = 0$*, where* Q' *is the multilinear version of* Q*;*
2. *For any* $A' \subseteq [0, p-1]$ *such that* $supp(z) \subseteq A'$*,* $\Pi_{a \in A'}(z - a) = 0$*;*
3. $z^{\ell(z)} - c = 0$ *for some* $c \in \mathbb{F}_p^*$*.*

In particular, if z is Nonsingular, then the polynomial equation $z^{p-1} - 1 = 0$ is implied by $z - Q = 0$ together with the Boolean axioms for $vars(Q)$.

Proof. Let $z - Q(w_{i_1}, \ldots, w_{i_\kappa}) = 0$ be the extension axiom for z, and let $supp(z) = A \subseteq A' \subseteq [1, p-1]$. First, we can derive the multilinear version of Q, Q', from Q together with the Boolean axoms $w^2 - w = 0$ for all $w \in vars(Q)$. Secondly, by definition, $supp(z) = A$ means that the allowable values for z over Boolean assignments to $vars(Q)$ are the values in A. Therefore, $z - Q = 0$ together with the Boolean axioms $w^2 - w = 0$ for all $w \in vars(Q)$ implies $\Pi_{a \in A}(z - a) = 0$. Furthermore, this polynomial has a PC derivation, by the derivational completeness of PC. Since $A \subseteq A'$, $\Pi_{a \in A'}(z - a) = 0$ is a weakening of $\Pi_{a \in A}(z - a) = 0$ and is therefore derivable from $\Pi_{a \in A}(z = a) = 0$. Lastly, we will argue that there exists some constant $c \in \mathbb{F}_p^*$ such that $z^{\ell(A)} - c = 0$ is semantically implied by $z - Q = 0$ plus the Boolean axioms for $vars(z)$ and therefore is derivable from these axioms. Since the only allowable values for z under the Boolean axioms are the values in A, and since by definition of $\ell(A)$, for every $a \in A$, $a^{\ell(A)} = c$ for some $c \in \mathbb{F}_p^*$, it follows that $z^{\ell(A)} - c = 0$. ◀

▶ **Definition 14.** *For a term* t *and a variable* w*,* $\deg(t, w)$ *is equal to the degree of* w *in* t*. If* w *is Nonsingular, then* $w^{p-1} = 1 \mod p$*, so* $\deg(t, w) < p - 1$*. On the other hand if* w *is Singular then we have* $w^p = w \mod p$ *and therefore* $\deg(t, w) < p$*. For a term* t *the degree of* t*,* $\deg(t)$*, equals* $\sum_{w \in vars(t)} \deg(t, w)$*.*

4.3 Quadratic degree

The next definition is a generalization/modification of Sokolov's definition of Quadratic degree for the more general scenario where the proof contains extension variables that are *Singular* as well as ones that are *Nonsingular*.

▶ **Definition 15** (Quadratic degree)**.** *Let* V *be a set of variables and let* S *be a subset of* V*. For a pair of terms* t_1, t_2 *over* V*, and a variable* $w \in V$*, we define* $Qdeg^S(t_1, t_2, w)$ *as follows. If* $w \in S$*, then* $Qdeg^S(t_1, t_2, w) = 1$ *if* w *occurs in at least one of* t_1 *or* t_2*; if* $w \notin S$*, then* $Qdeg^S(t_1, t_2, w) = 1$ *if and only if* $\deg(t_1, w) \neq \deg(t_2, w)$*. The overall quadratic degree*

of the pair t_1, t_2, $Qdeg^S(t_1, t_2)$, *is equal to* $\sum_{w \in V} Qdeg^S(t_1, t_2, w)$. *The quadratic degree of a polynomial* P *is equal to the maximum quadratic degree over all pairs* (t_1, t_2) *such that* $t_1, t_2 \in P$. *For a proof* Π, *the quadratic degree of* Π *is the maximum quadratic degree over all polynomials* $P \in \Pi$.

We usually instantiate the above definition with $V = \mathbf{x} \cup \mathbf{y} \cup \mathbf{z}$ and with S being the set of Singular variables as defined by the extension axioms corresponding to $\mathbf{z}$. However since $Qdeg^S$ is a different measure for every S, and our set of Singular variables can change under the application of a restriction ρ to the variables in V, we must make sure that our measure of Quadratic degree does not change significantly under a restriction[2]. Fortunately, we can show that for any two sets S and T such that $T \subseteq S$, $Qdeg^T \leq Qdeg^S$. Along with the simple observation that the set of Singular extension variables can only shrink under a restriction, this implies that our measure of Quadratic degree can only decrease under a restriction. We make this formal below.

▶ **Lemma 16.** *Let* V *be a set of variables and let* S *and* T *be subsets of* V *such that* $T \subseteq S$. *Then for any two terms* t_1, t_2 *over* V, $Qdeg^T(t_1, t_2) \leq Qdeg^S(t_1, t_2)$.

Proof. Note that for a variable $w \in S - T$, $Qdeg^S(t_1, t_2, w) = 1$ when w has a nonzero exponent in one of t_1 or t_2, otherwise zero. However, $Qdeg^T(t_1, t_2, w) = 1$ if and only if the previous condition is satisfied and the exponents of w in t_1 and t_2 are not equal. Thus the claim follows. ◀

Henceforth, when we refer to Quadratic degree, we always fix the set S to be the set of Singular variables w.r.t. the underlying extension axioms. We have the following important corollary that this measure always decreases under a restriction to the underlying variables.

▶ **Corollary 17.** *Let* F *be a set of unsatisfiable polynomials over variables* $\mathbf{x} \cup \mathbf{y}$ *and let* Ext *be a set of extension axioms of the form* $z - Q(w_{i_1}, \ldots, w_{i_\kappa})$ *for variables* $z \in \mathbf{z}$ *and* $w_{i_1}, \ldots, w_{i_\kappa} \in \mathbf{x} \cup \mathbf{y}$. *Let* ρ *be a restriction to* $\mathbf{x} \cup \mathbf{y}$ *and let* $Ext|_\rho$ *be the axioms given by* $z - Q|_\rho$ *for each axiom* $z - Q \in Ext$. *The Quadratic degree w.r.t.* $Ext|_\rho$ *is at most the Quadratic degree w.r.t.* Ext.

Proof. Since $supp(Q|_\rho) \subseteq supp(Q)$ for any polynomial Q, we have that the set of Singular variables under $Ext|_\rho$ is a subset of those under Ext. Therefore our claim follows from the previous lemma. ◀

▶ **Lemma 18** (Quadratic degree upper bounds degree of Singular variables). *For any term* t, $\deg(sing(t)) \leq pQdeg(t, t)$

Proof. For any Singular variable w, $Qdeg(t, t, w) = 1$ if and only if w occurs in t. Since w can occur in t with degree at most $p - 1$, the claim follows. ◀

▶ **Definition 19** (High quadratic degree terms). *For a proof* Π *and* $d \geq 0$, *let* $\mathcal{H}_d(\Pi)$ *denote the set of unordered pairs* (t_1, t_2) *of quadratic degree at least* d. *That is,* $\mathcal{H}_d(\Pi)$ *is the set of unordered pairs of terms* (t_1, t_2) *such that* t_1, t_2 *both occur in* P *for some polynomial* $P \in \Pi$, *and* $Qdeg(t_1, t_2) \geq d$.

[2] If the set S does not change under a restriction, $Qdeg^S$ can still change under the restriction as terms can shrink or disappear when variables are set by the restriction. However, this is no different from how the usual notion of degree changes under a restriction, and it is trivial to show that $Qdeg^S$ always decreases. Therefore we ignore this for the sake of simplicity.

▶ **Lemma 20.** *Let* Π *be a* PC + Ext *refutation of* F *and let* z *be a Nonsingular variable. Let* Π' *be the proof obtained from* Π *by reducing each line of* Π *by* $z^{\ell(z)} - c = 0$ *for some* $c \in \mathbb{F}_p^*$. *Then* $|\mathcal{H}_d(\Pi')| \leq |\mathcal{H}_d(\Pi)|$ *for any* $d \geq 0$.

Proof. Consider a polynomial $P \in \Pi$ and a pair of terms (t_1, t_2) that occur in P. For any variable w distinct from z, $Qdeg(t_1, t_2, w)$ is unaltered when P is reduced by $z^{\ell(z)} = c$. On the other hand, if z does not contribute to the Quadratic degree of (t_1, t_2) i.e. $Qdeg(t_1, t_2, z) = 0$, then it will still be 0 after reducing by $z^{\ell(z)} = c$. Therefore $Qdeg(t_1, t_2)$ never increases for any pair (t_1, t_2) and thus $|\mathcal{H}_d(\Pi')| \leq |\mathcal{H}_d(\Pi)|$. ◀

The following is a generalized version of the argument from [20] that shows how to convert a proof with low Quadratic degree to one with low degree.

▶ **Lemma 21.** *Let* F *be a set of unsatisfiable polynomials of degree* d_0 *with a* PC *refutation of Quadratic degree at most* $d \geq d_0$ *over* $\mathbb{F}_p$. *Then* F *has a* PC *refutation of degree at most* $3pd$.

Proof. The proof of this lemma is largely based on (a slightly cleaner version of) Sokolov's argument ([20], Lemma 3.6) that low Quadratic degree over $\{\pm 1\}$ variables implies low degree. Our first observation is that Sokolov's argument can be applied to any refutation of low Quadratic degree over $\mathbb{F}_p$ such that every term contains only Nonsingular variables. In particular if $\{P_j\}$ is a refutation that only contains Nonsingular terms, then we can use his argument to show that $\{t_j^{p-2} P_j\}$ is also a valid refutation for some carefully chosen term $t_j \in P_j$. Moreover, the degree of the latter refutation is bounded by a constant times the Quadratic degree of the former one. To see this, first note that for two Nonsingular terms t_1 and t_2, we have that $deg(t_1 t_2^{p-2}) \leq (p-1) \cdot Qdeg(t_1, t_2)$, because of the following. For a variable z that is Nonsingular such that z occurs in t_1 and t_2 with $deg(t_1, z) = deg(t_2, z)$, we have $deg(t_1 t_2^{p-2}, z) = Qdeg(t_1, t_2, z) = 0$ since it would appear in $t_1 t_2^{p-2}$ with an exponent that is a multiple of $p-1$, and $z^{p-1} = 1$ holds for Nonsingular variables. Any other Nonsingular z that occurs in at least one of t_1 and t_2 has $deg(t_1 t_2^{p-2}, z) < p-1$ and $Qdeg(t_1, t_2, z) = 1$. Therefore the degree of $t_1 t_2^{p-2}$ is at most $p \cdot Qdeg(t_1, t_2)$ when t_1 and t_2 contain only Nonsingular variables. This implies that the lines in the new refutation $\{t_j^{p-2} P_j\}$ have degree at most p times the Quadratic degree of the original refutation $\{P_j\}$. Sokolov additionally showed that each line in the new refutation can be derived from previous lines without exceeding degree equal to $2p$ times the Quadratic degree of the original refutation, completing the argument.

In our case we deal with terms containing both Singular and Nonsingular variables. The above argument cannot be applied directly to our case, since it crucially depends on the fact that Nonsingular variables can be raised to the power $p-1$ to make them vanish. Fortunately by Lemma 18, the degree of Singular variables in any term is at most p times the Quadratic degree with itself. Given this bound, we can ignore for each term the part that contains Singular variables, and apply the above argument only with respect to the Nonsingular part of each term, to reduce the degree of Nonsingular variables in each term of the refutation. Since we now have a bound on the degree of both Singular and Nonsingular variables in each term, we have bounded its degree. We describe this in full technical detail below.

Let $\{P_j\}$ be a refutation of F with Quadratic degree bounded by d. For any term t recall that $nsing(t)$ denotes the subterm of t containing only Nonsingular variables. Note that $nsing(t)^{p-1} = 1$ for any t. For every line P_j in the refutation, we pick a term $t_j \in P_j$ and define $P'_j = nsing(t_j)^{p-2} P_j$. Note that by the arguments outlined above, for any two terms t_1 and t_2 in P_j, we have $\deg(nsing(t_1)^{p-2} nsing(t_2)) \leq pd$ and thus the degree of Nonsingular variables in any term of P'_j is bounded by pd. Since the Singular

variables in any term remain unchanged under multiplication by $nsing(t_j)^{p-2}$, the Singular degree of P'_j the same as that of P_j and is bounded by pd (Lemma 18) and therefore $\deg(P'_j) \leq pd + pd = 2pd$. We now show that the set $\{P'_j\}$ forms a valid refutation of F and each P'_j can be derived from previous lines in degree $3pd$. If P_j is one of the axioms, we multiply by $nsing(t_j)^{p-2}$ to get P'_j for an arbitrary $t_j \in P_j$, and this takes degree $pd_0 \leq pd$. If $P_j = wP_{j_1}$ for $j_1 < j$ and some variable w, we choose $t_j \in P_j$ such that $t_j = wt_{j_1}$ where $t_{j_1} \in P_{j_1}$ was chosen earlier. If w is Singular, we have $nsing(t_j) = nsing(t_{j_1})$ and therefore $P'_j = nsing(t_j)^{p-2}P_j = w \cdot nsing(t_{j_1})^{p-2}P_{j_1} = wP'_{j_1}$. On the other hand, if w is Nonsingular, we have $nsing(t_j) = w \cdot nsing(t_{j_1})$ and therefore $P'_j = nsing(t_j)^{p-2}P_j = w^{p-1} \cdot nsing(t_{j_1})^{p-2}P_{j_1} = P'_{j_1}$. Finally, let $P_j = P_{j_1} + P_{j_2}$ for $j_1, j_2 < j$. We pick an arbitrary term $t_j \in P_j$. Note that since $nsing(t)^{p-1} = 1$ for any term t, $P_{j_1} = nsing(t_{j_1})P'_{j_1}$ and $P_{j_2} = nsing(t_{j_2})P'_{j_2}$ and thus we have $P'_j = nsing(t_j)^{p-2}nsing(t_{j_1})P'_{j_1} + nsing(t_j)^{p-2}nsing(t_{j_2})P'_{j_2}$ for $t_{j_1} \in P_{j_1}$ and $t_{j_2} \in P_{j_2}$ chosen earlier. We now show that $\deg(nsing(t_j)^{p-2}nsing(t_{j_1})) \leq pd$ and $\deg(nsing(t_j)^{p-2}nsing(t_{j_2})) \leq pd$ to conclude the proof. Since every term in P_j appears in one of P_{j_1}, P_{j_2}, let $t_j \in P_{j_1}$ without loss of generality. Then we have that t_j, t_{j_1} both appear in P_{j_1} and thus $\deg(nsing(t_j)^{p-2}nsing(t_{j_1})) \leq pd$. If $t_{j_2} \in P_j$ i.e. it is not cancelled in the sum $P_{j_1} + P_{j_2}$, then we have t_j, t_{j_2} both appear in P_j and hence $\deg(nsing(t_j)^{p-2}nsing(t_{j_2})) \leq pd$. If $t_{j_2} \notin P_j$, this implies that it was cancelled in the sum $P_{j_1} + P_{j_2}$ and therefore $t_{j_2} \in P_{j_1}$ and $\deg(nsing(t_j)^{p-2}nsing(t_{j_2})) \leq pd$. ◀

4.4 The Split Operation

In this section we will show how to apply a restriction and then use an operation *Split* (motivated by [20]) in order to eliminate high quadratic degree terms. Our main focus will be to handle the case where the variable to be set is an extension variable with extension axiom $z - Q = 0$ where z is $Nonsingular$, since in the other case we can potentially just set $z = 0$ to eliminate terms. We start by showing how to apply a small Boolean restriction ρ such that $Q|_\rho$ is a simple linear function of just one variable.

▶ **Lemma 22.** *Let z be an extension variable with extension axiom $z - Q(w_1, \ldots, w_k)$, for $k \leq \kappa$. Assume that z is Nonsingular (i.e. $supp(Q) \subseteq [1, \ldots, p-1]$) and $|supp(Q)| \geq 2$. Then for every $l \in [0, \ldots, \ell(supp(Q)) - 1]$, there exists a variable w^* in $vars(Q)$, and a restriction δ to $vars(Q) - w^*$ such that:*

(1) *$Q|_\delta = (b-a)w^* + a$, where $b, a \in supp(Q)$. Thus $Q|_\delta$ is a linear function of w^* and $supp(Q|_\delta) = \{a, b\}$;*

(2) *$a^l \neq b^l \pmod p$*

Proof. We will create a decision tree that will query $vars(Q)$ one-by-one. Associated with the root r is the set of values $S_r = \{a^l \mid a \in supp(Q)\}$. That is, we label the root with the set of all possible values that z^l can take on. Since $l < \ell(supp(z))$, it follows that $|S_r| \geq 2$ (since otherwise we would have $l = \ell(supp(z))$). At the root we query the first variable w_1, with left edge labelled by $w_1 = 0$ and right edge labelled by $w_1 = 1$. Now we label the left vertex with the set $\{a^l \mid a \in supp(Q|_{w_1=0})\}$, of all values that z^l can take on under the restriction $w_1 = 0$. Similarly we label the right vertex with the set $\{a^l \mid a \in supp(Q|_{w_1=1})\}$. We continue recursively, querying the next variable at each vertex v of the decision tree, as long as the set of allowable values for z^l under the partial restriction ρ_v associated with v is greater than one. Now consider the longest path, ξ in T. The partial restriction ρ associated with ξ sets the first k' variables, where $k' \geq 1$ since initially z^l takes on at least two values. Also since ξ is a complete path, the associated set $\{a^l \mid a \in supp(Q|_\rho)\}$ contains exactly one element, call it q.

Now consider the twin path ξ' with associated restriction ρ', where ρ' is obtained from ρ by toggling the value of the last variable, $w_{k'}$, queried. Again since ξ' is a complete path, the associated set $\{a^l \mid a \in supp(Q|_{\rho'})\}$ contains exactly one element, call it q'. Note that q, q' must be distinct.

Let δ be the following assignment to $vars(Q) - w_{k'}$: for $1 \leq j < k'$, we set $\delta(w_j) = \rho(w_j) = \rho'(w_j)$, and for $k' < j \leq k$, we set $\delta(w_j) = 0$. Setting $w^* = w_{k'}$, $Q|_\delta$ is a linear equation of the form $(b-a)w^* + a$, where $b, a \in supp(Q)$. Finally, by construction, $a^l \neq b^l$ (since otherwise the two paths corresponding to ρ, ρ' would be the same). ◀

In the remainder of this subsection, we will be interested in the case where we want to eliminate some Nonsingular extension variable z from the refutation, and we have already applied the above Lemma so that the extension axiom for z is of the form $z-((b-a)w+a) = 0$, where w is some variable in $\mathbf{x} \cup \mathbf{y}$. Thus, $supp(z) = \{a, b\}$. The next two Lemmas generalizes a similar argument due to Sokolov, and show how to remove Quadratic degree pairs of the form $(t_1 z^i, t_2 z^j)$ for a carefully chosen pair i, j from the refutation via the Split operation.

▶ **Lemma 23.** *Let z be an extension variable such that $supp(z) = \{a, b\}$, where $a \neq b$ and $a, b \in \mathbb{F}_p^*$ and let P be any polynomial. Then, for any two distinct numbers ℓ_0, ℓ_1 where $\ell_0 < \ell_1$ and $a^{\ell_1 - \ell_0} \neq b^{\ell_1 - \ell_0}$, there exists a unique polynomial $R = R_0 z^{\ell_0} + R_1 z^{\ell_1}$ such that $R = P \mod (z-a)(z-b)$. That is, $R(a) = P(a)$ and $R(b) = P(b)$, where $P(a)$ denotes the polynomial P under the substitution $z = a$.*

Proof. Let $z - Q = 0$ be the extension axiom for z, where $supp(z) = \{a, b\}$. Then by Lemma 13 the polynomial $(z-a)(z-b) = 0$ is implied by (and derivable from) the extension axiom for z plus the Boolean axioms. We can assume without loss of generality that P has the form $P_0 + zP_1 + \ldots + z^{p-2}P_{p-2}$.

Now we want to argue that there exists a polynomial $R = z^{\ell_0}R_0 + z^{\ell_1}R_1$, where R_0, R_1 are polynomials over $vars(P) - z$, and such that $R(a) = P(a)$, and $R(b) = P(b)$. We can find R_0 and R_1 by solving the following system of equations, where we view R_0, R_1 as the underlying variables, and treating $P(a), P(b)$ as constants:

$$a^{\ell_0}R_0 + a^{\ell_1}R_1 = P(a)$$

$$b^{\ell_0}R_0 + b^{\ell_1}R_1 = P(b)$$

This has a (unique) solution since the determinant of the associated matrix is $\begin{vmatrix} a^{\ell_0} & a^{\ell_1} \\ b^{\ell_0} & b^{\ell_1} \end{vmatrix} = a^{\ell_0}b^{\ell_0}(b^{\ell_1-\ell_0} - a^{\ell_1-\ell_0})$. By our assumption, this matrix is non-singular over $\mathbb{F}_p$ and therefore the above system of equations has a unique solution over $\mathbb{F}_p$, given by:

$$\begin{pmatrix} R_0 \\ R_1 \end{pmatrix} = \begin{pmatrix} a^{\ell_0} & a^{\ell_1} \\ b^{\ell_0} & b^{\ell_1} \end{pmatrix}^{-1} \begin{pmatrix} P(a) \\ P(b) \end{pmatrix}$$

Abbreviating $a^{\ell_0}, a^{\ell_1}, b^{\ell_0}, b^{\ell_1}$ by a_0, a_1, b_0, b_1 respectively, we have by definition of the inverse:

$$\begin{aligned} \begin{pmatrix} R_0 \\ R_1 \end{pmatrix} &= \begin{pmatrix} a_0 & a_1 \\ b_0 & b_1 \end{pmatrix}^{-1} \begin{pmatrix} P(a) \\ P(b) \end{pmatrix} \\ &= \frac{1}{a_0b_1 - a_1b_0} \begin{pmatrix} b_1 & -a_1 \\ -b_0 & a_0 \end{pmatrix} \begin{pmatrix} P(a) \\ P(b) \end{pmatrix} \end{aligned}$$

Solving for R_0 we have:

$$\begin{aligned}
R_0 &= \frac{b_1}{a_0b_1 - a_1b_0}P(a) - \frac{a_1}{a_0b_1 - a_1b_0}P(b) \\
&= \frac{b_1}{a_0b_1 - a_1b_0}(a_0P_{\ell_0} + a_1P_{\ell_1} + \sum_{i\neq\ell_0,\ell_1} a^iP_i) - \frac{a_1}{a_0b_1 - a_1b_0}(b_0P_{\ell_0} + b_1P_{\ell_1} + \sum_{i\neq\ell_0,\ell_1} b^iP_i) \\
&= \frac{a_0b_1}{a_0b_1 - a_1b_0}P_{\ell_0} + \frac{a_1b_1}{a_0b_1 - a_1b_0}P_{\ell_1} + \frac{b_1}{a_0b_1 - a_1b_0}(\sum_{i\neq\ell_0,\ell_1} a^iP_i) \\
&\qquad - \frac{a_1b_0}{a_0b_1 - a_1b_0}P_{\ell_0} - \frac{a_1b_1}{a_0b_1 - a_1b_0}P_{\ell_1} - \frac{b_1}{a_0b_1 - a_1b_0}(\sum_{i\neq\ell_0,\ell_1} b^iP_i) \\
&= P_{\ell_0} + \sum_{i\neq\ell_0,\ell_1} c_{0i}P_i
\end{aligned}$$

for some constants $c_{0i} \in \mathcal{F}_p$. And similarly solving for R_1, it has the following form:

$$R_1 = P_{\ell_1} + \sum_{i\neq\ell_0,\ell_1} c_{1i}P_i$$

for some constants $c_{1i} \in \mathcal{F}_p$. ◀

▶ **Definition 24** (*Split*). *Let z be an extension variable with extension axiom $z - Q = 0$ such that $supp(z) = \{a, b\} \subseteq [1, \ldots, p-1]$. For any polynomial P and for every $\ell_0 < \ell_1$ such that $a^{\ell_1-\ell_0} \neq b^{\ell_1-\ell_0}$, let $R = R_0z^{\ell_0} + R_1z^{\ell_1}$ be the unique polynomial given by Lemma 23 such that $R = P$ mod $(z-a)(z-b)$. Then $Split_{z,\ell_1,\ell_0}(P)$ is defined to be the pair of polynomials $\{R_0, R_1\}$. For a proof Π, and an extension variable z such that $supp(z) = \{a, b\}$, we define $Split_{z,\ell_0,\ell_1}(\Pi)$ to be the sequence of lines $Split_{z,\ell_0,\ell_1}(P)$, over all $P \in \Pi$.*

▶ **Lemma 25.** *Let Π be a refutation of a set of unsatisfiable polynomials F. Let z be a variable that occurs in Π such that the polynomials in F do not contain z except for the axiom $(z-a)(z-b) = 0$ for some $a, b \in \mathbb{F}_p^*$. Then for any ℓ_0, ℓ_1 such that $\ell_0 < \ell_1$ and $a^{\ell_1-\ell_0} \neq b^{\ell_1-\ell_0}$, $\Pi' = Split_{z,\ell_0,\ell_1}(\Pi)$ forms a valid refutation of F modulo $(z-a)(z-b)$*

Proof. Fix an extension variable z in Π such that it does not occur in any axioms except $(z-a)(z-b) = 0$, and let ℓ_0, ℓ_1 be such that $\ell_0 < \ell_1$ and $a^{\ell_1-\ell_0} \neq b^{\ell_1-\ell_0}$. We will show by induction on the number of lines in Π that $Split_{z,\ell_0,\ell_1}(\Pi)$ is a valid derivation that meets the conditions of the lemma. For the base case, note that all of the axioms are either free of z or eliminated as a result of reducing by $(z-a)(z-b)$, and hence their Split versions are derivable. Now suppose that the Lemma holds for the first $j-1$ lines of Π; that is, $Split_{z,\ell_0,\ell_1}(\Pi_{j-1})$ is a valid derivation, where Π_{j-1} denotes the first $j-1$ lines of Π.

The first case is where P_j is a linear combination of two previously derived lines, so $P_j = \alpha P_{j_1} + \beta P_{j_2}$ for some j_1 and j_2 less than j and $\alpha, \beta \in \mathbb{F}_p$. Using the inductive hypothesis, we have:

$$\begin{aligned}
P_j &= \alpha(z^{\ell_0}R_{j_10} + z^{\ell_1}R_{j_11}) + \beta(z^{\ell_0}R_{j_20} + z^{\ell_1}R_{j_21}) \mod (z-a)(z-b) \\
&= z^{\ell_0}(\alpha R_{j_10} + \beta R_{j_20}) + z^{\ell_1}(\alpha R_{j_11} + \beta R_{j_21}) \mod (z-a)(z-b)
\end{aligned}$$

By the uniqueness of the polynomial $R_j = z^{\ell_0}R_{j0} + z^{\ell_1}R_{j1}$ that is equivalent to P_j mod $(z-a)(z-b)$ (by Lemma 23), this implies that $R_{j0} = \alpha R_{j_10} + \beta R_{j_20}$ and similarly $R_{j1} = \alpha R_{j_11} + \beta R_{j_21}$, and thus R_{j0} can be derived from a linear combination of R_{j_10} and R_{j_20} and similarly for R_{j1}.

The second case is when P_j is derived from a previously derived line $P_{j'}$ by multiplying $P_{j'}$ by a variable w. That is, $P_j = wP_{j'}$ for some $j' < j$. If $w \neq z$, then we have that $R_{j1} = wR_{j'1}$ (similarly for R_{j0}). If $w = z$ then we have:

$$\begin{pmatrix} R_{j'1} \\ R_{j'0} \end{pmatrix} = \begin{pmatrix} a^{\ell_1} & a^{\ell_0} \\ b^{\ell_1} & b^{\ell_0} \end{pmatrix}^{-1} \begin{pmatrix} P_{j'}(a) \\ P_{j'}(b) \end{pmatrix}$$

from which we need to derive

$$\begin{aligned} \begin{pmatrix} R_{j1} \\ R_{j0} \end{pmatrix} &= \begin{pmatrix} a^{\ell_1} & a^{\ell_0} \\ b^{\ell_1} & b^{\ell_0} \end{pmatrix}^{-1} \begin{pmatrix} P_j(a) \\ P_j(b) \end{pmatrix} \\ &= \begin{pmatrix} a^{\ell_1} & a^{\ell_0} \\ b^{\ell_1} & b^{\ell_0} \end{pmatrix}^{-1} \begin{pmatrix} aP_{j'}(a) \\ bP_{j'}(b) \end{pmatrix} \\ &= \begin{pmatrix} a^{\ell_1} & a^{\ell_0} \\ b^{\ell_1} & b^{\ell_0} \end{pmatrix}^{-1} \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix} \begin{pmatrix} a^{\ell_1} & a^{\ell_0} \\ b^{\ell_1} & b^{\ell_0} \end{pmatrix} \begin{pmatrix} R_{j'1} \\ R_{j'0} \end{pmatrix}. \end{aligned}$$

Thus, R_{j1} can be derived as a linear combination of $R_{j'1}$ and $R_{j'0}$, and similarly for R_{j0}. ◀

4.5 Proof of Main Theorem

The proof of our lower bound for the tautology $F^{SEL}_{n,k}$ with extension axioms Ext proceeds by choosing a variable in the given refutation Π that contributes to a lot of high quadratic degree pairs of terms in Π. If this variable is Singular, we apply the restriction that sets it to zero. On the other hand, if it is Nonsingular and therefore an extension variable z, we first reduce it to depend on a single variable w^* by applying a restriction chosen from Lemma 22, and then use a more complicated case analysis (see Lemma 30) in order to apply the Split operation from Lemmas 23 and 25 on z. In both of these cases we are able to remove a small fraction of high Quadratic degree terms, and thus after sufficiently many iterations we obtain a refutation of low Quadratic degree. We convert this to a refutation of low (usual) degree using Lemma 21, and then substitute for the pigeon variables $\mathbf{y}$ to select a subset of equations from $F_{n,k}$ that require high degree, obtaining a contradiction.

4.5.1 Cleanup operations

In order to get the contradiction at the end of the above argument, we need to ensure that our process above is always working with a subset of equations of $F_{n,k}$ that are untouched, i.e. unaffected by earlier restrictions to variables. We also need to eliminate any partially assigned pigeons so that we have full choice over the equations we are able to pick at the end. Additionally, a key requirement of the Split lemmas (Lemmas 23 and 25) is that the variable z we Split on must not appear in any axioms except for one of the form $(z - a)(z - b) = 0$, which indicates that it takes two distinct values. In particular, we cannot set z or the underlying variable w^* described above in order to eliminate them from the refutation. This presents us with a unique requirement: for any choice of a variable $w^* \in \mathbf{x} \cup \mathbf{y}$, we need to be able to eliminate all axioms containing w^* without actually setting it. We show how to perform these operations by making use of the structure of our tautology $F^{SEL}_{n,k}$.

We first show how to "ban" an equation $E_{b_1 \dots b_{\log m}}$ from $F_{n,k}$ by switching to a set of axioms that prevent any pigeon from being assigned to $b_1 \dots b_{\log m}$.

▶ **Lemma 26.** *Let Π be a refutation of $F^{SEL}_{n,k}|_\rho$ for some restriction ρ and let $(y_i \neq b_1 \dots b_{\log m}) \cdot E_{b_1 \dots b_{\log m}} = 0$ be one of its axioms. Then there exists another valid refutation Π' with the latter axiom replaced by the axiom $(y_i \neq b_1 \dots b_{\log m}) \equiv \prod_{b_j=1} y_{i,j} \prod_{b_j=0} \overline{y_{i,j}}$, such that the quadratic degree of Π' is at most that of Π.*

Proof. Note that the axiom $(y_i \neq b_1 \dots b_{\log m}) \cdot E_{b_1 \dots b_{\log m}} = 0$ can be derived from the axiom $(y_i \neq b_1 \dots b_{\log m}) \equiv \prod_{b_j=1} y_{i,j} \prod_{b_j=0} \overline{y_{i,j}}$ by multiplying by the polynomial $E_{b_1 \dots b_{\log m}}$. Since this derivation involves only singular variables, the degree can never drop and therefore the quadratic degree of this derivation is at most that of the final polynomial. We construct Π' as follows. We first derive the former axiom from the latter in Π'. Besides this derivation, Π' involves the same steps as Π. ◀

▶ **Definition 27.** *An equation $E_{b_1 \dots b_{\log m}}$ is said to be banned when the previous lemma is applied repeatedly to eliminate all occurrences of it from the axioms.*

▶ **Definition 28.** *A clean version of $F_{n,k}^{SEL}$ is any subset of axioms of $F_{n,k}^{SEL}$ along with axioms that ban some subset of equations of the form $E_{b_1 \dots b_{\log m}}$.*

4.5.1.1 Cleanup(ρ)

We now describe how to perform the cleanup operations, which we collectively call **Cleanup**(ρ), that takes as input an "unclean" version of $F_{n,k}^{SEL}$ derived by applying a restriction ρ to a clean version, and outputs another clean version that is in some sense a subset of the input. Suppose that we are given a restriction ρ that has been applied to a clean version of $F_{n,k}^{SEL}$, with a variable $w^* \in \rho$ possibly set to $\star$, indicating that it must remain unset. To eliminate an axiom that has been affected by a $\mathbf{x}$ variable in ρ not set to $\star$, we simply obtain the refutation that bans the corresponding equation $E_{b_1 \dots b_{\log m}}$ as described in the above lemma. Note that since we are eliminating the axiom without setting any variables in it, we can also do this in case our variable $w^* \in \mathbf{x}$. Suppose that y_{ij} is a $\mathbf{y}$ variable in ρ not set to $\star$. We first note that any axiom that contained y_{ij} before the application of ρ contains all the variables $y_{i1} \dots y_{i \log m}$ corresponding to the i^{th} pigeon y_i. We first make sure that this i^{th} pigeon does not contain our variable w^* that must remain unset. If it doesn't, we proceed as follows. We set all the other variables in this pigeon to select some equation $E_{b_1 \dots b_{\log m}}$ that has not been banned. Such an equation exists provided that the number of banned equations so far is bounded, and the size of the restriction ρ is also bounded (we formalize this in the lemma below). We then apply an additional restriction to the $\mathbf{x}$ variables that satisfies this equation $E_{b_1 \dots b_{\log m}}$ picked above. We then ban all the equations affected by this additional restriction, like we did above for the part of ρ containing $\mathbf{x}$ variables. This eliminates the pigeon y_i. We are left with the case where our variable w^* belongs to some pigeon y_j. We set all the variables in the pigeon y_j except for w^*, such that neither of the two equations $E_{b_1 \dots b_{\log m}}$ and $E_{b'_1 \dots b'_{\log m}}$ that would be selected if w^* is set to zero or one are banned (again, these exist under the same conditions as above). We then proceed as before, i.e. apply an additional restriction to satisfy both these equations, and then ban any other equations that have been affected by this additional restriction. With this we have eliminated the axioms of pigeon y_j which select an equation, but we are still left with the axioms that prevent y_j from colliding with any other pigeon, which are now of the form $w^* \cdot (y_{j'} \neq b_1 \dots b_{\log m})$ and $\overline{w^*} \cdot (y_{j'} \neq b'_1 \dots b'_{\log m})$ indicating that any pigeon $y_{j'}$ distinct from y_j must not be mapped to the equations $E_{b_1 \dots b_{\log m}}$ and $E_{b'_1 \dots b'_{\log m}}$ if one of them is selected by setting w^* to zero or one. To remove the latter axioms we do something similar to the process of banning an equation, where we simply replace these axioms by the axioms $(y_{j'} \neq b_1 \dots b_{\log m})$ and $(y_{j'} \neq b'_1 \dots b'_{\log m})$, effectively banning the equations $E_{b_1 \dots b_{\log m}}$ and $E_{b'_1 \dots b'_{\log m}}$ for the remaining pigeons.

4.5.1.2 Correctness of Cleanup(ρ)

We note that the above cleanup operations over $\mathbf{y}$ variables terminate successfully only when there are enough equations that have not been banned by prior calls to cleanup, and also the size of the restriction ρ is bounded. We make this formal by the below lemma.

▶ **Lemma 29** (Correctness of **Cleanup(ρ)**). *Let ρ be a restriction of size κ. If the number of banned equations (from previous calls to **Cleanup**) is $\ll m/2^{\kappa}$, then **Cleanup(ρ)** terminates correctly. Moreover, it bans at most $O(\kappa)$ additional equations and removes at most $O(\kappa)$ pigeons in its run.*

Proof. In **Cleanup(ρ)**, note that we can remove the axioms that contain $\mathbf{x}$ variables unconditionally. When we remove a pigeon $y_i = y_{i1} \dots y_{i \log m}$, we rely on having an equation it can be set to that is not already banned. Since the size of ρ is bounded by κ, note that at most κ variables from $y_{i1} \dots y_{i \log m}$ can be set by ρ. Therefore there are at least $\log m - \kappa$ of them unset, corresponding to selecting $m/2^{\kappa}$ many equations. Since we assume that the number of banned equations is much less than this, we can always find one that is not banned to assign this pigeon to.

We now count the number of new equations banned and the number of pigeons removed by this call to **Cleanup(ρ)**. Since each $\mathbf{x}$ variable appears in a constant number of equations, the number of equations we ban while processing it is a constant. When we process a $\mathbf{y}$ variable, we pick and satisfy an equation, and ban all other equations affected in the process. Since every equation also contains a constant number of variables, satisfying it affects only a constant number of other equations. Therefore, for every variable we process we ban only a constant number of equations, and thus the total number of equations banned is $O(\kappa)$. We remove only those pigeons with a variable in ρ, so this is also bounded by $O(\kappa)$. ◀

4.5.2 The Main Theorem

We need first the following key lemma that shows how to apply the Split operation to reduce high quadratic degree terms.

Algorithm 1 Algorithm for Lemma 30.

Input: A refutation Π, and a nonsingular variable z with extension axiom $z - Q = 0$ satisfying the pre-conditions of Lemma 30

Output: A refutation Π' satisfying post-conditions of Lemma 30

Let $\ell_0 < \ell_1$ be such that $|\mathcal{H}_d(\Pi, z, \ell_0, \ell_1)| \geq |\mathcal{H}_d(\Pi, z)|/p^2$.
Apply Lemma 22 with $l = \ell_1 - \ell_0$ to obtain δ, w^*, a, b satisfying post-conditions of Lemma 22.
$\Pi = \Pi|_\delta$ (and in particular $z - Q|_\delta = z - (b-a)w^* - a$)
Cleanup($\delta \cup \{w^* = \star\}$) (Cleanup axioms affected by δ and remove w^* from all axioms other than $z - (b-a)w^* - a$ while keeping it alive.)
if *w^* contributes to $\geq \epsilon/4p^2$ fraction of pairs in $\mathcal{H}_d(\Pi)$* **then**
    $\Pi = \Pi|_{w^*=0}$
end
else
    Apply the substitution $(z-a)/(b-a)$ for w^* in Π
    Let $\Pi' = Split_{z,l_0,l_1}(\Pi)$
end

▶ **Lemma 30.** *Let F be a system of unsatisfiable polynomials and let z be a nonsingular extension variable with the extension axiom $z - Q$. Let $\ell = \ell(supp(Q))$ so that $z^\ell = c$ holds for some $c \in \mathbb{F}_p$. Let Π be a refutation of $F \cup \{z - Q\}$ modulo $z^\ell = c$ such that for at least an ϵ fraction of pairs (t_1, t_2) in $\mathcal{H}_d(\Pi)$, $Qdeg(t_1, t_2, z) = 1$, for some $d \geq 0$. Then there exists a refutation Π' of F such that $|\mathcal{H}_d(\Pi')| \leq (1 - \epsilon/4p^2)|\mathcal{H}_d(\Pi)|$*

Proof. We will apply a procedure as described by Algorithm 1 in order to modify the proof to satisfy the post-conditions of the Lemma. Here we give a detailed description of the algorithm, together with its correctness. Let $\mathcal{H}_d(\Pi, z)$ be the set of all unordered pairs $(t_1, t_2) \in \mathcal{H}_d(\Pi)$ that z contributes to. That is, $\mathcal{H}_d(\Pi, z)$ is the set of all unordered pairs $(t_1, t_2) \in \mathcal{H}_d(\Pi)$ such that $Qdeg(t_1, t_2, z) = 1$. There are many different ways that z can contribute to $\mathcal{H}_d(\Pi, z)$: namely, for all i, j such that $i < j < \ell$, let $\mathcal{H}_d(\Pi, z, i, j)$ be the set of all unordered pairs $(t_1, t_2) \in \mathcal{H}_d(\Pi, z)$, such that the degree of z in t_1 is i and the degree of z in t_2 is j. Note that for any two pairs (i, j) and (i', j') such that $i \neq i'$ or $j \neq j'$, $\mathcal{H}_d(\Pi, z, i, j)$ and $\mathcal{H}_d(\Pi, z, i', j')$ are disjoint. Therefore, there exists a "good" pair $\ell_0 < \ell_1 < \ell$ such that removing $\mathcal{H}_d(\Pi, z, \ell_1, \ell_0)$ from $\mathcal{H}_d(\Pi, z)$ will remove at least a $1/p^2$ fraction of $\mathcal{H}_d(\Pi, z)$ and therefore a ϵ/p^2 fraction of pairs in $\mathcal{H}_d(\Pi)$, since $|\mathcal{H}_d(\Pi, z)| \geq \epsilon|\mathcal{H}_d(\Pi)|$.

We want to apply the Split operation $Split_{z,\ell_0,\ell_1}$ to remove all such pairs. But in order to do this we have to satisfy the preconditions of Lemmas 23 and 25: we need two values a, b such that $a^{\ell_1-\ell_0} \neq b^{\ell_1-\ell_0}$ and all the axioms should be free of z except for $(z - a)(z - b) = 0$. The first step (Line 2 of 1) is to apply Lemma 22 with $l = \ell_1 - \ell_0$. This gives us $w^* \in vars(Q)$, $a, b \in supp(Q)$ and a partial restriction δ to $vars(Q) - w^*$ such that $(z - Q)|_\delta = z - (b - a)w^* - a$, where $a^{\ell_1-\ell_0} \neq b^{\ell_1-\ell_0} \mod p$. Next, we apply the restriction δ to Π (Line 3).

Now we have a simpler *linear* extension axiom for z of the form $z - (b - a)w^* - a = 0$. Next we would like to make the substitution $w^* = (z - a)/(b - a)$ in Π in order to satisfy this extension axiom, towards the goal of eliminating z from the axioms so that we have the preconditions of Lemma 25 and therefore are able to apply $Split_{z,\ell_1,\ell_0}$. However, if w^* appears in any of the axioms in F, this would create additional occurrences of z and we would not make any progress. Therefore, we have to make sure that none of the axioms of F contain w^*. But we also cannot set w^* to zero or one in an attempt to get rid of it, since this would set z to either a or b through the above extension axiom, and Split requires that z take on two distinct values. We thus have to get rid of all axioms mentioning w^* either by setting other variables or by replacing these axioms with stronger versions, such that the former can be derived from the latter. This is what the subroutine **Cleanup** does, in addition to removing the axioms in F that were affected by our earlier restriction δ, so that we have a clean version of $F^{SEL}_{n,k}$ as defined in the previous section.

We are now ready to make the substitution $w^* = (z - a)/(b - a)$. Under this substitution, the Boolean axiom $w^{*2} - w = 0$ reduces to $(z - a)(z - b) = 0$, and the original extension axiom for z disappears (since under this substitution it becomes $0 = 0$.) Thus this substitution would satisfy all of the preconditions of Lemmas 23, 25. However, this substitution can create a new problem: it can cause a blow up in the size of $\mathcal{H}_d(\Pi)$ since for every pair of terms (t_1, t_2) such that one of them contains w^*, we could have up to four new terms after the substitution. In order to deal with this potential blow up we do a simple case analysis: If w^* contributes to at least an $\epsilon/4p^2$ fraction of pairs (t_1, t_2) in $\mathcal{H}_d(\Pi)$, then we set $w^* = 0$ (Lines 4-5). This gives us the required reduction in the size of $\mathcal{H}_d(\Pi)$ (z is also set to a constant by setting $w^* = 0$, but we don't care about that since we have obtained a reduction in high Quadratic degree terms without needing to use Split). Otherwise, the blowup caused by the substitution $w^* = (z - a)/(b - a)$ adds at most $3\epsilon/4p^2$ fraction of pairs to $\mathcal{H}_d(\Pi)$, and

thus if we remove all pairs in $\mathcal{H}_d(\Pi, z, \ell_0, \ell_1)$ (after this blowup) then overall we will will have reduced the size of $\mathcal{H}_d(\Pi)$ to $(1 - \epsilon/4p^2)|\mathcal{H}_d(\Pi)|$. So in this latter case, we apply the substitution mentioned above (Line 8) which simultaneously removes w^* from all axioms, and replaces the linear axiom for z by $(z-a)(z-b) = 0$. Now all preconditions for Lemma 8 hold so we can apply $Split_{z,\ell_0,\ell_1}$ (Line 9) to get a valid refutation. It is left to argue that this indeed removes the set $\mathcal{H}_d(\Pi, z, \ell_1, \ell_0)$. More precisely, we argue that high Quadratic degree pairs of terms in the refutation obtained after applying Split have a one to one mapping to the set $\mathcal{H}_d(\Pi) - \mathcal{H}_d(\Pi, z, \ell_1, \ell_0)$. Fix a line $P \in \Pi$. Since we are working modulo $z^\ell = c$, we can assume that $P = P_0 + zP_1 + \ldots z^{\ell-1}P_{\ell-1}$. Let $R = z^{\ell_0}R_0 + z^{\ell_1}R_1$ be the unique polynomial equivalent to P mod $(z-a)(z-b)$. $Split_{z,\ell_0,\ell_1}(\Pi)$ is the refutation with lines R_1, R_0 for all $P \in \Pi$. By the proof of Lemma 23 R_0, R_1 have the form:

$$R_1 = P_{\ell_1} + \sum_{i<\ell, i\neq\ell_0} c_{1i}P_i$$

$$R_0 = P_{\ell_0} + \sum_{i<\ell, i\neq\ell_1} c_{0i}P_i$$

for some constants $c_{1i}, c_{0i} \in \mathbb{F}_p$.

For a pair of terms (t_i, t_j) in R_1 such that $t_i \in P_i$ and $t_j \in P_j$ and $Qdeg(t_i, t_j) \geq d$, we map it to the pair $(t_i z^i, t_j z^j) \in P$, and similarly for R_0. Clearly this is a one-one mapping, and since P_{ℓ_0} does not occur in R_1 and P_{ℓ_1} does not occur in R_0, it is a mapping to $\mathcal{H}_d(\Pi) - \mathcal{H}_d(\Pi, z, \ell_1, \ell_0)$. Therefore we have that for the refutation $\Pi' = Split_{z,\ell_0,\ell_1}(\Pi)$ whose lines are $\{R_1, R_0\}$, $|\mathcal{H}_d(\Pi')| \leq |\mathcal{H}_d(\Pi) - \mathcal{H}_d(\Pi, z, \ell_1, \ell_0)| \leq (1 - \epsilon/4p^2)|\mathcal{H}_d(\Pi)|$. ◀

▶ **Theorem 31.** *For n sufficiently large, any (M, κ)-PC + Ext refutation of $F^{SEL}_{n,k}$ has size* $\exp\left(\frac{\Omega(n^2)}{10^\kappa(M + n\log n)}\right)$.

Proof. Let Π be an alleged (M, κ)-PC + Ext refutation of $F^{SEL}_{n,k}$ with logarithm of its size less than $\gamma n^2/(10^\kappa(M + n\log n))$, for a small enough constant γ. Given Π, Algorithm 2 (defined below) will apply a sequence of restrictions and cleanup steps in order to produce a refutation Π' of a clean version of $F^{SEL}_{n,k}$ (see Definition 29) with the property that the Quadratic degree of Π' is at most $d = \nu n/\kappa$ for a small enough constant $\nu > 0$. The algorithm contains a while loop which iteratively removes all pairs of terms of high Quadratic degree. From Π', we will apply a further restriction to all of the remaining unset $\mathbf{y}$-variables (i.e. pigeons that select equations from $F_{n,k}$), to extract a refutation of a subset of m' equations from $F_{n,k}$ of low degree, contradicting the degree lower bound given in Lemma 35. Recall that $F_{n,k}$ is defined over variables $\mathbf{x}$ and we pick a subset of these equations by matching pigeons y_i to equations in $F_{n,k}$ through a complete bipartite graph.

The algorithm first initializes a few things. Set $d = \nu n/\kappa$ for a small enough constant $\nu > 0$. Let $M' = M + n\log n$, which upper bounds the total number of variables occurring in the refutation. Let S be the set of all variables that are Singular w.r.t. the current set of extension axioms. We initialize S to be the set of all variables $\mathbf{x} \cup \mathbf{y} \cup \mathbf{z}$ since this is the largest possible set we will be dealing with; this will be updated at every iteration of the while loop, although we note that it can only reduce as we apply restrictions. Henceforth when we refer to Quadratic degree, we mean $Qdeg^S$. Finally, we initialize H to be the set of all pairs of terms in Π with Quadratic degree greater than d.

In the while loop, we first update the set S by checking which of the extension variables z have zero in their support according to their current extension axioms, and deleting those that don't. For each extension variable z that we delete from S, we reduce the refutation Π

Algorithm 2 Eliminating high Quadratic degree terms from the proof.

Input: A refutation Π of $F_{n,k}^{SEL}$ with extension axioms Ext
Output: A refutation Π' with Quadratic degree less than d

1 $d \leftarrow \nu n/\kappa$, where ν is a sufficiently small constant.
2 $M' \leftarrow M + n\log(n)$. ($M'$ upper bounds $|\mathbf{x} \cup \mathbf{y} \cup \mathbf{z}|$, the total number of variables)
3 $S \leftarrow \mathbf{x} \cup \mathbf{y} \cup \mathbf{z}$ (the current set of singular variables: all Boolean variables are singular by default and we initialize all extension variables to also be singular. This could possibly reduce in each iteration.)
4 $H \leftarrow \{(t_1, t_2) \mid t_1, t_2 \in \Pi \text{ and } Qdeg^S(t_1, t_2) \geq d\}$ (the set of all pairs of terms of large Quadratic degree according to S)
5 **while** *H is non empty* **do**
6 **for** *every extension axiom $z - Q \in Ext$* **do**
7 **if** $0 \notin supp(Q)$ **then**
8 $S \leftarrow S - \{z\}$
9 Compute c such that $z^{\ell(z)} = c$ and reduce Π by the latter identity
10 **end**
11 **end**
12 $H \leftarrow \{(t_1, t_2) \mid t_1, t_2 \in \Pi \text{ and } Qdeg^S(t_1, t_2) \geq d\}$ (update H to reflect changes due to the above **for** loop)
13 Pick a variable w that, by an averaging argument, occurs in at least an ϵ fraction of terms in H, where we choose $\epsilon = d/M'$.
14 **if** $w \in S$ **then**
15 Let σ be a restriction on $\mathbf{x} \cup \mathbf{y}$ such that $w|_\sigma = 0$
16 $\Pi \leftarrow \Pi|_\sigma$
17 **Cleanup**(σ)
18 **end**
19 **else**
20 Apply Algorithm 1, which by Lemma 30 satisfies the post-conditions of Lemma 30
21 **end**
22 **end**

by $z^{\ell(z)} = c$. Such an identity exists and is derivable by Lemma 13, and does not increase the size of H by Lemma 20. Once we have updated S, we recompute the set of high Quadratic degree pairs H with respect to the updated set S. This also does not increase the size of H, by Lemma 16. We then pick a variable w that contributes to the Quadratic degree of at least a d/M' fraction of pairs in H by averaging.

There are two cases depending on whether $w \in S$ or not. In the first case (lines 14-18), w is Singular so we apply the restriction σ such that $w|_\sigma = 0$ and call **Cleanup**(σ) to restore to a clean version of our tautology. This eliminates the contribution to high Quadratic degree from terms containing w, and hence obtains a $(1 - d/M')$-factor reduction in the size of H. In the second case (lines 19-34), w is Nonsingular so we apply Algorithm 1, which uses the Split operation non-trivially to reduce the size of H. Lemma 30 proves correctness of the algorithm, and thus upon termination of one call to Algorithm 1, we have obtained a $(1 - d/(4p^2M'))$-factor reduction in the number of high Quadratic degree terms.

Repeating the above for $-\log|H|/\log(1 - d/4p^2M') \approx 4p^2M'\log|H|/d \leq O(\gamma)\kappa n/10^\kappa$ iterations, we eliminate all terms in H from the proof and thus obtain a refutation of Quadratic degree less than d. Since we call **Cleanup** once per iteration, and in each call it

bans at most $O(\kappa)$ many equations and removes at most $O(\kappa)$ many pigeons (by Lemma 29), we have banned at most $O(\gamma)\kappa^2 n/10^\kappa$ equations and removed at most those many pigeons in total. Therefore, we always satisfy the invariant that the number of banned equations is much less than $m/2^\kappa$ (where $m = 10n$), satisfying the required conditions for correctness of **Cleanup** from Lemma 29.

Let Π' denote the modified proof upon termination of Algorithm 2. Note that out of the $m' = (1-\epsilon)m$ pigeons, there are at least a $1 - O(\gamma)$ fraction of pigeons still alive (i.e. not removed by **Cleanup**) and a $1 - O(\gamma)$ fraction of the m equations not banned. We now substitute for the remaining pigeons $\mathbf{y}$ so that we select a subset of at least $(1-2\epsilon)m$ unsatisfiable equations from $F_{n,k}$ that are not banned and obtain a refutation of them of Quadratic degree at most d (assuming γ is small enough). By Lemma 21, we can obtain a refutation of these equations of degree at most $3pd$. Now, for all surviving extension variables we substitute them with their definitions in terms of the variables $\mathbf{x}$. Note that since each extension variable is a degree κ polynomial this raises the degree to at most $3\kappa pd$. Since $d = \nu n/\kappa$, for sufficiently small ν we end up with a refutation of $(1-2\epsilon)$ equations from $F_{n,k}$ of degree less than $c_2 n$, contradicting Lemma 35. ◀

References

1. Michael Alekhnovich and Alexander A. Razborov. Lower bounds for polynomial calculus: Non-binomial case. In *42nd Annual Symposium on Foundations of Computer Science, FOCS 2001, 14-17 October 2001, Las Vegas, Nevada, USA*, pages 190–199. IEEE Computer Society, 2001. doi:10.1109/SFCS.2001.959893.
2. Yaroslav Alekseev. A lower bound for polynomial calculus with extension rule. In Valentine Kabanets, editor, *36th Computational Complexity Conference, CCC 2021, July 20-23, 2021, Toronto, Ontario, Canada (Virtual Conference)*, volume 200 of *LIPIcs*, pages 21:1–21:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.CCC.2021.21.
3. Robert Andrews and Michael A. Forbes. Ideals, determinants, and straightening: Proving and using lower bounds for polynomial ideals. *CoRR*, abs/2112.00792, 2021. arXiv:2112.00792.
4. Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on hilbert's nullstellensatz and propositional proofs. *Proceedings of the London Mathematical Society*, 3(1):1–26, 1996.
5. Paul Beame, Richard Karp, Toniann Pitassi, and Michael Saks. The efficiency of resolution and davis–putnam procedures. *SIAM Journal on Computing*, 31(4):1048–1075, 2002.
6. Eli Ben-Sasson and Russell Impagliazzo. Random cnfs are hard for the polynomial calculus. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 415–421. IEEE, 1999.
7. Eli Ben-Sasson and Avi Wigderson. Short proofs are narrow – Resolution made simple. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 517–526, 1999.
8. Sam Buss, Dima Grigoriev, Russell Impagliazzo, and Toniann Pitassi. Linear gaps between degrees for the polynomial calculus modulo distinct primes. *Journal of Computer and System Sciences*, 62(2):267–289, 2001.
9. Vašek Chvátal and Endre Szemerédi. Many hard examples for resolution. *Journal of the ACM (JACM)*, 35(4):759–768, 1988.
10. Matthew Clegg, Jeffery Edmonds, and Russell Impagliazzo. Using the groebner basis algorithm to find proofs of unsatisfiability. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 174–183, 1996.
11. Stephen A Cook and Robert A Reckhow. The relative efficiency of propositional proof systems. *The journal of symbolic logic*, 44(1):36–50, 1979.

12 Stefan S. Dantchev and Søren Riis. On relativisation and complexity gap. In Matthias Baaz and Johann A. Makowsky, editors, *Computer Science Logic, 17th International Workshop, CSL 2003, 12th Annual Conference of the EACSL, and 8th Kurt Gödel Colloquium, KGC 2003, Vienna, Austria, August 25-30, 2003, Proceedings*, volume 2803 of *Lecture Notes in Computer Science*, pages 142–154. Springer, 2003. doi:10.1007/978-3-540-45220-1_14.

13 Michael A. Forbes, Amir Shpilka, Iddo Tzameret, and Avi Wigderson. Proof complexity lower bounds from algebraic circuit complexity. *Theory Comput.*, 17:1–88, 2021. URL: https://theoryofcomputing.org/articles/v017a010/.

14 Nashlen Govindasamy, Tuomas Hakoniemi, and Iddo Tzameret. Simple hard instances for low-depth algebraic proofs. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 – November 3, 2022*, pages 188–199. IEEE, 2022.

15 Armin Haken. The intractability of resolution. *Theoretical computer science*, 39:297–308, 1985.

16 Russell Impagliazzo, Sasank Mouli, and Toniann Pitassi. The surprising power of constant depth algebraic proofs. In *Proceedings of the 35th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 591–603, 2020.

17 Russell Impagliazzo, Pavel Pudlák, and Jirí Sgall. Lower bounds for the polynomial calculus and the gröbner basis algorithm. *Comput. Complex.*, 8(2):127–144, 1999. doi:10.1007/s000370050024.

18 Alexander A Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Math. notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

19 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 77–82, 1987.

20 Dmitry Sokolov. (semi) algebraic proofs over $\{\pm 1\}$ variables. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 78–90, 2020.

A Appendix

We will prove Theorem 9, which we state again here for convenience.

▶ **Theorem 32** (Theorem 9). *Let $m = 10n$. Then there exists constants $k > 0$, $0 < \epsilon < 1$ such that for sufficiently large n, there exists k-CSP formulas $\{F_{n,k}\}$ with m k-local constraints such that for $m' = (1-\epsilon)m$, every subset of m' constraints is unsatisfiable and requires linear degree* PC *refutations.*

First we'll show that a random regular bipartite graph has good boundary expansion. This has been used implicitly in other works ([9], [5]), but for completeness we state and prove it here. Let $G = (L, R, E)$ be a bipartite graph, and let $A \subseteq R$. The *boundary* for A, $\partial(A)$, is the set of vertices x in L so that $|N(x) \cap A| = 1$, i.e., vertices with a unique neighbor in A. A bipartite graph is (d, k) regular if every vertex in L has degree d and every vertex in R has degree k. In this case, for $n = |L|, m = |R|$, we have $dn = km$.

▶ **Theorem 33.** *Let d, k, n, m be positive integers with $dn = km$, $k \geq 12$. Then with high probability for a random (d, k) regular bipartite graph with $|L| = n, |R| = m$, for all $A \subset R$, $|A| < n/(e^6 k^2)$, we have $\partial(A) \geq k|A|/2$.*

Proof. Let $N(A)$ be all the neighbors of A. Since the total degrees of vertices in A is $k|A|$, and each element of $N(A) - \partial(A)$ is contingent on two such edges, $k|A| \geq 2(|N(A)| - |\partial(A)|) + |\partial(A)$, or $\partial(A) \geq 2|N(A)| - k|A|$. We will show that with high probability for all such A, $|N(A)| > 3k|A|/4$, and hence $\partial(A) \geq k|A|/2$.

If not, there are sets $A \subset R$ and $B \subset L$ so that $N(A) \subseteq B$ and $|B| = 3k|A|/4$. We will bound the probability that this is true for fixed sets A, B and then take a union bound. We can view picking a random (d, k) bipartite graph as picking a random matching between d half-edges adjacent to each $x \in L$ and k such half-edges adjacent to each $y \in R$; if a half edge for x is matched to a half-edge for y, it forms an edge between x and y.

We can form this matching by going through the half edges for nodes in R and for each randomly selecting an unmatched half-edge for some node in L. We start with the edges for A in an arbitrary order. If we condition on all previous neighbors for A being in B, the number of half-edges left still available for B is less than $d|B|$, whereas the number for $\overline{B}$ stays at exactly $d(n - |B|)$. Thus, the conditional probability that the next edge formed is also in B is at most $|B|/n$, and we do this for each of $k|A|$ edges, meaning the probability that all neighbors are in B is at most $(|B|/n)^{k|A|}$.

Now, for a fixed $|A|$ and setting $|B| = 3k|A|/4$, we take the union bound over all subsets A and B. This gives a total probability of failure for some set A of size a as :

$$\binom{m}{a}\binom{n}{3ka/4}(3ka/4n)^{ka}$$

$$\leq (em/a)^a(4en/3ka)^{3ka/4}(3ka/4n)^{ka}$$

$$\leq (em/a)^a(e^3ka/n)^{ka/4} = (ekn/da)^a(e^3ka/n)^{ka/4} = (e^{3k/4+1}a^{k/4-1}k^{k/4+1}/dn^{k/4-1})^a$$

Since we are assuming $a < n/(e^6k^2)$, the base in the above expression is at most

$$e^{3k/4+1}(n/e^6k^2)^{k/4-1}k^{k/4+1}/dn^{k/4-1}$$

$$= e^{7-3k/4}k^{3-k/4}/d$$

which for $k \geq 12$ is bounded below e^{-2}, meaning the probability of such a bad set existing is exponentially small in a, and the probability of such a bad set existing for any a is less than $1/2$. ◀

▶ **Definition 34.** *For a Boolean vector $X = \{x_1, \ldots, x_n\}$, we define $\mathcal{L}_{n,m,k_1,k}(X)$ to be the distribution over k-CSP formulas over n variables $X = \{x_1, \ldots, x_n\}$ obtained by selecting m parity equations, where each parity is represented by a node on the right of a randomly chosen bipartite graph $G(L, R, E)$, with $|L| = n$, $|R| = m$, and with left degree bounded by k_1 and right degree bounded by k.*

▶ **Lemma 35.** *Let $F_{n,k}$ be a tautology given by the system of parity equations $AX = b$ over variables $X = \{x_1, \ldots, x_n\}$ drawn at random from $\mathcal{L}_{n,m,k_1,k}$ where $m = 10n$, for large enough constants $k_1, k > 0$, and b is chosen randomly. Then the following hold with high probability for a small enough $\epsilon > 0$:*

a) *Any subset of a $(1-\epsilon)$-fraction of the equations in $F_{n,k}$ is unsatisfiable*

b) *Any subset of a $(1-\epsilon)$-fraction of the equations in $F_{n,k}$ requires* PC *degree $c_2(n)$ to refute, for some $c_2 > 0$.*

Proof.

a) The probability that a set of $(1-\epsilon)10n$ random parities (i.e. for a random choice of b) is satisfiable is at most 2^{-9n} for a small enough ϵ. The probability that any such subset of $F_{n,k}$ is satisfiable is therefore at most $2^{(-n(9-10H(\epsilon)))}$, which is exponentially small for a small enough ϵ (where $H(\epsilon)$ is the binary entropy function).

b) This follows directly from [1], Theorem 3.8 and Theorem 4.4, since by Theorem 33 the bipartite graph underlying the system of parity equations A has good boundary expansion with high probability. ◀

Spectral Expanding Expanders

Gil Cohen
Department of computer science, Tel Aviv University, Israel

Itay Cohen
Department of computer science, Tel Aviv University, Israel

Abstract

Dinitz, Schapira, and Valadarsky [5] introduced the intriguing notion of *expanding expanders* – a family of expander graphs with the property that every two consecutive graphs in the family differ only on a small number of edges. Such a family allows one to add and remove vertices with only few edge updates, making them useful in dynamic settings such as for datacenter network topologies and for the design of distributed algorithms for self-healing expanders. [5] constructed explicit expanding-expanders based on the Bilu-Linial construction of spectral expanders [3]. The construction of expanding expanders, however, ends up being of *edge* expanders, thus, an open problem raised by [5] is to construct *spectral* expanding expanders (SEE).

In this work, we resolve this question by constructing SEE with spectral expansion which, like [3], is optimal up to a poly-logarithmic factor, and the number of edge updates is optimal up to a constant. We further give a simple proof for the existence of SEE that are close to Ramanujan up to a small additive term. As in [5], our construction is based on interpolating between a graph and its lift. However, to establish spectral expansion, we carefully weigh the interpolated graphs, dubbed partial lifts, in a way that enables us to conduct a delicate analysis of their spectrum. In particular, at a crucial point in the analysis, we consider the *eigenvectors* structure of the partial lifts.

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors; Theory of computation → Random walks and Markov chains

Keywords and phrases Expanders, Normalized Random Walk, Spectral Analysis

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.8

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2022/154/`

Funding *Gil Cohen*: Supported by ERC starting grant 949499 and by the Israel Science Foundation grant 1569/18.
Itay Cohen: Supported by ERC starting grant 949499.

1 Introduction

Expander graphs are among the most useful combinatorial objects in theoretical computer science, and in computer science in general. In theory, expanders proved to be pivotal in many groundbreaking results (e.g., [22, 1, 10, 6, 17, 19, 7, 4]). Informally, expanders are sparse undirected graphs that have many desirable pseudorandom properties.

There are several ways of defining the expansion of a graph. Taking the combinatorial perspective, one thinks of the edge- or vertex-expansion, whereas from the spectral point of view, the spectral expansion is considered. The latter coincides with the Markovian point of view as it captures the rate at which random walks converge. If one is willing to absorb some deterioration in parameters, it is possible to move from one definition to the next, and so in the non-extreme regime of parameters, and only there, the different definitions are, in a sense, equivalent. This work concerns with spectral expansion and so we recall the definition right away. For the formal definition of other notions of expansion, and for the relations between them, we refer the reader to the wonderful texts [9, 21, 20, 18].

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 8; pp. 8:1–8:19

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


Let G be an undirected graph with adjacency matrix $\mathbf{A}$. Since $\mathbf{A}$ is symmetric it has n real eigenvalues which we denote by $\lambda_1 \geq \cdots \geq \lambda_n$. The *spectral expansion*[1] of G is defined by $\lambda(G) \triangleq \max(\lambda_2, |\lambda_n|)$. As mentioned, the reason $\lambda(G)$ is of interest is mostly due to the fact that it captures the rate of converges of random walks on *regular* graphs. Indeed, for d-regular graphs the adjacency matrix is a simple normalization of the random walk matrix, $\mathbf{W} = \frac{1}{d}\mathbf{A}$. For many applications in theoretical computer science, and in particular in a typical work on expander graphs, restricting to regular graphs is a nonissue, and so the spectrum of the adjacency matrix is studied instead of that of the random walk matrix. For arbitrary undirected graphs, as those we will work with [2], the random walk matrix can be written as $\mathbf{W} = \mathbf{A}\mathbf{D}^{-1}$, where $\mathbf{D}$ is the diagonal matrix that encodes the degrees of the vertices of G. Since $\mathbf{W}$ is similar to the symmetric matrix $\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}}$, its eigenvalues are real, denoted $\bar{\lambda}_1 \geq \cdots \geq \bar{\lambda}_n$. The parameter of interest, which determines the rate of converges, is then the *normalized spectral expansion*, defined by $\bar{\lambda}(G) = \max(\bar{\lambda}_2, |\bar{\lambda}_n|)$.

Typically, when working with expanders one cares not about one graph but rather about an infinite family of graphs $\mathcal{G} = (G_n)_{n\in\mathcal{I}}$ where, for each $n \in \mathcal{I}$, G_n is an undirected graph on n vertices. It is typically assumed that all graphs in the family have bounded degree d and, naturally, the denser $\mathcal{I}$ is in $\mathbb{N}$, the better. We denote $\lambda(\mathcal{G}) = \sup_n \lambda(G_n)$ and similarly for the normalized quantity $\bar{\lambda}(\mathcal{G}) = \sup_n \bar{\lambda}(G_n)$. For most applications, one requires the expander family $\mathcal{G}$ to be explicit which, in this work, means that given $n \in \mathcal{I}$, the graph G_n can be generated in poly(n)-time.

1.1 Expanding expanders

Dinitz, Schapira and Valadarsky [5] introduced a natural and intriguing aspect of families of expander graphs, which, informally captures the extent to which the family is "continuous". Formally, let $\mathcal{G} = (G_n = (V_n, E_n))_{n\in\mathcal{I}}$ be a family of expander graphs. Dinitz *et al.* considered the *expansion cost* $c(\mathcal{G})$ which is the number of edges one must add or remove, from any graph in the family so as to obtain the next. If we denote the least element in $\mathcal{I}$ that is larger than n by $\text{next}_{\mathcal{I}}(n)$, then the expansion cost can be written as

$$c(\mathcal{G}) \triangleq \max_{n\in\mathcal{I}} \left|E_n \triangle E_{\text{next}_{\mathcal{I}}(n)}\right|$$

if this maximum exists and ∞ otherwise.

Clearly, high $\delta(\mathcal{G}) \triangleq \max_{n\in\mathcal{I}} (\text{next}_{\mathcal{I}}(n) - n)$ implies high expansion cost. Typically, we will think of $\mathcal{I}$ as a very dense set in $\mathbb{N}$, in particular, $\delta(\mathcal{G})$ will be bounded by some small universal constant (independent of the degree). Dinitz, Schapira and Valadarsky initiated the study of families of expander graphs with bounded expansion cost $c(\mathcal{G}) < \infty$. As noted by [5], for d-regular expanders, $c(\mathcal{G}) \geq \frac{3d}{2}$ and so the authors asked whether there is an infinite family of expanders whose expansion cost is bounded by some constant $c = c(d)$.

With such a family at hand, one can add and remove vertices with low cost in terms of edge updates while maintaining expansion. This stands in contrast to, say, a randomly sampled family in which the difference between expanders of consecutive size is linear in the number of vertices and, in particular, is unbounded. The motivation of [5] for studying such families, dubbed *expanding expanders*, originated from datacenter network topologies.

---

[1] There is some harmless inconsistency in the literature regarding the definition of spectral expansion. Some sources refer to $d - \lambda(G)$ as the spectral expansion. Others consider $1 - \frac{1}{d}\lambda(G)$. In some cases, it is only λ_2 that is considered.

[2] As we prove in the full version of the paper, in our setting, one must resort to the use of irregular graphs.

The different servers of a datacenters correspond to the vertices of a graph and the edges represent the wires connecting the servers. The degrees being bounded implies low cost in wiring, and the expansion of the graph is essential for avoiding traffic routed ineffectively. As datacenters grow regularly, low expansion cost translates to a low overhead in rewiring when adding a new server.

As another application, [5] showed how to obtain better distributed algorithms for self-healing expanders, improving upon a prior work by Pandurangan, Robinson, and Trehan [16]. Informally, self-healing expanders are expanders that can, distributively, fix themselves when vertices are added or removed. In this setting, the fact that the expander family is deterministically constructed makes the distributive task significantly simpler as there is no randomness that is needed to be communicated.

The main result of [5] is an explicit construction of a family of d-regular expanders where the expansion is with respect to edge-expansion. More concretely, it was shown how to maintain edge expansion of roughly $\frac{d}{3}$ while guaranteeing expansion cost of at most $\frac{5d}{2}$.

1.2 Our results

The construction of Dinitz, Schapira, and Valadarsky [5] for edge-expanding expanders is based on the work of Bilu and Linial [3] who analyzed the operation of "lifting" a spectral expander as a way of obtaining a graph on, say, twice the number of vertices, while maintaining spectral expansion. The [5] construction goes by way of interpolating between every two consecutive Bilu-Linial *spectral* expanders, assuring good *edge*-expansion.

Dinitz *et al.* observed that in their interpolation, some of the graphs in the family are only weak spectral expanders. That is, despite the fact that they are interpolating between spectral expanders, the construction ends up only having good edge expansion. Therefore, Dinitz *et al.* left open the question of whether spectral expanding expanders can be constructed (or exist for that matter). In this work we answer this question to the affirmative by constructing spectral expanding expanders via lifting which, spectrally, are essentially as good as the ones we are interpolating between.

▶ **Theorem 1** (Main result). *For every integer $d \geq 3$ there exists an explicit family of undirected graphs $\mathcal{G}$ such that all vertices of every graph in the family has degree bounded in $[d, 4d]$. The expansion cost $c(\mathcal{G}) = O(d)$, and the spectral- and normalized-spectral expansions are given by*

$$\lambda(\mathcal{G}) = O\left(\sqrt{d\log^3 d}\right), \quad \bar{\lambda}(\mathcal{G}) = O\left(\sqrt{\frac{\log^3 d}{d}}\right).$$

We wish to stress that, as our construction is of irregular graphs, hence, a bound on $\lambda(\mathcal{G})$ does not imply the bound on the normalized $\bar{\lambda}(\mathcal{G})$. As it turns out, our proof for the bound on $\bar{\lambda}(\mathcal{G})$ is significantly more involved than the proof for the unnormalized $\lambda(\mathcal{G})$, though the latter too requires careful analysis. In particular, the bound on $\lambda(\mathcal{G})$ is used for obtaining the bound on $\bar{\lambda}(\mathcal{G})$. The proofs of the two bounds are given in Section 4 and Section 5, and the straightforward derivation of Theorem 1 is then given in Section 6.

Theorem 1 makes the important tool of random walks on expanders available to applications that require the dynamical setting offered by expanding expanders.

Before moving on, we remark that the graphs in the family $\mathcal{G}$ that is constructed in Theorem 1 have multiple edges. As for the density of the family, $\delta(\mathcal{G}) = O(1)$ and $\min \mathcal{I} = O(d)$.

1.2.1 Expanding Ramanujan graphs

The bound obtained by Theorem 1 for $\lambda(\mathcal{G})$, and similarly for the normalized version, is off by a poly-logarithmic factor from the spectral expansion of a Ramanujan graph, namely, $2\sqrt{d-1}$ which is optimal up to a negligible additive term [15]. We leave it as an intriguing open problem to determine whether expanding Ramanujan graphs exist. It seems to us that the known algebraic constructions (e.g., [11]) are not adequate for the setting of expanding expanders. In their seminal work [12, 13, 14] Marcus, Spielman and Srivastava proved the existence of bipartite Ramanujan graphs. The question of whether expanding bipartite Ramanujan graphs exists is too an interesting one. It is unclear to us whether the proof technique of Marcus *et al.* is suitable in the expanding setting.

In the full version of the paper we give a simple proof for the existence of nearly-Ramanujan spectral expanding expanders. While our proof does not yield an explicit construction, the question of whether expanding Ramanujan graphs exist is of interest already on the combinatorial level.

▶ **Theorem 2.** *For every $\epsilon > 0$ and every even integer $d \geq 6$, there exists an infinite family $\mathcal{G}$ of d-regular spectral expanding expanders with expansion cost $c(\mathcal{G}) \leq 3d$ and spectral expansion $\lambda(\mathcal{G}) \leq 2\sqrt{d-1} + \varepsilon$.*

1.2.2 On spectral expanding expanders meeting the Ramanujan bound and the ubiquitous of Ramanujan graphs

We find the question on the existence of expanding Ramanujan graphs, as well as its bipartite analog, to be interesting also in the context of non-expanding expanders. Indeed, the existence of a family of expanding Ramanujan graphs would arguably be an indication for an affirmative answer to the fundamental open problem that asks whether a random d-regular graph, under a natural distribution, is Ramanujan with positive probability. The reasoning being that if Ramanujan graphs are sparse among graphs then they should have a certain underlying structure. That this structure happens to coincide with the structure required by expanding expanders seems far-fetched.

By the above discussion, proving the existence of spectral expanding expanders that attain the Ramanujan bound would be an indication for the ubiquitous of Ramanujan graphs. We remark, as further discussed in the full version of the paper, that for the construction of regular expanding expanders, as required in the context of Ramanujan graphs, one would need to employ a different paradigm than ours.

1.3 Organization

In Section 2 we give a proof overview of our results. The reader may freely skip this section and jump straight to Sections 4–6 in which the formal proof of Theorem 1 is given. The first section, Section 4, covers the bulk of the proof for the unnormalized case, Section 5 handles the normalized case, and in Section 6 we derive the proof of Theorem 1. Theorem 2 is proved in the full version of the paper. In the full version of the paper we further discuss the extent to which irregular graphs are necessary for constructing spectral expanding expanders. We prove that within a fairly general framework for constructing spectral expanding expanders, regularity must be abandoned.

2 Proof Overview

In this section, we give a brief account on some of the ideas that go into the proof of Theorem 1. As mentioned, bounding $\lambda(\mathcal{G})$ is simpler than (and used by) the proof for the bound on $\bar{\lambda}(\mathcal{G})$. Therefore, we start by discussing the unnormalized spectral expansion. In any case, both proofs are based on the well-known notion of a *lift* of a graph, as well as on our extension of this notion we dub *partial lifts*. These are covered in Section 2.1 and Section 2.2, respectively. A more extensive treatment of lifts can be found in [2] and references therein.

2.1 Lifts

For an integer $k \geq 2$, a *k-lift* of an undirected graph $G = (V, E)$ is an undirected graph $\widehat{G}$ on the vertex set $[k] \times V$ where each edge $\{u, v\} \in E$ induces k edges in $\widehat{G}$ that form a perfect matching between the vertices $[k] \times \{u\}$ and $[k] \times \{v\}$. The set $[k] \times \{v\}$ is called the *fiber* of v. Thus, a k-lift is determined by a choice of one perfect matching per edge $\{u, v\} \in E$ that is placed between the fibers of the two endpoints u, v. A slightly more formal treatment of the notion of a k-lift is given in Section 3.2.

A well-known fact is that the spectrum of G is inherited by that of $\widehat{G}$. Bilu and Linial [3] constructed explicit d-regular expanders by a repeated application of a 2-lift of some base graph (e.g., the clique on $d+1$ vertices). To this end, they proved that every d-regular graph has a 2-lift whose spectrum contains, on top of the eigenvalues of G, only eigenvalues that are bounded, in absolute value, by $O(\sqrt{d \log^3 d})$, thus forming a family of d-regular graphs, one for each size of the form $(d+1)2^k$, $k \in \mathbb{N}$, having spectral expansion $O(\sqrt{d \log^3 d})$. Bilu and Linial further gave an explicit construction based on a suitable derandomization of their existential proof.

We digress a bit and mention that Bilu and Linial conjectured that every d-regular graph has a 2-lift all of whose "new" eigenvalues are bounded, in absolute value, by $2\sqrt{d-1}$. This conjecture has been proved by Marcus, Spielman, and Srivastava [12] for the bipartite case.

2.2 Partial lifts

As mentioned, [5] obtained their result by interpolating between every two consecutive 2-lifts, guaranteeing that every graph between a pair of consecutive 2-lifts is a good edge-expander. Following [5], to prove Theorem 1, we also interpolate between consecutive lifts. However, proving *spectral* expansion require us to use a completely different proof technique. The underlying idea is to carefully weigh the interpolated graphs in a way that will allow us to argue about their *eigenvectors*. We dub these interpolated graphs *partial lifts*, and turn to define them.

▶ **Definition 3** (Partial lifts). *Let $G = (V, E)$ be an undirected simple graph with a k-lift $\widehat{G} = (\widehat{V} = [k] \times V, \widehat{E})$. Let (B, L) be a partition of V. The* L-partial lift *of G (with respect to $\widehat{G}$) is defined to be the undirected weighted graph $\widehat{G}_L = (\widehat{V}_L, \widehat{E}_L)$ whose vertex set consists of the vertices of B and the fibers of the vertices of L, namely, $\widehat{V}_L = B \cup ([k] \times L)$. The edge set $\widehat{E}_L$ is the union of the edges of three sets:*

1. *The edges of G connecting vertices in B.*
2. *The edges of $\widehat{G}$ connecting vertices in L.*
3. *For every edge $\{u, v\} \in E$ with $u \in B$ and $v \in L$, we add an edge of weight $\frac{1}{\sqrt{k}}$ between u and each of the vertices in the fiber of v. The edges from Items (1) and (2) have weight 1.*

The slightly more formal definition of a partial lift is given in Definition 5. Informally, we think of the vertices in B as vertices of the base graph G that are not yet lifted, and of those in L as the already lifted vertices. The edges from Items (1) and (2) form the corresponding induced graphs. The set of "hybrid" edges, appearing in Item (3), connect already-lifted and not-yet-lifted vertices, where the weight assigned to these edges is chosen in hindsight.

Note that $\widehat{G}_L$ is a simple weighted undirected graph. Moreover, $\widehat{G}_L$ interpolates between G and $\widehat{G}$ in the sense that $\widehat{G}_\emptyset = G$ and $\widehat{G}_V = \widehat{G}$. Already here we mention that for the proof of Theorem 1 we will be working with 4-lifts, or with any whole square number k for that matter, as then the $\frac{1}{\sqrt{k}}$ weight can be "simulated" without weights using parallel edges.

Before we proceed, we set some notation. We denote $b = |B|$, $\ell = |L|$, and further denote the number of vertices of $\widehat{G}_L$ by m, noting that $m = b + k\ell$. The number of vertices in G is denoted $n = b + \ell$. We denote the smallest eigenvalue of the adjacency matrix of an undirected graph H by $\lambda_{\mathsf{min}}(H)$. This will be convenient as the different graphs that we will be considering (G and $\widehat{G}_L$) will be on a different number of vertices.

2.3 Bounding the spectral expansion

Our main result with regards to the bound on the unnormalized spectral expansion is that for every L-partial lift of G (with respect to $\widehat{G}$) it holds that

$$\lambda_{\mathsf{min}}(\widehat{G}) \leq \lambda_{\mathsf{min}}(\widehat{G}_L) \leq \lambda_2(\widehat{G}_L) \leq \lambda_2(\widehat{G}). \tag{2.1}$$

This in particular implies that the spectral expansion of every L-partial lift is as good as the spectral expansion of the (fully) lifted graph, namely, $\lambda(\widehat{G}_L) \leq \lambda(\widehat{G})$. See Proposition 6 for the more complete statement.

To prove Equation (2.1), we consider the subspace $F^{\parallel}$ of $\mathbb{R}^m$ that consists of all vectors that are constants on the fibers of the lifted vertices. We denote the orthogonal complement of $F^{\parallel}$ by $F^{\perp}$, noting that it contains all vectors that sum up to zero on the fibers of the lifted vertices, and that vanish on the unlifted vertices. In the first step of the proof we characterize the eigenvectors of $\widehat{G}_L$ that lay inside $F^{\parallel}$. To do so, we order the vertices of $\widehat{G}_L$ such that the unlifted vertices, those in B, appear first. With this ordering, consider the $m \times n$ matrix

$$\mathbf{U} = \begin{pmatrix} \mathbf{I}_b & \mathbf{0} \\ \mathbf{0} & \frac{1}{\sqrt{k}}\mathbf{I}_\ell \\ \vdots & \vdots \\ \mathbf{0} & \frac{1}{\sqrt{k}}\mathbf{I}_\ell \end{pmatrix}.$$

We prove (see Lemma 8) that $\mathbf{U}\mathbf{x}$ is an eigenvector of $\widehat{G}_L$ if and only if $\mathbf{x}$ is an eigenvector of G. In fact, we weigh the hybrid edges as we did precisely for the purpose of making this statement true. At any rate, this accounts for n eigenvectors of $\widehat{G}_L$, all of which are contained in $F^{\parallel}$, whose eigenvalues are the same as those of G. In particular, every eigenvalue of G is an eigenvalue of $\widehat{G}_L$ with the same, or with higher multiplicity.

Since $\widehat{G}_L$ is symmetric, its eigenvectors are orthogonal to each other, and so the remaining eigenvectors of $\widehat{G}_L$ are contained in $F^{\perp}$. While we cannot argue that these correspond to eigenvectors of $\widehat{G}$, as one might have hoped, in the second step of the proof we show that these correspond to eigenvectors of some principal submatrix $\mathbf{M}$ of the adjacency matrix of $\widehat{G}$. This suffices for the purpose of bounding the eigenvalues as one can invoke the eigenvalue interlacing theorem (Theorem 4).

We stress that not every eigenvector of $\mathbf{M}$ induces an eigenvector of $\widehat{G}_L$, a fact that is crucial to the proof. Indeed, the crux of the proof is in showing that some "problematic" eigenvectors of $\mathbf{M}$ do not affect the spectrum of $\widehat{G}_L$. Although this is a key part of the proof, we cannot cover it without delving into more details, and so at this point we refer the reader to the formal treatment that is given in Section 4.

2.4 Bounding the normalized spectral expansion

Our main result regarding the normalized spectral expansion, which recall determines the rate of convergence of a random walk, is given by Proposition 11 and essentially states that assuming k is sufficiently small compared to $\lambda(G)$, for all $L \subseteq V$ it holds that

$$\bar{\lambda}(\widehat{G}_L) = O(k \cdot \bar{\lambda}(\widehat{G})). \tag{2.2}$$

We remark that in the normalized case, a stronger statement as in Equation (2.1) does not hold. Namely, $\lambda_2(\widehat{G}_L)$ depends on both $\lambda_2(\widehat{G})$ and $\lambda_{\mathsf{min}}(\widehat{G})$, and similarly for $\lambda_{\mathsf{min}}(\widehat{G}_L)$. Moreover, note that, unlike the unnormalized case, here k affects the bound, though the reader should keep in mind that for our construction of spectral expanding expanders, as given in Theorem 1, we will anyhow set k to 4. Another technical caveat worth mentioning is that we can only prove Equation (2.2) for a regular base graph G (which, again, suffices for the proof of Theorem 1). This is essentially because we need a good handle on an eigenvector of G that corresponds to its largest eigenvalue.

To discuss our proof strategy we introduce some notation. Let $\mathbf{M}_{\widehat{G}_L}$ be the adjacency matrix of $\widehat{G}_L$ and $\mathbf{W}_{\widehat{G}_L}$ be its random walk matrix. More precisely, if we denote by $\mathbf{D}_{\widehat{G}_L}$ the diagonal matrix that encodes the degrees of vertices in $\widehat{G}_L$ then $\mathbf{W}_{\widehat{G}_L} = \mathbf{M}_{\widehat{G}_L}\mathbf{D}_{\widehat{G}_L}^{-1}$. We first note that $\mathbf{z} = \mathbf{D}_{\widehat{G}_L}\mathbf{1}$ is an eigenvector of $\mathbf{W}_{\widehat{G}_L}$ with eigenvalue 1, and so to prove Equation (2.2) it suffices to bound the Rayleigh quotient, with respect to $\mathbf{W}_{\widehat{G}_L}$, of vectors orthogonal to $\mathbf{z}$. As $F^{\|}$ and $F^{\perp}$ are invariant subspaces of $\mathbf{W}_{\widehat{G}_L}$, it suffices to do so for each of these subspaces separately. In the first step of the proof, we use the result we already proved for the unnormalized case to deduce that

$$\forall \mathbf{x} \in F^{\perp} \quad \frac{\mathbf{x}^{\mathsf{T}}\mathbf{W}_{\widehat{G}_L}\mathbf{x}}{\mathbf{x}^{\mathsf{T}}\mathbf{x}} \leq \sqrt{k} \cdot \bar{\lambda}(\widehat{G}),$$

which allows us to turn our focus to $F^{\|}$.

To bound the Rayleigh quotient of vectors in $F^{\|}$, we characterize the eigenvectors of $\mathbf{W}_{\widehat{G}_L}$ laying in $F^{\|}$ by the eigenvectors of another operator. Formally, if $\mathbf{M}_G$ is the adjacency matrix of the base graph G then, in Lemma 14, we prove that there exists a diagonal $n \times n$ matrix $\mathbf{D}$ such that a vector $\mathbf{x} \in \mathbb{R}^n$ is an eigenvector of $\mathbf{M}_G\mathbf{D}^{-1}$ if and only if $\mathbf{U}\mathbf{x}$ is an eigenvector of $\mathbf{W}_{\widehat{G}_L}$, and both correspond to the same eigenvalue. We stress that the matrix $\mathbf{D}$ is *not* the matrix encoding the degrees of G (as indeed it should somehow encode information about L) but rather it encodes the degree of vertices in the lifted graph, where from every fiber we take only one representative.

The above leaves us with the task of studying the eigenvalues of the matrix $\mathbf{M}_G\mathbf{D}^{-1}$ which, as eluded to above, "skews" the degrees of vertices in G according to the partial lift structure. The crux of the proof, which we will not be able to cover in this high level proof overview, boils down to bounding the sum of reciprocal of these skewed degrees

$$\sum_{v \in V} \frac{1}{\mathbf{D}_{v,v}}$$

(see Lemmas 19 and 20). We refer the reader to Section 5 for the formal treatment.

3 Preliminaries

We start by setting some fairly standard notation from spectral graph theory.

3.1 Spectral graph theory

The adjacency matrix of an undirected graph $G = (V, E)$ is denote by $\mathbf{M}_G$. Being real and symmetric, $\mathbf{M}_G$ has $n = |V|$ real eigenvalues which we denote by $\lambda_1(\mathbf{M}_G) \geq \cdots \geq \lambda_n(\mathbf{M}_G)$. For $i \in [n]$ we define $\lambda_i(G) = \lambda_i(\mathbf{M}_G)$, and write $\lambda_{\mathsf{min}}(G)$ for $\lambda_n(G)$. We refer to the eigenvectors of $\mathbf{M}_G$ as the eigenvectors of G. The *spectral expansion* of G is given by $\lambda(G) \triangleq \max(\lambda_2(G), |\lambda_n(G)|)$.

Let $\mathbf{D}_G$ be the *degrees matrix* of G, that is, the matrix that encodes the degrees of vertices in G (under the same order that they appear in $\mathbf{M}_G$). Assuming G has no isolated vertices, the random walk matrix of G, denoted $\mathbf{W}_G$, is defined by $\mathbf{W}_G = \mathbf{M}_G\mathbf{D}_G^{-1}$. Note that $\mathbf{W}_G$ has n real eigenvalues as it is similar to the symmetric matrix $\mathbf{D}_G^{-\frac{1}{2}}\mathbf{M}_G\mathbf{D}_G^{-\frac{1}{2}}$. We denote these by $\bar{\lambda}_1(G) \geq \cdots \geq \bar{\lambda}_n(G) \triangleq \bar{\lambda}_{\mathsf{min}}(G)$ and refer to them as the *normalized eigenvalues* of G. The *normalized spectral expansion* of G is given by $\bar{\lambda}(G) \triangleq \max(\bar{\lambda}_2(G), |\bar{\lambda}_n(G)|)$.

For a family $\mathcal{G} = (G_n)_{n\in\mathcal{I}}$ of expander graphs, we let $\lambda(\mathcal{G}) = \sup_{n\in\mathcal{I}} \lambda(G_n)$ if the maximum exists, and ∞ otherwise, and similarly define $\bar{\lambda}(\mathcal{G}) = \sup_{n\in\mathcal{I}} \bar{\lambda}(G_n)$.

We make use of the well-known fact that the eigenvalues of a real symmetric matrix interlace with the eigenvalues of any of its principal submatrices. For a proof see, e.g., [8], Theorem 9.1.1.

▶ **Theorem 4** (Eigenvalue Interlacing Theorem). *Let* $\mathbf{N}$ *be a real symmetric* $n \times n$ *matrix and let* $\mathbf{M}$ *be an* $m \times m$ *principal submatrix of* $\mathbf{N}$*. Then, for all* $i \in [m]$*,*

$$\lambda_i(\mathbf{N}) \geq \lambda_i(\mathbf{M}) \geq \lambda_{n-m+i}(\mathbf{N}).$$

3.2 Lifts

In contrast to the introductory part, from here on we define the notion of a lift in a somewhat more formal way, which is also easier to work with. To this end, we first recall the notion of graph orientation. Let $G = (V, E)$ be a simple undirected graph. An orientation of G is an assignment of a direction to each of its edges, resulting with a directed graph which we denote by $\vec{G} = (V, \vec{E})$. That is, for every undirected edge $\{u, v\}$ of G exactly one of (u, v), (v, u) is included in $\vec{E}$.

In what comes next, we consider maps $\pi : \vec{E} \to S_k$ where $\vec{E}$ is the edge (multi-)set of some oriented graph $\vec{G}$ and, as customary, S_k is the group of permutations on $[k]$. For ease of notation, we write $\pi_{u,v}$ instead of the more cumbersome expression $\pi((u, v))$.

Let $G = (V, E)$ be an undirected simple graph on n vertices, $\vec{G} = (V, \vec{E})$ an orientation of G, and let $\pi : \vec{E} \to S_k$ for some integer $k \geq 1$. The *π-lift* of $\vec{G}$ is the graph $\vec{G}_\pi = ([k] \times V, E_\pi)$ where for every $(u, v) \in \vec{E}$ we include the edges

$$\{(i, u), (\pi_{u,v}(i), v)\} \quad \text{for} \quad i = 1, 2, \ldots, k$$

in E_π. Note that regardless of the choice of orientation (and regardless of the choice of π), since G is simple so is $\vec{G}_\pi$. For $v \in V$, the set of vertices $[k] \times \{v\}$ of $\vec{G}_\pi$ is called the *fiber* of v. For ease of notation, from hereon we write G_π for $\vec{G}_\pi$ despite the fact that this graph depends on the chosen orientation.

We extend the map $\pi : \vec{E} \to S_k$ to the set $\{(v,u) \mid (u,v) \in \vec{E}\}$ as follows: For $(u,v) \in \vec{E}$ we set $\pi_{v,u} = \pi_{u,v}^{-1}$. With this, it is convenient to write down the adjacency matrix of G_π as follows. For $i,j \in [k]$ define the zero-one $n \times n$ matrix $\mathbf{M}_G^{i,j}$ by

$$(\mathbf{M}_G^{i,j})_{u,v} = 1 \quad \iff \quad \{u,v\} \in E \ \text{ and } \ \pi_{u,v}(i) = j.$$

Then, the adjacency matrix $\mathbf{M}_{G_\pi}$ of G_π is the $k \times k$ block matrix, where block (i,j) is given by the $n \times n$ matrix $(\mathbf{M}_{G_\pi})_{i,j} = \mathbf{M}_G^{i,j}$. That is,

$$\mathbf{M}_{G_\pi} = \begin{pmatrix} \mathbf{M}_G^{1,1} & \mathbf{M}_G^{1,2} & \cdots & \mathbf{M}_G^{1,k} \\ \mathbf{M}_G^{2,1} & \mathbf{M}_G^{2,2} & \cdots & \mathbf{M}_G^{2,k} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{M}_G^{k,1} & \mathbf{M}_G^{k,2} & \cdots & \mathbf{M}_G^{k,k} \end{pmatrix}.$$

Note that $\mathbf{M}_G^{i,j} = (\mathbf{M}_G^{j,i})^\mathsf{T}$. A well-known fact about lifting is that the spectrum of the base graph G is contained in the spectrum of the lifted graph G_π. More precisely, if λ is an eigenvalue of G with multiplicity r then λ is an eigenvalue of G_π with multiplicity at least r. This is easily seen by noting that for every $i \in [k]$,

$$\sum_{j=1}^{k} \mathbf{M}_G^{i,j} = \mathbf{M}_G, \tag{3.1}$$

and so every eigenvector $\mathbf{x}$ of $\mathbf{M}_G$ induces the eigenvector $(\mathbf{x}, \ldots, \mathbf{x})$ of $\mathbf{M}_{G_\pi}$ with the same eigenvalue.

4 Bounding the Spectral Expansion

We start this section by introducing the notion of a *partial lift* and study its properties. Throughout, we make use of the notation from Section 3.2.

▶ **Definition 5.** Let $G = (V,E)$ be an undirected simple graph, and let $\vec{G} = (V,\vec{E})$ be an orientation of G. Let $\pi : \vec{E} \to S_k$ for some $k \geq 1$, and let (B,L) be a partition of the vertices of G. The *L-partial π-lift* of $\vec{G}$ is defined to be the undirected weighted graph $G_{\pi,L} = (V_{\pi,L}, E_{\pi,L})$ whose vertex set is $V_{\pi,L} = B \cup ([k] \times L)$. The edge set $E_{\pi,L}$ is the union of

$$E_B = \{\{u,v\} \in E \mid u,v \in B\},$$
$$E_L = \{\{(i,u),(j,v)\} \in E_\pi \mid u,v \in L\},$$

and

$$E_H = \{\{u,(i,v)\} \mid i \in [k] \text{ and } u \in B,\, v \in L \text{ s.t. } \{u,v\} \in E\},$$

with weight of $\frac{1}{\sqrt{k}}$ assigned to each edge in E_H. The edges in E_B, E_L have a unit weight assigned to them.

Note that E_B is the set of edges of the B-induced sub-graph of the base graph G, and E_L is the set of edges of the induced graph of G_π with respect to the fibers of the lifted vertices. The set $E_{\pi,L}$ contains the "hybrid" edges, connecting already-lifted and not-yet-lifted vertices where the weight assigned to these edges is chosen with a hindsight.

Note that $G_{\pi,L}$ is a weighted undirected simple graph. Moreover, $G_{\pi,L}$ interpolates between G and G_π in the sense that $G_{\pi,\emptyset} = G$ and $G_{\pi,V} = G_\pi$. Observe that, assuming G is d-regular, the weighted degree of every vertex in $G_{\pi,L}$ is in the range $[d/\sqrt{k}, \sqrt{k} \cdot d]$.

The main result of this section is the following proposition, which is the more complete and formal version of Equation (2.1) from the Proof Overview section.

▶ **Proposition 6.** Let $G = (V, E)$ be an undirected simple graph with orientation $\vec{G} = (V, \vec{E})$. Let $\pi : \vec{E} \to S_k$, and (B, L) a partition of V. Then,

$$\lambda_{\mathsf{min}}(G_\pi) \leq \lambda_{\mathsf{min}}(G_{\pi,L}) \leq \lambda_2(G_{\pi,L}) \leq \lambda_2(G_\pi) \leq \lambda_1(G_\pi) = \lambda_1(G_{\pi,L}).$$

Proof. Note that the non-trivial inequalities and equality, which we set to prove, are

$$\lambda_{\mathsf{min}}(G_\pi) \leq \lambda_{\mathsf{min}}(G_{\pi,L}), \tag{4.1}$$

$$\lambda_2(G_{\pi,L}) \leq \lambda_2(G_\pi), \tag{4.2}$$

$$\lambda_1(G_\pi) = \lambda_1(G_{\pi,L}). \tag{4.3}$$

Let $\mathbf{M}_G$ be the adjacency matrix of G where we order the rows and columns so that those corresponding to vertices in B appear first, namely,

$$\mathbf{M}_G = \begin{pmatrix} \mathbf{M}_B & \mathbf{M}_H \\ \mathbf{M}_H^\mathsf{T} & \mathbf{M}_L \end{pmatrix}.$$

Note that $\mathbf{M}_B, \mathbf{M}_L$ are the adjacency matrices of the B-induced and L-induced subgraphs of G, respectively. Observe that the adjacency matrix of $G_{\pi,L}$ is given by

$$\mathbf{M}_{G_{\pi,L}} = \begin{pmatrix} \mathbf{M}_B & \frac{1}{\sqrt{k}}\mathbf{M}_H & \cdots & \frac{1}{\sqrt{k}}\mathbf{M}_H \\ \frac{1}{\sqrt{k}}\mathbf{M}_H^\mathsf{T} & \mathbf{M}_L^{1,1} & \cdots & \mathbf{M}_L^{1,k} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{\sqrt{k}}\mathbf{M}_H^\mathsf{T} & \mathbf{M}_L^{k,1} & \cdots & \mathbf{M}_L^{k,k} \end{pmatrix},$$

where $\mathbf{M}_L^{i,j}$ is a slight abuse of notation (recall we defined $\mathbf{M}_H^{i,j}$ for a *graph* H) and is used as a shorthand for $\mathbf{M}_H^{i,j}$, H being the L-induced subgraph of G. Similar to Equation (3.1), we have

$$\forall i \in [k] \quad \sum_{j=1}^{k} \mathbf{M}_L^{i,j} = \mathbf{M}_L. \tag{4.4}$$

Let n, m be the number of vertices in G and in $G_{\pi,L}$, respectively. Denote $b = |B|, \ell = |L|$, and note that $n = b + \ell$ and $m = b + k\ell$. Define

$$F^{\parallel} = \left\{ \mathbf{x} \in \mathbb{R}^m \mid \mathbf{x}_{b+j} = \mathbf{x}_{b+\ell+j} = \cdots = \mathbf{x}_{b+(k-1)\ell+j} \quad \text{for } j = 1, 2, \ldots, \ell \right\}.$$

Informally, $F^{\parallel}$ is the space of vectors that are constant on the fibers of the lifted vertices, and are otherwise arbitrary. Let $F^{\perp}$ be the dual subspace of $F^{\parallel}$, namely,

$$F^{\perp} = \left\{ \mathbf{x} \in \mathbb{R}^m \mid \mathbf{x}_1 = \cdots = \mathbf{x}_b = 0 \text{ and } \sum_{i=0}^{k-1} \mathbf{x}_{b+i\ell+j} = 0 \quad \text{for } j = 1, 2, \ldots, \ell \right\}.$$

It is easy to verify, using Equation (4.4), that both $F^{\parallel}$ and $F^{\perp}$ are invariant subspaces of $\mathbf{M}_{G_{\pi,L}}$. Define the matrix $\mathbf{U} \in \mathbb{R}^{m \times n}$

$$\mathbf{U} = \begin{pmatrix} \mathbf{I}_b & \mathbf{0} \\ \mathbf{0} & \frac{1}{\sqrt{k}}\mathbf{I}_\ell \\ \vdots & \vdots \\ \mathbf{0} & \frac{1}{\sqrt{k}}\mathbf{I}_\ell \end{pmatrix}. \tag{4.5}$$

The following claim lists some useful, easy to prove, properties of $\mathbf{U}$.

▷ Claim 7. $\mathbf{U}$ satisfies the following properties:

1. $\mathrm{Im}(\mathbf{U}) = F^{\|}$.
2. The right kernel of $\mathbf{U}$ is $\mathbf{0}$.
3. $\mathbf{U}^{\mathsf{T}}\mathbf{U} = \mathbf{I}_n$.
4. $\mathbf{U}\mathbf{U}^{\mathsf{T}}$ is the orthogonal projection to $F^{\|}$.
5. $\mathbf{U}^{\mathsf{T}}\mathbf{M}_{G_{\pi,L}}\mathbf{U} = \mathbf{M}_G$.

We can now easily characterize all the eigenvectors of $G_{\pi,L}$ laying in $F^{\|}$.

▶ **Lemma 8.** *For every $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}$ is an eigenvector of G corresponding to an eigenvalue λ if and only if $\mathbf{U}\mathbf{x}$ is an eigenvector of $G_{\pi,L}$ laying in $F^{\|}$ and corresponding to λ.*

Proof.

$$\begin{aligned}\mathbf{M}_G\mathbf{x} = \lambda\mathbf{x} \quad &\iff \quad \mathbf{U}^{\mathsf{T}}\mathbf{M}_{G_{\pi,L}}\mathbf{U}\mathbf{x} = \lambda\mathbf{x} \\ &\iff \quad \mathbf{U}\mathbf{U}^{\mathsf{T}}\mathbf{M}_{G_{\pi,L}}\mathbf{U}\mathbf{x} = \lambda\mathbf{U}\mathbf{x} \\ &\iff \quad \mathbf{M}_{G_{\pi,L}}\mathbf{U}\mathbf{x} = \lambda\mathbf{U}\mathbf{x},\end{aligned}$$

where the first implication follows from Item (5) of Claim 7, the second from Item (2), and the last follows by Items (1) and (4) together with the invariance of $F^{\|}$ under $\mathbf{M}_{G_{\pi,L}}$. ◀

To summarize, we have found n eigenvectors of $\mathbf{M}_{G_{\pi,L}}$, all of which are contained in $F^{\|}$, whose eigenvalues correspond to those of $\mathbf{M}_G$. In particular, every eigenvalue of $\mathbf{M}_G$ is an eigenvalue of $\mathbf{M}_{G_{\pi,L}}$ with the same, or higher, multiplicity. Observe that $F^{\|}$ is defined by $(k-1)\ell$ linear constraints. Hence, its dimension is exactly $n = b + k\ell - (k-1)\ell$. We conclude that the characterized eigenvectors are exactly all the eigenvectors of $G_{\pi,L}$ in $F^{\|}$.

We proceed to explore the remaining eigenvectors of $\mathbf{M}_{G_{\pi,L}}$. Since $\mathbf{M}_{G_{\pi,L}}$ is symmetric, its eigenvectors are orthogonal to each other, and so the remaining eigenvectors of $\mathbf{M}_{G_{\pi,L}}$ are contained in $F^{\perp}$. We turn to prove that while we cannot argue that these correspond to eigenvectors of $\mathbf{M}_{G_\pi}$, as one might hope, they will correspond to eigenvectors of some principal submatrix of $\mathbf{M}_{G_\pi}$, at which point we can invoke the Eigenvalue Interlacing Theorem (see Theorem 4) so to bound the corresponding eigenvalues.

Take $\mathbf{x} \in F^{\perp}$ an eigenvector of $\mathbf{M}_{G_{\pi,L}}$ with eigenvalue λ. Then, $\mathbf{x} = (\mathbf{0}, \mathbf{z})$ for some nonzero $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_k) \in \mathbb{R}^{k\ell}$ where $\sum_{i=1}^{k} \mathbf{z}_i = \mathbf{0} \in \mathbb{R}^{\ell}$. Therefore,

$$\lambda\mathbf{x} = \mathbf{M}_{G_{\pi,L}}\mathbf{x} = \begin{pmatrix} \mathbf{0} \\ \mathbf{M}_{\pi,L}\mathbf{z} \end{pmatrix},$$

where

$$\mathbf{M}_{\pi,L} = \begin{pmatrix} \mathbf{M}_L^{1,1} & \cdots & \mathbf{M}_L^{1,k} \\ \vdots & \ddots & \vdots \\ \mathbf{M}_L^{k,1} & \cdots & \mathbf{M}_L^{k,k} \end{pmatrix}.$$

Hence, $\mathbf{z}$ is an eigenvector of $\mathbf{M}_{\pi,L}$. That is, all eigenvectors of $\mathbf{M}_{G_{\pi,L}}$ that are contained in $F^{\perp}$ correspond to eigenvectors of $\mathbf{M}_{\pi,L}$.

We stress that not every eigenvector of $\mathbf{M}_{\pi,L}$ induces an eigenvector of $\mathbf{M}_{G_{\pi,L}}$, a fact that will be crucial in what follows. Indeed, the eigenvectors of $\mathbf{M}_{G_{\pi,L}}$ coming from $F^{\perp}$ have the special structure described above of being orthogonal to $\mathbf{1}$ on each fiber. This can also be seen by a dimension argument, noting that $\mathbf{M}_{\pi,L}$ has $k\ell$ eigenvectors, and together with the n eigenvectors that are induced from $\mathbf{M}_G$ these amount to $k\ell + n$ eigenvectors. However, $\mathbf{M}_{G_{\pi,L}}$ is a matrix of order $(k\ell + b) \times (k\ell + b)$, and so $n - b = \ell$ eigenvectors of $\mathbf{M}_{\pi,L}$ do not induce eigenvectors of $\mathbf{M}_{G_{\pi,L}}$. At any rate, for convenience, we summarize the analysis so far.

▷ Claim 9. The spectrum of $\mathbf{M}_{G_{\pi,L}}$ consists of the spectrum of $\mathbf{M}_G$ with corresponding eigenvectors in $F^{\|}$ as well as of a subset of the spectrum of $\mathbf{M}_{\pi,L}$ with corresponding eigenvectors in $F^{\perp}$, where we remind the reader that we consider the spectrum as a multi-set so to track multiplicities correctly.

Note that $\mathbf{M}_{\pi,L}$ is a principal submatrix of $\mathbf{M}_{G_\pi}$ and so, by the Eigenvalue Interlacing Theorem (Theorem 4),

$$\lambda_{\mathsf{min}}(G_\pi) \leq \lambda_{\mathsf{min}}(\mathbf{M}_{\pi,L}) \leq \lambda_2(\mathbf{M}_{\pi,L}) \leq \lambda_2(G_\pi). \tag{4.6}$$

Further, recall that the spectrum of G_π contains that of G, in particular,

$$\lambda_{\mathsf{min}}(G_\pi) \leq \lambda_{\mathsf{min}}(G) \leq \lambda_2(G) \leq \lambda_2(G_\pi). \tag{4.7}$$

By putting together Claim 9, Equation (4.6) and Equation (4.7), we establish Equation (4.1). For proving Equation (4.2) we are left to prove that an eigenvector $\mathbf{x} \in F^{\perp}$ of $G_{\pi,L}$ cannot correspond to an eigenvalue $\lambda > \lambda_2(\mathbf{M}_{\pi,L})$. To this end, let $\mathbf{x} = (\mathbf{0}, \mathbf{z}) \in F^{\perp}$ be an eigenvector of $G_{\pi,L}$. Recall that for every $j \in [\ell]$,

$$\sum_{i=0}^{k-1} \mathbf{z}_{b+i\ell+j} = 0. \tag{4.8}$$

Assume for contradiction that $\mathbf{x}$ is an eigenvector of $G_{\pi,L}$ corresponding to an eigenvalue $\lambda > \lambda_2(\mathbf{M}_{\pi,L})$. Then, by the above discussion, $\mathbf{z}$ is an eigenvector of $\mathbf{M}_{\pi,L}$ with eigenvalue $\lambda > \lambda_2(\mathbf{M}_{\pi,L})$, meaning $\lambda = \lambda_1(\mathbf{M}_{\pi,L})$. As the vector corresponding the largest eigenvalue, $\mathbf{z}$ maximizes the Rayleigh quotient, we have that

$$\lambda_1(\mathbf{M}_{\pi,L}) = \frac{\mathbf{z}^{\mathsf{T}}\mathbf{M}_{\pi,L}\mathbf{z}}{\mathbf{z}^{\mathsf{T}}\mathbf{z}} = \max_{\mathbf{w}\neq\mathbf{0}} \frac{\mathbf{w}^{\mathsf{T}}\mathbf{M}_{\pi,L}\mathbf{w}}{\mathbf{w}^{\mathsf{T}}\mathbf{w}}.$$

Note, however, that the vector $|\mathbf{z}|$, which is obtained by taking the absolute value of every entry of $\mathbf{z}$, satisfies

$$\frac{|\mathbf{z}|^{\mathsf{T}}\mathbf{M}_{\pi,L}|\mathbf{z}|}{|\mathbf{z}|^{\mathsf{T}}|\mathbf{z}|} \geq \frac{\mathbf{z}^{\mathsf{T}}\mathbf{M}_{\pi,L}\mathbf{z}}{\mathbf{z}^{\mathsf{T}}\mathbf{z}}, \tag{4.9}$$

and so, as a maximizer of the Rayleigh quotient, $|\mathbf{z}|$ is also an eigenvector with eigenvalue $\lambda_1(\mathbf{M}_{\pi,L})$. However, by Equation (4.8), $\mathbf{z}$ and $|\mathbf{z}|$ are linearly independent. Indeed, there is a fiber on which $\mathbf{z}$ attains both a positive and negative entries. Hence, we have found two independent vectors corresponding to $\lambda_1(\mathbf{M}_{\pi,L})$, implying $\lambda_1(\mathbf{M}_{\pi,L}) = \lambda_2(\mathbf{M}_{\pi,L})$. This stands in contradiction to $\lambda_1(\mathbf{M}_{\pi,L}) = \lambda > \lambda_2(\mathbf{M}_{\pi,L})$. Putting this result together with Claim 9, Equation (4.6) and Equation (4.7) completes the proof of Equation (4.2). To prove Equation (4.3), we will use a general result on graph lifts.

▶ **Lemma 10.** *Let $G = (V, E)$ be an undirected simple graph with orientation $\vec{G} = (V, \vec{E})$. Let $\pi : \vec{E} \to S_k$. Then, $\lambda_1(G) = \lambda_1(G_\pi)$.*

Proof. By invoking Lemma 8 to $G_\pi = G_{\pi,V}$, we get that $\lambda_1(G)$ is an eigenvalue of G_π, which implies $\lambda_1(G) \leq \lambda_1(G_\pi)$. Proving $\lambda_1(G_\pi) \leq \lambda_1(G)$ will thus finish the proof. For any vector $\mathbf{x}$ on the vertices of G_π, take the vector $\mathbf{y}(\mathbf{x}) = \mathbf{y}$ on the vertices of G defined by $\mathbf{y}_v = \sqrt{\sum_{i=1}^{k} \mathbf{x}_{i,v}^2}$. Now, recall the definition of $|\mathbf{x}|$ and note that

$$\mathbf{y}^{\mathsf{T}}\mathbf{y} = \sum_{v\in V} \mathbf{y}_v^2 = \sum_{v\in V}\sum_{i=1}^{k} \mathbf{x}_{i,v}^2 = \mathbf{x}^{\mathsf{T}}\mathbf{x} = |\mathbf{x}|^{\mathsf{T}}|\mathbf{x}|.$$

Therefore,

$$\begin{aligned}
\frac{\mathbf{y}^\mathsf{T}\mathbf{M}_G\mathbf{y}}{\mathbf{y}^\mathsf{T}\mathbf{y}} &= \frac{1}{\mathbf{y}^\mathsf{T}\mathbf{y}} \sum_{(u,v)\in\vec{E}} \mathbf{y}_u\mathbf{y}_v \\
&= \frac{1}{\mathbf{x}^\mathsf{T}\mathbf{x}} \sum_{(u,v)\in\vec{E}} \sqrt{\sum_{i=1}^{k}\mathbf{x}_{i,u}^2 \cdot \sum_{i=1}^{k}\mathbf{x}_{i,v}^2} \\
&= \frac{1}{|\mathbf{x}|^\mathsf{T}|\mathbf{x}|} \sum_{(u,v)\in\vec{E}} \sqrt{\sum_{i=1}^{k}|\mathbf{x}_{i,u}|^2 \cdot \sum_{i=1}^{k}|\mathbf{x}_{\pi_{u,v}(i),v}|^2} \\
&\geq \frac{1}{|\mathbf{x}|^\mathsf{T}|\mathbf{x}|} \sum_{(u,v)\in\vec{E}} \sum_{i=1}^{k}|\mathbf{x}_{i,u}||\mathbf{x}_{\pi_{u,v}(i),v}| \\
&= \frac{|\mathbf{x}|^\mathsf{T}\mathbf{M}_{G_\pi}|\mathbf{x}|}{|\mathbf{x}|^\mathsf{T}|\mathbf{x}|} \\
&\geq \frac{\mathbf{x}^\mathsf{T}\mathbf{M}_{G_\pi}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}},
\end{aligned}$$

where the third equality is just a reordering of the elements in the summation, as $\pi_{u,v}$ is a permutation on $[k]$. The first inequality follows by the Cauchy-Schwarz inequality, and the second inequality is as in Equation (4.9). Thus,

$$\lambda_1(G_\pi) = \max_{\mathbf{x}\neq 0}\frac{\mathbf{x}\mathbf{M}_{G_\pi}\mathbf{x}}{\mathbf{x}^\mathsf{T}\mathbf{x}} \leq \max_{\mathbf{x}\neq 0}\frac{\mathbf{y}(\mathbf{x})\mathbf{M}_G\mathbf{y}(\mathbf{x})}{\mathbf{y}(\mathbf{x})^\mathsf{T}\mathbf{y}(\mathbf{x})} \leq \max_{\mathbf{z}\neq 0}\frac{\mathbf{z}\mathbf{M}_G\mathbf{z}}{\mathbf{z}^\mathsf{T}\mathbf{z}} = \lambda_1(G). \quad \blacktriangleleft$$

Combining the results we proved so far, we conclude that

$$\lambda_1(G) \leq \lambda_1(G_{\pi,L}) \leq \lambda_1(G_\pi) = \lambda_1(G).$$

Indeed, the first inequality follows by Lemma 8, the second follows by Claim 9 and by the fact that $\mathbf{M}_{\pi,L}$ is a principal submatrix of $\mathbf{M}_{G_{\pi,L}}$, together with the Eigenvalue Interlacing Theorem (Theorem 4). Lastly, the equality follows by Lemma 10, completing the proof of Proposition 6. ◀

5 Bounding the Normalized Spectral Expansion

In this section we bound the normalized eigenvalues of a partial lift. That is, we show that a random walk on a partial lift converges quickly given that the random walk on the (full) lift does so. Unlike the unnormalized case, we restrict ourselves to d regular base graphs.

▶ **Proposition 11.** *Let $G = (V, E)$ be an undirected simple d-regular graph having orientation $\vec{G} = (V, \vec{E})$. Let $\pi : \vec{E} \to S_k$, and (B, L) a partition of V. Assume that $\sqrt{k} \leq \frac{\lambda(G)}{3} + 1$. Then,*

$$\bar{\lambda}(G_{\pi,L}) = O(k \cdot \bar{\lambda}(G_\pi)).$$

Proof. Note that the degrees matrix of $G_{\pi,L}$, $\mathbf{D}_{G_{\pi,L}}$, is constant on every fiber, and so $F^{\|}$ and $F^{\perp}$ are invariant subspaces of $\mathbf{D}_{G_{\pi,L}}^{-1}$. As noted in the proof of Proposition 6, these are also invariant subspaces of $\mathbf{M}_{G_{\pi,L}}$, hence, also of $\mathbf{W}_{G_{\pi,L}} = \mathbf{M}_{G_{\pi,L}}\mathbf{D}_{G_{\pi,L}}^{-1}$.

Equation (4.2) and Equation (4.1) implies that $\lambda(G_{\pi,L}) \leq \lambda(G_\pi)$. This, together with the fact that the eigenvector corresponding to the largest eigenvalue of $G_{\pi,L}$ lays in $F^{\|}$, implies that

$$\mathbf{x}, \mathbf{y} \in F^{\perp} \quad \left|\mathbf{x}^{\mathsf{T}} \mathbf{M}_{G_{\pi,L}} \mathbf{y}\right| \leq \lambda(G_\pi) \|\mathbf{x}\|_2 \|\mathbf{y}\|_2.$$

This, together with Item (4) of Claim 13, which we state below, yields the desired bound on the Rayleigh quotient for all vectors in $F^{\perp}$. Indeed, for every $\mathbf{x} \in F^{\perp}$, we have that

$$\frac{|\mathbf{x}^{\mathsf{T}} \mathbf{W}_{G_{\pi,L}} \mathbf{x}|}{\mathbf{x}^{\mathsf{T}} \mathbf{x}} = \frac{|\mathbf{x}^{\mathsf{T}} \mathbf{M}_{G_{\pi,L}} \mathbf{D}_{G_{\pi,L}}^{-1} \mathbf{x}|}{\mathbf{x}^{\mathsf{T}} \mathbf{x}} \leq \frac{\lambda(G_\pi) \cdot \|\mathbf{x}\|_2 \cdot \|\mathbf{D}_{G_{\pi,L}}^{-1} \mathbf{x}\|_2}{\mathbf{x}^{\mathsf{T}} \mathbf{x}} \leq \frac{\sqrt{k}}{d} \cdot \lambda(G_\pi) = \sqrt{k} \cdot \bar{\lambda}(G_\pi).$$

We summarize this in the following corollary.

▶ **Corollary 12.** *The Rayleigh quotient of all eigenvectors laying in $F^{\perp}$, with respect to $\mathbf{W}_{G_{\pi,L}}$, are bounded by $\sqrt{k} \cdot \bar{\lambda}(G_\pi)$.*

Since $F^{\|}$ and $F^{\perp}$ are invariant subspaces of $\mathbf{W}_{G_{\pi,L}}$, we are left to analyze the vectors laying in $F^{\|}$. To this end, define the diagonal matrix $\mathbf{D} \in \mathbb{R}^{n \times n}$ by $\mathbf{D} = \mathbf{U}^{\mathsf{T}} \mathbf{D}_{G_{\pi,L}} \mathbf{U}$, where we recall the reader that the definition of $\mathbf{U}$ is in Equation (4.5). By this definition, since the degrees of vertices on the same fiber are equal, we get

$$\mathbf{D}_{u,u} = \begin{cases} \deg_{G_{\pi,L}}(u) & u \in B; \\ \deg_{G_{\pi,L}}((1,u)) & u \in L. \end{cases} \tag{5.1}$$

For a vertex v of G we define θ_v, the *cut degree* of v, to be

$$\theta_v = \begin{cases} |E_G(v, L)| & v \in B; \\ |E_G(v, B)| & v \in L, \end{cases} \tag{5.2}$$

where, for $S, T \subseteq V$, $|E_G(S,T)|$ is the sum of weights of edges between (S,T) in G. Denote $k_B = \sqrt{k} - 1$ and $k_L = \frac{1}{\sqrt{k}} - 1$. The following claim is easy to verify and is stated without a proof.

▷ Claim 13. The matrices $\mathbf{D}_{G_{\pi,L}}$ and $\mathbf{D}$ have the following properties:

1. $\mathbf{D}_{G_{\pi,L}} \mathbf{U} = \mathbf{U}\mathbf{D}$ and $\mathbf{D}_{G_{\pi,L}}^{-1} \mathbf{U} = \mathbf{U}\mathbf{D}^{-1}$.
2. For a vertex $u \in B$ the value of $\mathbf{D}_{u,u}$ is given by

$$1 \cdot |E_G(u, B)| + \sqrt{k} \cdot |E_G(u, L)| = d + k_B \theta_u.$$

3. For a vertex $u \in L$ the value of $\mathbf{D}_{u,u}$ is given by

$$1 \cdot |E_G(u, L)| + \frac{1}{\sqrt{k}} \cdot |E_G(u, B)| = d + k_L \theta_u.$$

4. $(\mathbf{D}_{G_{\pi,L}})_{u,u} \in [\frac{1}{\sqrt{k}} d, \sqrt{k} d]$.
5. $\mathbf{D}_{u,u} \in [\frac{1}{\sqrt{k}} d, \sqrt{k} d]$.

▶ **Lemma 14.** *A vector $\mathbf{x} \in \mathbb{R}^n$ is an eigenvector of $\mathbf{M}_G \mathbf{D}^{-1}$ corresponding to eigenvalue λ if and only if $\mathbf{U}\mathbf{x}$ is an eigenvector of $\mathbf{W}_{G_{\pi,L}}$ corresponding to the same eigenvalue.*

Proof. By Items (5) and (2) of Claim 7,

$$\begin{aligned}\mathbf{M}_G\mathbf{D}^{-1}\mathbf{x} = \lambda\mathbf{x} \quad &\iff \quad \mathbf{U}^\mathsf{T}\mathbf{M}_{G_{\pi,L}}\mathbf{U}\mathbf{D}^{-1}\mathbf{x} = \lambda\mathbf{x} \\ &\iff \quad \mathbf{U}\mathbf{U}^\mathsf{T}\mathbf{M}_{G_{\pi,L}}\mathbf{U}\mathbf{D}^{-1}\mathbf{x} = \lambda\mathbf{U}\mathbf{x}.\end{aligned}$$

Thus, by Item (1) of Claim 13,

$$\mathbf{M}_G\mathbf{D}^{-1}\mathbf{x} = \lambda\mathbf{x} \quad \iff \quad \mathbf{U}\mathbf{U}^\mathsf{T}\mathbf{W}_{G_{\pi,L}}\mathbf{U}\mathbf{x} = \lambda\mathbf{U}\mathbf{x}.$$

By Item (1) of Claim 7, $\mathbf{U}\mathbf{x} \in F^{\|}$ and since $F^{\|}$ is an invariant subspace of $\mathbf{W}_{G_{\pi,L}}$, we get that $\mathbf{W}_{G_{\pi,L}}\mathbf{U}\mathbf{x} \in F^{\|}$. Item (4) of Claim 7 then implies that

$$\mathbf{U}\mathbf{U}^\mathsf{T}\mathbf{W}_{G_{\pi,L}}\mathbf{U}\mathbf{x} = \mathbf{W}_{G_{\pi,L}}\mathbf{U}\mathbf{x},$$

and so

$$\mathbf{M}_G\mathbf{D}^{-1}\mathbf{x} = \lambda\mathbf{x} \quad \iff \quad \mathbf{W}_{G_{\pi,L}}\mathbf{U}\mathbf{x} = \lambda\mathbf{U}\mathbf{x},$$

as desired. ◀

Given Corollary 12 and Lemma 14, and since every vector in $F^{\|}$ is of the form $\mathbf{U}\mathbf{x}$ for some $\mathbf{x} \in \mathbb{R}^n$ (recall Item (1) of Claim 7), we can turn our focus to characterizing the eigenvalues of $\mathbf{M}_G\mathbf{D}^{-1}$. We do so by analyzing the eigenvalues of the symmetric matrix $\widetilde{\mathbf{M}}_G = \mathbf{D}^{-\frac{1}{2}}\mathbf{M}_G\mathbf{D}^{-\frac{1}{2}}$ as, note, it is similar to $\mathbf{M}_G\mathbf{D}^{-1}$. To start with, observe that $\mathbf{D}_{G_{\pi,L}}\mathbf{1}_m$ is an eigenvector of $\mathbf{W}_{G_{\pi,L}}$ corresponding to its largest eigenvalue, 1. Thus, Lemma 14 together with Item (4) of Claim 7 imply that $\mathbf{U}^\mathsf{T}\mathbf{D}_{G_{\pi,L}}\mathbf{1}$ is an eigenvector of $\mathbf{M}_G\mathbf{D}^{-1}$ corresponding to the eigenvalue 1. By Item (1) of Claim 13, $\mathbf{U}^\mathsf{T}\mathbf{D}_{G_{\pi,L}} = \mathbf{D}\mathbf{U}^\mathsf{T}$, and so the latter eigenvector can be written as $\mathbf{D}\mathbf{U}^\mathsf{T}\mathbf{1}$. As $\mathbf{M}_G\mathbf{D}^{-1}$ is similar to $\widetilde{\mathbf{M}}_G$, by the above, the latter has an eigenvector $\widetilde{\mathbf{x}}_1 = \mathbf{D}^{\frac{1}{2}}\mathbf{U}^\mathsf{T}\mathbf{1}$ corresponding to its largest eigenvalue, 1. Since $\widetilde{\mathbf{M}}_G$ is symmetric, an eigenvector corresponding to its second largest eigenvalue has to be orthogonal to $\widetilde{\mathbf{x}}_1$.

As we assume G is regular, $\mathbf{1}_n$ is an eigenvector corresponding to $\mathbf{M}_G$-s largest eigenvalue. For a vector $\mathbf{y}$, let $\mathbf{y}^{\|} = \langle \mathbf{1}_n, \mathbf{y}\rangle \cdot \mathbf{1}_n$ be the orthogonal projection of $\mathbf{y}$ onto $\mathbf{1}_n$. We write $\mathbf{y}^{\perp} = \mathbf{y} - \mathbf{y}^{\|}$ and turn to analyze the Rayleigh quotient of a vector $\mathbf{y} \perp \widetilde{\mathbf{x}}_1$. Let $\mathbf{z} = \mathbf{D}^{-\frac{1}{2}}\mathbf{y}$ and note that $\mathbf{y} \perp \widetilde{\mathbf{x}}_1$ if and only if $\mathbf{z} \perp \mathbf{D}^{\frac{1}{2}}\widetilde{\mathbf{x}}_1$. With this, we have that

$$\frac{\mathbf{y}^\mathsf{T}\widetilde{\mathbf{M}}_G\mathbf{y}}{\mathbf{y}^\mathsf{T}\mathbf{y}} = \frac{\mathbf{z}^\mathsf{T}\mathbf{M}_G\mathbf{z}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}} = \frac{(\mathbf{z}^{\|})^\mathsf{T}\mathbf{M}_G\mathbf{z}^{\|}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}} + \frac{(\mathbf{z}^{\perp})^\mathsf{T}\mathbf{M}_G\mathbf{z}^{\perp}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}}. \tag{5.3}$$

Using Item (5) of Claim 13, it is clear that $\frac{d}{\sqrt{k}}\|\mathbf{z}\|^2 \leq \mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}$. We can thus get a good bound on the size of the second summand in the right hand side of Equation (5.3), namely,

$$\left|\frac{(\mathbf{z}^{\perp})^\mathsf{T}\mathbf{M}_G\mathbf{z}^{\perp}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}}\right| \leq \frac{\lambda(G)\|\mathbf{z}\|^2}{\frac{d}{\sqrt{k}}\|\mathbf{z}\|^2} = \sqrt{k}\cdot\bar{\lambda}(G). \tag{5.4}$$

The first summand in the right hand side of Equation (5.3) equals to

$$\frac{(\mathbf{z}^{\|})^\mathsf{T}\mathbf{M}_G\mathbf{z}^{\|}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}} = d\cdot\frac{\|\mathbf{z}^{\|}\|^2}{\|\mathbf{z}\|^2_{\mathbf{D}}} \tag{5.5}$$

and is always non-negative. This already gives a bound on $\bar{\lambda}_{\mathsf{min}}(\widetilde{\mathbf{M}}_G)$, because for every $\mathbf{y} \perp \widetilde{\mathbf{x}}_1$, we have

$$\frac{\mathbf{y}^\mathsf{T}\widetilde{\mathbf{M}}_G\mathbf{y}}{\mathbf{y}^\mathsf{T}\mathbf{y}} = \frac{(\mathbf{z}^{\|})^\mathsf{T}\mathbf{M}_G\mathbf{z}^{\|}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}} + \frac{(\mathbf{z}^{\perp})^\mathsf{T}\mathbf{M}_G\mathbf{z}^{\perp}}{\mathbf{z}^\mathsf{T}\mathbf{D}\mathbf{z}} \geq 0 - \sqrt{k}\cdot\bar{\lambda}(G).$$

We summarize this in the following corollary.

▶ **Corollary 15.** $\left|\lambda_{\mathsf{min}}(\widetilde{\mathbf{M}}_G)\right| \leq \sqrt{k} \cdot \bar{\lambda}(G)$.

Going back to the first summand in the right hand side of Equation (5.3), per Equation (5.5), we are left to bound the quotient $\frac{\|\mathbf{z}^{\|}\|}{\|\mathbf{z}\|_{\mathbf{D}}}$. We start with the numerator. As $\mathbf{z} \perp \mathbf{D}^{\frac{1}{2}}\widetilde{\mathbf{x}}_1$, for every $\alpha \in \mathbb{R}$ we have that

$$\|\mathbf{z}^{\|}\| = \langle \mathbf{z}, \mathbf{1} \rangle = \left\langle \mathbf{z}, \mathbf{1} - \alpha \mathbf{D}^{\frac{1}{2}}\widetilde{\mathbf{x}}_1 \right\rangle = \left\langle \mathbf{D}^{\frac{1}{2}}\mathbf{z}, \mathbf{D}^{-\frac{1}{2}}\mathbf{1} - \alpha\widetilde{\mathbf{x}}_1 \right\rangle \leq \|\mathbf{D}^{\frac{1}{2}}\mathbf{z}\| \|\mathbf{D}^{-\frac{1}{2}}\mathbf{1} - \alpha\widetilde{\mathbf{x}}_1\|. \quad (5.6)$$

Choosing the optimal α for the bound, $\alpha = \frac{\left\langle \mathbf{D}^{-\frac{1}{2}}\mathbf{1}, \widetilde{\mathbf{x}}_1 \right\rangle}{\|\widetilde{\mathbf{x}}_1\|^2}$, we get that

$$\|\mathbf{D}^{-\frac{1}{2}}\mathbf{1} - \alpha\widetilde{\mathbf{x}}_1\|^2 = \|\mathbf{D}^{-\frac{1}{2}}\mathbf{1}\|^2 - \frac{\left\langle \mathbf{D}^{-\frac{1}{2}}\mathbf{1}, \widetilde{\mathbf{x}}_1 \right\rangle^2}{\|\widetilde{\mathbf{x}}_1\|^2}. \quad (5.7)$$

We now turn to analyze each of the summands in the above expression. To this end, we introduce the following notations. Let $s = |(B, L)|$ be the size of the cut (B, L) in G, and let $e = s - \frac{b\ell}{n}d$. Note that, had G been a d-regular random graph, s would have equal to $\frac{b\ell}{n}d$ in expectation, and so we think of e as the "cut size error". It will also be convenient to consider e, b and ℓ-s normalized counterpart $\bar{e} = \frac{e}{nd}$, $\bar{b} = \frac{b}{n}$ and $\bar{\ell} = \frac{\ell}{n}$.

From this point, denote $\lambda = \lambda(G)$ and $\bar{\lambda} = \bar{\lambda}(G) = \frac{\lambda}{d}$. We further define $\mu = b + \sqrt{k}\ell$ and its normalized counterpart $\bar{\mu} = \frac{\mu}{\sqrt{mn}}$ (where this normalization is in hindsight). The analysis of Equation (5.7) is divided to the following three claims.

▷ Claim 16.

$$\|\widetilde{\mathbf{x}}_1\|^2 = \frac{2\,|E(G_{\pi,L})|}{m} = d\bar{\mu}^2 - (\sqrt{k}-1)^2\frac{e}{m},$$

where recall $|E(G_{\pi,L})|$ counts the number of edges, accounting for the weights.

▷ Claim 17. $\left\langle \mathbf{D}^{-\frac{1}{2}}\mathbf{1}, \widetilde{\mathbf{x}}_1 \right\rangle = \bar{\mu}$.

▷ Claim 18. $\|\mathbf{D}^{-\frac{1}{2}}\mathbf{1}\|^2 \leq \frac{1+k\bar{\lambda}}{d}$.

Claim 16 and Claim 17 follow by a fairly straightforward calculation. Their proofs can be found in the full version of the paper. Claim 18, whose proof also appears in the full version, follows by the following two more substantial lemma.

▶ **Lemma 19.**

$$\sum_{v \in B} \frac{1}{\mathbf{D}_{v,v}} \leq \frac{b}{d}\left(\frac{1}{1 + k_B\bar{\ell}} + \sqrt{k}\bar{\lambda}\right).$$

▶ **Lemma 20.**

$$\sum_{v \in L} \frac{1}{\mathbf{D}_{v,v}} \leq \frac{\ell}{d}\left(\frac{1}{1 + k_L\bar{b}} + k\bar{\lambda}\right).$$

The proof of Lemma 19 and Lemma 20 can be found in the full version of the paper, and we proceed with the proof of Proposition 11. Using Claim 16 and Claim 17, we write

$$\begin{aligned}\frac{\left\langle \mathbf{D}^{-\frac{1}{2}}\mathbf{1}, \widetilde{\mathbf{x}}_1 \right\rangle^2}{\|\widetilde{\mathbf{x}}_1\|^2} &= \frac{1}{d}\cdot\frac{\bar{\mu}^2}{\bar{\mu}^2-(\sqrt{k}-1)^2\frac{e}{dm}} \\ &= \frac{1}{d}\cdot\frac{1}{1-(\sqrt{k}-1)^2\frac{e}{dm\bar{\mu}^2}} \\ &\geq \frac{1}{d}\left(1+(\sqrt{k}-1)^2\frac{e}{dm\bar{\mu}^2}\right) \\ &\geq \frac{1}{d}\left(1-(\sqrt{k}-1)^2\frac{|e|}{dm\bar{\mu}^2}\right). \end{aligned} \tag{5.8}$$

Focusing on the error term,

$$(\sqrt{k}-1)^2\frac{|e|}{dm\bar{\mu}^2} = (\sqrt{k}-1)^2\frac{|\bar{e}|}{(\mu/n)^2},$$

observe that according to the expander mixing lemma, $|\bar{e}| \leq \bar{\lambda}\sqrt{\bar{b}\bar{\ell}} \leq \bar{\lambda}$. Additionally, $\frac{\mu}{n}$ is bounded below by 1. The error term is thus bounded above by $\bar{\lambda}(\sqrt{k}-1)^2$. Plugging the above back to Equation (5.8), we get

$$\frac{\left\langle \mathbf{D}^{-\frac{1}{2}}\mathbf{1}, \tilde{\mathbf{x}}_1 \right\rangle^2}{\|\tilde{\mathbf{x}}_1\|^2} \geq \frac{1}{d}\cdot\left(1-(\sqrt{k}-1)^2\bar{\lambda}\right). \tag{5.9}$$

By Equation (5.6), Equation (5.7), Claim 18, and Equation (5.9),

$$\begin{aligned}\|\mathbf{z}^{\|}\| &\leq \|\mathbf{D}^{\frac{1}{2}}\mathbf{z}\|\cdot\sqrt{\|\mathbf{D}^{-\frac{1}{2}}\mathbf{1}\|^2-\frac{\left\langle \mathbf{D}^{-\frac{1}{2}}\mathbf{1}, \widetilde{\mathbf{x}}_1 \right\rangle^2}{\|\widetilde{\mathbf{x}}_1\|^2}} \\ &\leq \frac{\|\mathbf{D}^{\frac{1}{2}}\mathbf{z}\|\cdot\sqrt{\bar{\lambda}}}{\sqrt{d}}\cdot\sqrt{k+(\sqrt{k}-1)^2}. \end{aligned} \tag{5.10}$$

By Equation (5.5) and Equation (5.10), using that $\|\mathbf{z}\|_{\mathbf{D}} = \|\mathbf{D}^{\frac{1}{2}}\mathbf{z}\|$,

$$\frac{(\mathbf{z}^{\|})^{\mathsf{T}}\mathbf{M}_G\mathbf{z}^{\|}}{\mathbf{z}^{\mathsf{T}}\mathbf{D}\mathbf{z}} = d\cdot\frac{\|\mathbf{z}^{\|}\|^2}{\|\mathbf{z}\|_{\mathbf{D}}^2} \leq (k+(\sqrt{k}-1)^2)\bar{\lambda}.$$

Thus, by Equation (5.3) and Equation (5.4),

$$\lambda_2(\widetilde{\mathbf{M}}_G) \leq d\cdot\frac{\|\mathbf{z}^{\|}\|^2}{\|\mathbf{z}\|_{\mathbf{D}}^2} + \sqrt{k}\cdot\bar{\lambda}(G) \leq (k+\sqrt{k}+(\sqrt{k}-1)^2)\bar{\lambda} = O(k\cdot\bar{\lambda}).$$

Combining this with Corollary 15, we obtain $\lambda(\widetilde{\mathbf{M}}_G) = O(k\cdot\bar{\lambda})$. Restating this result, for every $\mathbf{x}\in F^{\|}$ with $\mathbf{x}\perp\mathbf{D}_{G_{\pi,L}}\mathbf{1}_m$ we have $\mathbf{x}^{\mathsf{T}}\mathbf{W}_{G_{\pi,L}}\mathbf{x} = O(k\cdot\bar{\lambda})$. Recall $\bar{\lambda} = \bar{\lambda}(G) \leq \bar{\lambda}(G_\pi)$. Combining this and Corollary 12 we conclude $\bar{\lambda}(G_{\pi,L}) = O(k\cdot\bar{\lambda}(G_\pi))$, which completes the proof. ◀

6 Proof of Theorem 1

In this section we wrap it all up and prove Theorem 1.

Proof of Theorem 1. By repeatedly applying Corollary 3.1 of [3] to the base graph, denoted BL_0, which is the clique on $d+1$ vertices, we obtain an explicit family of d-regular expanders $\mathcal{BL} = (\mathrm{BL}_n)_{n\in\mathcal{I}}$, where $\mathcal{I} = \{(d+1)\cdot 2^i \mid i \in \mathbb{N}\}$ and $\lambda(\mathcal{BL}) = O(\sqrt{d\log^3 d})$. Moreover, as BL_0 is simple, all graphs in $\mathcal{BL}$ are simple. By regularity,

$$\bar{\lambda}(\mathcal{BL}) = \frac{1}{d}\lambda(\mathcal{BL}) = O\left(\sqrt{\frac{\log^3 d}{d}}\right).$$

Let $n : \mathbb{N} \to \mathcal{I}$ be the function defined by $n(i) = (d+1)\cdot 2^i$. That is, $n(i)$ is the number of vertices of the i-th graph in $\mathcal{BL}$. Observe that for every $i \in \mathcal{I}$, $\mathrm{BL}_{n(i+2)}$ is a π_i-lift of $\mathrm{BL}_{n(i)}$ with $k = 4$. Indeed, the 2-lift of a 2-lift is a 4-lift.

Fix $i \in \mathbb{N}$. Choose an arbitrary ordering on the vertices of $\mathrm{BL}_{n(i)}$ and denote the first j vertices, under this ordering, by $L_{n(i),j}$. Define $P_{n(i),j}$ to be the $L_{n(i),j}$-partial π_i-lift of $\mathrm{BL}_{n(i)}$. Note that the edge weights of $P_{n(i),j}$ are 1 and $\frac{1}{2}$. In order to avoid fractional edges, multiply every edge by 2 to obtain the unweighted graph with multiple edges $G_{n(i),j}$. Note that the latter is a graph on $n(i)+3j$ vertices, and that $G_{n(i),n(i)} = G_{n(i+2),0}$. The family that we construct is given by $\mathcal{G} = (G_{n,j})_{(n,j)\in\mathcal{I}'}$, where the index set

$$\mathcal{I}' = \{(n(i), j) \mid i \in 2\mathbb{N} \text{ and } 0 \le j \le n(i)\}$$

is lexicographically ordered.

Per Definition 5, and as we duplicated all edges, the degrees of all vertices in this family are in the range $[2\frac{d}{\sqrt{4}}, 2\sqrt{4}d] = [d, 4d]$, as claimed. Proposition 6 and Proposition 11 readily imply the bounds on $\lambda(\mathcal{G})$ and $\bar{\lambda}(\mathcal{G})$, respectively. To conclude the proof, note that the expansion cost is $O(d)$.

◀

References

1 Miklós Ajtai, János Komlós, and Endre Szemerédi. Deterministic simulation in logspace. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 132–140, 1987.

2 Alon Amit and Nathan Linial. Random graph coverings. I. General theory and graph connectivity. *Combinatorica*, 22(1):1–18, 2002. doi:10.1007/s004930200000.

3 Yonatan Bilu and Nathan Linial. Lifts, discrepancy and nearly optimal spectral gap. *Combinatorica*, 26(5):495–519, 2006. doi:10.1007/s00493-006-0029-7.

4 Nikolas P. Breuckmann and Jens N. Eberhardt. Balanced product quantum codes. *IEEE Trans. Inform. Theory*, 67(10):6653–6674, 2021. doi:10.1109/tit.2021.3097347.

5 Michael Dinitz, Michael Schapira, and Asaf Valadarsky. Explicit expanding expanders. *Algorithmica*, 78(4):1225–1245, 2017. doi:10.1007/s00453-016-0269-x.

6 Irit Dinur. The PCP theorem by gap amplification. *J. ACM*, 54(3):Art. 12, 44, 2007. doi:10.1145/1236457.1236459.

7 Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 357–374, 2022.

8 Chris Godsil and Gordon F Royle. *Algebraic graph theory*, volume 207. Springer Science & Business Media, 2001.

9 Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

10 Russell Impagliazzo, Noam Nisan, and Avi Wigderson. Pseudorandomness for network algorithms. In *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 356–364, 1994.

11 Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, 1988.

12 Adam Marcus, Daniel A Spielman, and Nikhil Srivastava. Interlacing families I: Bipartite Ramanujan graphs of all degrees. In *2013 IEEE 54th Annual Symposium on Foundations of computer science*, pages 529–537. IEEE, 2013.

13 Adam W Marcus, Daniel A Spielman, and Nikhil Srivastava. Interlacing families IV: Bipartite Ramanujan graphs of all sizes. *SIAM Journal on Computing*, 47(6):2488–2509, 2018.

14 Adam W. Marcus, Daniel A. Spielman, and Nikhil Srivastava. Finite free convolutions of polynomials. *Probab. Theory Related Fields*, 182(3-4):807–848, 2022. doi:10.1007/s00440-021-01105-w.

15 A. Nilli. On the second eigenvalue of a graph. *Discrete Math.*, 91(2):207–210, 1991. doi:10.1016/0012-365X(91)90112-F.

16 Gopal Pandurangan, Peter Robinson, and Amitabh Trehan. DEX: self-healing expanders. *Distrib. Comput.*, 29(3):163–185, 2016. doi:10.1007/s00446-015-0258-3.

17 Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):Art. 17, 24, 2008. doi:10.1145/1391289.1391291.

18 Daniel A Spielman. Spectral and algebraic graph theory, 2019. URL: http://cs-www.cs.yale.edu/homes/spielman/sagt/.

19 Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 238–251, 2017.

20 Luca Trevisan. Lecture notes on graph partitioning, expanders and spectral methods, 2017. University of California, Berkeley, https://people.eecs.berkeley.edu/luca/books/expanders-2016.pdf.

21 Salil P Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012.

22 Leslie G. Valiant. Graph-theoretic properties in computational complexity. *J. Comput. System Sci.*, 13(3):278–285, 1976. doi:10.1016/S0022-0000(76)80041-4.

Hardness Against Linear Branching Programs and More

Eshan Chattopadhyay
Cornell University, Ithaca, NY, USA

Jyun-Jie Liao
Cornell University, Ithaca, NY, USA

Abstract

In a recent work, Gryaznov, Pudlák and Talebanfard (CCC '22) introduced a linear variant of read-once branching programs, with motivations from circuit and proof complexity. Such a read-once linear branching program is a branching program where each node is allowed to make $\mathbb{F}_2$-linear queries, and is read-once in the sense that the queries on each path is linearly independent. As their main result, they constructed an explicit function with average-case complexity $2^{n/3-o(n)}$ against a slightly restricted model, which they call strongly read-once linear branching programs. The main tool in their lower bound result is a new type of extractor, called directional affine extractors, that they introduced.

Our main result is an explicit function with $2^{n-o(n)}$ average-case complexity against the strongly read-once linear branching program model, which is almost optimal. This result is based on a new connection from this problem to sumset extractors, which is a randomness extractor model introduced by Chattopadhyay and Li (STOC '16) as a generalization of many other well-studied models including two-source extractors, affine extractors and small-space extractors. With this new connection, our lower bound naturally follows from a recent construction of sumset extractors by Chattopadhyay and Liao (STOC '22). In addition, we show that directional affine extractors imply sumset extractors in a restricted setting. We observe that such restricted sumset sources are enough to derive lower bounds, and obtain an arguably more modular proof of the lower bound by Gryaznov, Pudlák and Talebanfard.

We also initiate a study of pseudorandomness against linear branching programs. Our main result here is a hitting set generator construction against regular linear branching programs with constant width. We derive this result based on a connection to Kakeya sets over finite fields.

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Expander graphs and randomness extractors; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases linear branching programs, circuit lower bound, sumset extractors, hitting sets

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.9

Related Version *Preliminary Version*: `https://eccc.weizmann.ac.il/report/2022/153/`

Funding Supported by NSF CAREER award 2045576.

Acknowledgements We thank Jason Gaitonde for collaboration during initial stages of this project. We thank anonymous reviewers for helpful comments.

1 Introduction

The central goal of complexity theory is to understand the power and limitation of different computation models. Motivated by this goal, it is natural to study the *lower bound problem*: given a computation model and a corresponding complexity measure, can we find an explicit function (e.g. computable in polynomial time) that has large complexity? Researchers have studied this problem on many interesting circuit models such as bounded-depth circuits

(AC^0), DeMorgan formula and branching programs, and many interesting results have been found. For example, one of the most notable results in this field is that it requires exponential number of gates to compute parity in AC^0 [54, 31]. (See the excellent book by Jukna [34] for more about circuit lower bound problems.)

Interestingly, circuit lower bound problems have found connections with *randomness extraction*, another central problem in complexity theory. The theory of randomness extraction is concerned with the following problem: we are given an unknown distribution $\mathbf{X}$ which is guaranteed to have some amount of entropy, and our goal is to find an efficiently computable function Ext, which is called a *randomness extractor*, such that $\mathrm{Ext}(\mathbf{X})$ is (close to) the uniform distribution. Unfortunately, it turns out to be impossible to design extractors in this generality, and a central line of inquiry has been to consider extracting random bits assuming some additional structure on $\mathbf{X}$. Randomness extractors have also found a variety of applications in other areas of theoretical computer science, including proving lower bounds for various computational models. For example, the state-of-the-art lower bound for Boolean circuits is based on affine extractors [40], which are extractors that work for weak sources that are uniform over affine subspaces. Affine extractors were also used to obtain almost optimal lower bounds for DNF of parities and parity decision trees [19]. As another example, extractors for sources sampled by small-space algorithms [35] were shown to be average-case hard against read-once branching program [8].

The main idea behind this connection is as follows. Suppose one can show that for every function $f : \mathcal{X} \to \{0, 1\}$ with small complexity measure, the uniform distribution over the larger pre-image (say, $f^{-1}(0)$) is a source $\mathbf{X}$ with some specific structure. If one can construct an extractor for weak sources with this structure, then $f(\mathbf{X})$ is a constant while $\mathrm{Ext}(\mathbf{X})$ is close to uniform, immediately implying that f and Ext cannot be the same function. In fact, f cannot even approximately compute Ext much better than a random guess, i.e., Ext exhibits *average-case hardness* against f. For instance, to derive average-case lower bounds for parity decision trees, for which it is not hard to see that the pre-image is a disjoint union of affine subspaces, one can choose Ext to be an affine extractor. However, the choice of the extractor is not always obvious. For example, the connection between general Boolean circuits and affine extractors [20, 25, 40] is more non-trivial.

In this paper, we study the lower bound problem for *read-once linear branching programs* [29]. Our main contribution is a new connection between lower boundS for read-once linear branching programs and *sumset extractors* [10], which we will discuss in later sections.

1.1 Linear branching programs

Read-once linear branching programs (ROLBPs) were first studied by Gryaznov, Pudlák and Talebanfard [29], motivated by its connection to proof complexity. Roughly speaking, a ROLBP is a read-once branching program that can make linear queries. We leave the definition of "read-once" for later and define a linear branching program first.

▶ **Definition 1** (Linear branching program [29]). *A linear branching program on $\mathbb{F}_2^n$ is a directed acyclic graph P with the following properties:*

- *There is only one source s in P.*
- *There are two sinks in P, labeled with* 0 *and* 1 *respectively.*
- *Every non-sink node v is labeled with a linear function $\ell_v : \mathbb{F}_2^n \to \mathbb{F}_2$, which is called the* (linear) query *on node v. Moreover, there are exactly two outgoing edges from v, one is labeled with* 1 *and the other is labeled with* 0*.*

The size *of P is the number of non-sink nodes in P. We say P computes a boolean function $f : \mathbb{F}_2^n \to \{0,1\}$ in the following way. For every input $x \in \mathbb{F}_2^n$, we define the* computation path *of x as starting from s, and when on a node v which is not a sink, moving to the next node following the edge with label $\ell_v(x) \in \{0,1\}$. We repeat this process until the path ends at a sink. $f(x)$ is defined to be the label on this sink.*[1]

The most natural definition of "read-once" for a linear branching program is that the queries made on every path is linearly independent. In this paper, we focus on a more restricted model called *strongly read-once*.[2]

▶ **Definition 2** (Strongly Read-Once [29]). *For every node v in a branching program P, define Pre_v to be the span of all queries that appear on any path from the source to v, and Post_v to be the span of all queries that appear on any path from v to a sink. (For every non-sink node v, both Pre_v and Post_v include ℓ_v. For any sink w we define $\mathsf{Post}_w = \{0\}$.) We say P is* strongly read-once *if the following two properties hold.*

- *For every edge $e = (u \to v)$, $\mathsf{Pre}_u \cap \mathsf{Post}_v = \{0\}$.*
- *For every non-sink node v, $\mathsf{Pre}_v \cap \mathsf{Post}_v = \{0, \ell_v\}$.*

As pointed out in [29], although being more restricted than the natural definition of read-once, strongly read-once linear branching programs still generalize two important models: parity decision trees and read-once branching programs. A parity decision tree is a decision tree which can make linear queries. This model was first defined by Kushilevitz and Mansour [37] for its connection with Fourier analysis, and has recently received attention because of its connections to special cases of the log-rank conjecture in communication complexity [51, 32] and quantum query complexity [28]. A read-once branching program is another generalization of decision tree such that different paths can share nodes, and can be used to model streaming algorithms and randomized small-space algorithms. Similar to how decision trees are generalized to parity decision trees, it is natural to study ROBPs with linear queries.

The lower bound problem we are trying to answer is the following:

▶ **Question 3.** *For a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, let $\mathsf{ROLBP}(f)$ denote the smallest possible size of a strongly read-once linear branching program that computes f. Can we find an explicit function f which is computable in polynomial time such that $\mathsf{ROLBP}(f)$ is as large as possible?*

Note that every function f has a trivial size upper bound $\mathsf{ROLBP}(f) \leq 2^n$ (e.g. a trivial decision tree of depth n), so our goal is to find a function f such that $\mathsf{ROLBP}(f)$ is as close to 2^n as possible. We are also interested in answering the average-case lower bound problem:

▶ **Question 4.** *For a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ and any $\varepsilon > 0$, let $\mathsf{ROLBP}_\varepsilon(f)$ denote the smallest size of strongly read-once linear BP P such that*

$$\Pr_{x \sim \mathbb{F}_2^n}[P(x) = f(x)] \geq \frac{1}{2} + \varepsilon.$$

Can we find a function f which is computable in polynomial time such that $\mathsf{ROLBP}_\varepsilon(f)$ is as large as possible?

[1] In this paper, we sometimes abuse notation and also use P to denote the function computed by P.

[2] Our definition here is slightly more general than the original one in [29], but we don't view it as a substantial difference. We choose the definition here merely for simpler notation in the proofs. See Appendix A for further discussions.

1.2 Prior work

To obtain a lower bound for strongly read-once linear branching programs, [29] introduced a new type of extractor called *directional affine extractors*. (We refer the reader to Section 2 for standard notation in the context of extractors.)

▶ **Definition 5** (Directional Affine Extractor [29]). *We say* $\mathrm{DAExt} : \mathbb{F}_2^n \to \mathbb{F}_2$ *is a* (d, ε)*-directional affine extractor if for any distribution* $\mathbf{X} \in \mathbb{F}_2^n$ *which is uniform over an affine subspace of dimension* d*, and any non-zero vector* $a \in \mathbb{F}_2^n$*, it holds that*

$$\mathrm{DAExt}(\mathbf{X} + a) + \mathrm{DAExt}(\mathbf{X}) \approx_\varepsilon \mathbf{U}_1.$$

[29] proved that a directional affine extractor for small dimension has a large average-case lower bound for strongly read-once linear BP.

▶ **Theorem 6** ([29, Theorem 17]). *Let* DAExt *be a* (d, ε)*-directional affine extractor. Then*

$$\mathsf{ROLBP}_{\sqrt{2\varepsilon}}(\mathrm{DAExt}) \geq \varepsilon 2^{n-d-1}.$$

In [29], they constructed a directional affine extractor for dimension $(2/3 + o(1))n$, which implied a $2^{n/3-o(n)}$ average-case lower bound for ROLBPs. A natural open question left in [29] was to construct a directional affine extractor for dimension $d = o(n)$, which would directly imply a $2^{n-o(n)}$ average-case lower bound for ROLBPs. However, this seems like a challenging problem. Indeed, even constructing affine extractors for dimension $d = o(n)$ has been a difficult task that has been recently resolved [41, 7]; a directional affine extractor is an affine extractor with additional *non-malleable* properties (see Appendix B) and it is not clear how to use known techniques to construct such extractors for low dimension.

1.3 Our results

In this work, we take a different approach and show that to get an average-case lower bound strongly read-once linear BP, it suffices to construct a *sumset extractor*. Informally, a sumset extractor is a function that can extract uniform randomness from sum of two independent weak sources (such sources are called sumset sources). The formal definition of sumset extractors is as follows.[3]

▶ **Definition 7** (Sumset Extractor [10]). *A function* $\mathrm{SumExt} : \mathbb{F}_2^n \to \{0,1\}$ *is a* (k_1, k_2, ε)*-sumset extractor if for any two independent distributions* $\mathbf{A}, \mathbf{B}$ *on* $\mathbb{F}_2^n$ *with* $\mathrm{H}_\infty(\mathbf{A}) \geq k_1$ *and* $\mathrm{H}_\infty(\mathbf{B}) \geq k_2$*,*

$$\mathrm{SumExt}(\mathbf{A} + \mathbf{B}) \approx_\varepsilon \mathbf{U}_1.$$

Our main theorem is as follows:

▶ **Theorem 8.** *Let* SumExt *be a* (k_1, k_2, ε)*-sumset extractor. Then*

$$\mathsf{ROLBP}_{9\varepsilon}(\mathrm{SumExt}) > 2^{n-k_1-k_2-2}.$$

Sumset extractors were first introduced by Chattopadhyay and Li [10] as a "unified" extractor model for many other important extractor problems such as two-source extractors, affine extractors and small-space extractors. (We refer the reader to [13] for a more elaborate discussion on sumset extractors.) A recent work [13] gave an explicit construction of sumset extractors for polylogarithmic entropy.

[3] For simplicity, we present the definition where the output length of the extractor is just 1 bit.

▶ **Theorem 9** ([13]). *There is a* $(\text{polylog}(n), \text{polylog}(n), n^{-\Omega(1)})$*-sumset extractor that can be computed in polynomial time.*

Plugging this extractor into Theorem 8, we improve the best lower bound for strongly read-once linear BP from $2^{n/3-o(n)}$ to $2^{n-o(n)}$, which is almost optimal.

▶ **Theorem 10.** *There is a function* SumExt *which can be computed in polynoimal time such that*

$$\mathsf{ROLBP}_{n^{-\Omega(1)}}(\text{SumExt}) > 2^{n-\text{polylog}(n)}.$$

1.4 On average-case lower bound with negligible error

One drawback of the average-case lower bound based on Theorem 8 is that we don't yet know any explicit construction of (k_1, k_2, ε)-sumset extractors such that $k_1 + k_2 \leq n$ and $\varepsilon = n^{-\omega(1)}$.[4] Thus we cannot directly use Theorem 8 to derive non-trivial average-case lower bound in the negligible correlation setting (for functions in P). (Note that the $2^{n/3-o(n)}$ lower bound in [29] does have negligible correlation.) However, a closer inspection at the proof of Theorem 8 actually shows that it suffices to construct extractors for sumset sources $\mathbf{A} + \mathbf{B}$ with two additional properties, that we describe below.

▶ **Theorem 11.** *Let* $\text{SumExt}' : \mathbb{F}_2^n \to \{0,1\}$ *be a function such that* $\text{SumExt}'(\mathbf{A} + \mathbf{B}) \approx_\varepsilon \mathbf{U}_1$ *for any independent distributions* $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$ *which satisfy* $\text{H}_\infty(\mathbf{A}) \geq k_1, \text{H}_\infty(\mathbf{B}) \geq k_2$ *and the following two additional properties.*

- $\mathbf{B}$ *is* almost affine*: the span of* $\text{Supp}(\mathbf{B})$ *is of dimension* $\leq k_2 + 1$.
- $\mathbf{A}$ *and* $\mathbf{B}$ *have* non-intersecting span*:* $\text{span}(\text{Supp}(\mathbf{A})) \cap \text{span}(\text{Supp}(\mathbf{B})) = \{0\}$.

Then

$$\mathsf{ROLBP}_{9\varepsilon}(\text{SumExt}') > 2^{n-k_1-k_2-2}.$$

Next we show two different extractor constructions that utilize the first and second property, respectively. The first construction is exactly the directional affine extractors in [29]. Our main observation is that directional affine extractors can extract from the restricted class of sumset sources with the almost affine property in Theorem 11, which gives an alternative proof of the average-case lower bound in [29] (Theorem 6). Furthermore, the proof of this statement is just a simple application of leftover hash lemma [33]. We view this as a more modular proof of the lower bound result in [29].

We note that a directional affine extractor is a strictly stronger notion than a sumset extractor with the almost affine property. Indeed, given any sumset extractor, one can modify it to get a new sumset extractor so that the extractor ignores its first bit of input; however it is easy to see that the modified sumset extractor is not a directional affine extractor (see Remark 36). Thus, it could be an easier task to build sumset extractors with the almost affine property.

Our second construction is based on the interleaved-source extractor constructed in [12]. An interleaved source is a source over $\{0,1\}^{2n}$ of the form $(\mathbf{A} \circ \mathbf{B})_\pi$, where $\mathbf{A}, \mathbf{B}$ are independent sources over $\{0,1\}^n$, and π is a fixed but unknown permutation of the $2n$ bits. We observe that their extractor construction can be extended to work for for a more general class of sources: sumset sources with the non-intersecting span property. In fact, we prove a slightly more general result.

[4] A Paley graph extractor [17] with proper choice of parameters is actually a (k_1, k_2, ε)-sumset extractor for $k_1 + k_2 = (\frac{1}{2} + \gamma)n$ and negligible ε, for any constant $\gamma > 0$. (See [14, Theorem 4.2].) However, it is not known how to compute such an extractor in polynomial time.

▶ **Theorem 12.** *For every constant $\delta > 0$, there is a function* $\mathrm{ILExt} : \mathbb{F}_2^n \to \{0,1\}$ *computable in polynomial time such that for every independent sources* $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$ *which satisfy* $\mathrm{H}_\infty(\mathbf{A}) \geq (\frac{1}{3} + \delta)n$, $\mathrm{H}_\infty(\mathbf{B}) \geq (\frac{1}{3} + \delta)n$ *and* $\mathrm{H}_\infty(\mathbf{A} + \mathbf{B}) \geq (\frac{2}{3} + 2\delta)n$,

$$\mathrm{ILExt}(\mathbf{A} + \mathbf{B}) \approx_{2^{-\Omega(n)}} \mathbf{U}_1.$$

It's not hard to see that the additional entropy requirement on $\mathbf{A} + \mathbf{B}$ is implied by the non-intersecting span property, and hence we can apply Theorem 11 on the extractor in Theorem 12. Interestingly, while the two constructions are very different, they give the same average-case lower bound as in [29].

▶ **Corollary 13.** *For every constant $\delta > 0$, there exists a constant $\gamma > 0$ and a function $f : \mathbb{F}_2^n \to \{0,1\}$ computable in polynomial time such that*

$$\mathsf{ROLBP}_{2^{-\gamma n}}(f) > 2^{(1/3-\delta)n}.$$

1.5 Pseudorandomness against linear branching programs

Motivated by close connections between hardness and pseudorandomness [47], we initiate the study of obtaining pseudorandomness results against linear branching programs. Generally speaking, in the pseudorandomness problem for a function class $\mathcal{F}$, our goal is to construct a *pseudorandom distribution* which can be generated with only $r \ll n$ random bits but is indistinguishable from the n-bit uniform distribution $\mathbf{U}_n$ by any function in $\mathcal{F}$. We now formally define a hitting set generator (HSG), which is the one-sided variant of a pseudorandom generator (PRG).

▶ **Definition 14** (Hitting Set Generators)**.** *We say a set $H \subseteq \{0,1\}^n$ is a* hitting *set with error ε for a class of functions $\mathcal{F}$ (on n-bit input), if for every function $f : \{0,1\}^n \to \{0,1\}$ in $\mathcal{F}$ such that $\Pr_{x \sim \mathbf{U}_n}[f(x) = 1] \geq \varepsilon$, there exists $h \in H$ such that $f(h) = 1$. Moreover, $G : \{0,1\}^d \to \{0,1\}^n$ is called a* hitting set generator *(HSG) with error ε for a class of functions $\mathcal{F}$ if $\{G(s)\}_{s \in \{0,1\}^d}$ is a hitting set for $\mathcal{F}$, and s is called the seed length of G.*

Constructing good HSGs for (standard) read-once branching programs is a central problem in complexity theory. If one can construct an explicit HSG with seed length $O(\log(n))$ and $O(1)$ error for ROBPs of size $\mathrm{poly}(n)$, this would imply $\mathbf{RL} = \mathbf{L}$, which is a major open problem in complexity theory. Interestingly, it was recently shown [15] that HSGs suffice to even derandomize $\mathbf{BPL}$.

We note that for the above derandomization applications, it suffices to construct a HSG for oblivious ROBPs with ordered input. That is, given an n-bit input $x = (x_1, \ldots, x_n)$, the ROBPs read the bits $x_1, x_2, \ldots, x_n$ in order, regardless of what x is. For oblivious ROBPs with ordered input, the best known construction is due to Nisan [46] that has seed length $O(\log^2(n))$ (which in fact is a pseudorandom generator). However, in spite of the improvement in several restricted sub-classes of ROBPs, Nisan's result remains the best known construction in the general setting after three decades of work.

A recent research direction has been to find approaches that are completely different from Nisan's construction. In this direction, researchers considered the task of constructing PRGs (and HSGs) for a natural generalization of ROBPs called as (oblivious) unordered ROBPs for which it is known that Nisan's construction fails to work [52, 5]. An unordered ROBPs still read the bits of x in a fixed order that does not depend on x, but this order is unknown. By an impressive line of work culminating with a beautiful construction by Forbes and Kelley [27], we now have explicit PRGs with seed length $O(\log^3(n))$ for unordered

ROBPs. The general approach used to construct PRGs for this model is based on analyzing the effects of random restrictions on ROBPs and leveraging bounds on the Fourier spectrum of branching programs [49, 9].

This gives us further motivation to study pseudorandomness against *oblivious ROLBPs*, which is a further generalization of unordered ROBPs.

▶ **Definition 15** (Oblivious ROLBPs). *We say a read-once linear branching programs P on input $\mathbb{F}_2^n$ is oblivious if the nodes can be divided into layers $L_0, \ldots, L_n$ such that*

- *L_0 only contains the source, and L_n consists of all the sinks.*
- *For every $0 \leq i < n$, every edge from nodes in L_i connects to a node in L_{i+1}.*
- *For every $0 \leq i < n$, every node on L_i is labeled with the same linear query ℓ_i.*
- *$(\ell_0, \ldots, \ell_{n-1})$ is a basis of $\mathbb{F}_2^n$.*

The width *of P is defined as* $\max_{i \in [n]}(|L_i|)$.

We note that unordered ROBPs correspond to the case of $(\ell_0, \ldots, \ell_{n-1})$ being a permutation of the standard basis. Thus, Nisan's PRG construction fails to work for oblivious ROLBPs. Further, it is not clear how to use the techniques of random restriction based constructions employed for unordered ROBPs when the layers can be arbitrary linear functions. Thus, it looks like we need new ideas to obtain pseudorandomness against oblivious ROLBPs.

Our first observation is that the case of width $w = 2$ is easy since it is well known that a small-biased distribution [45, 1, 50] fools such programs.[5] This follows since small-biased distributions are invariant under full-rank linear transformations. Further, [3] proved that sum of small-biased distributions fools width-2 ROBPs that reads multiple bits. Thus, one can obtain a similar result for the linear analogue of these programs. It has been asked by Vadhan and Reingold (see [39]) whether sums of small-biased distributions can be employed to construct PRGs (or HSG) for general ROBPs. Indeed a positive answer to this question would immediately imply a PRGs (or HSG) for oblivious ROLBPs. We are not able to resolve this conjectured approach, and take a different route that we describe below.

We take an initial step towards constructing HSGs against oblivious ROLBPs of width more than 2, and focus on the sub-class of regular oblivious ROLBPs. A regular linear branching program is a linear branching program in which every non-source node has in degree 2. We note that the sub-class of regular (standard and unordered) ROBPs have been well-studied [6, 49, 4, 38]. In fact, a recent result [4] proved that obtaining a HSG against regular ROBPs would imply a HSG with similar parameters against all ROBPs.

As our main result here, we construct a *hitting set generator* with $(1 - \Omega(1))n$ seed length for *regular* oblivious ROLBPs with *constant width*.

▶ **Theorem 16.** *For every $w \in \mathbb{N}$, there is an explicit construction of HSG for regular oblivious ROLBPs of width w with seed length $(w-1) + \lceil (1 - 2^{-(w-1)})n \rceil$.*

Interestingly, our construction is based on a well-studied problem called rank-r Kakeya set [24, 36], which is a set that contains a r-dimensional affine subspace in every direction.

▶ **Definition 17.** *A set $K \subseteq \mathbb{F}_2^n$ is called a* rank-r Kakeya set *(over $\mathbb{F}_2^n$) if for every r-dimension subspace $V \subseteq \mathbb{F}_2^n$, there exists $b \in \mathbb{F}_2^n$ such that $V + b \subseteq K$.*

We prove the following theorem.

[5] This result is due to Saks and Zuckerman. See [3] for sketch of a proof.

▶ **Theorem 18.** *A rank-r Kakeya set is a hitting set for oblivious read-once regular linear BP of width* $(r+1)$.

To get an efficiently computable HSG, we take the following simple construction of rank-r Kakeya set constructed by Kopparty, Lev, Saraf and Sudan [36].

▶ **Theorem 19** ([36])**.** *For every* $r, n \in \mathbb{N}$ *s.t.* $r \leq n$*, there is an explicit construction of rank-r Kakeya set* $K_{n,r} \subseteq \mathbb{F}_2^n$ *with size at most* $2^{\lceil (1-2^{-r})n \rceil + r}$*, which is defined as follows. Let* $I_1, \ldots, I_{2^r}$ *be a parition of* $[n]$*, each having size at least* $\lfloor 2^{-r} n \rfloor$*. Then*

$$K_{n,r} = \bigcup_{t=1}^{2^r} \operatorname{span}\left(\{e_i\}_{i \notin I_i}\right).^6$$

In other words, $K_{n,r}$ *is the union of* 2^r *boolean subcubes where the* i*-th subcube contains every point* $x \in \mathbb{F}_2^n$ *such that the* x_{I_i} *is* 0*.*

To prove Theorem 16, observe that we can construct an efficient HSG with seed length $r + \lceil (1-2^{-r})n \rceil$ that uses the first r bits to select a set I_i and use the remaining $\lceil (1-2^{-r})n \rceil \geq n - |I_i|$ bits to choose a point in the subcube corresponding to I_i.

We note our approach based on Kakeya set does not seem to extend beyond regular ROLBPs. For non-regular oblivious ROLBPs, we observe that the construction in Theorem 19 is not a hitting set for width 3, because a read-once CNF $\bigwedge_{t=1}^{2^r} (\bigvee_{i \in I_t} x_i)$ always outputs 0 on $K_{n,r}$, and a read-once CNF can be computed by a width-3 ROBP.

Further, while one might hope to extend our result to larger width (for regular ROLBPs) with a better construction of Kakeya sets, we show that the construction in Theorem 19 is essentially optimal. This negative result also answers an open question in [36] (for the case of $\mathbb{F}_2^n$), where they asked whether there is a better construction of rank-r Kakeya sets than Theorem 19. This lower bounds may be of independent interest.

▶ **Theorem 20.** *Every rank-r Kakeya set over* $\mathbb{F}_2^n$ *has size at least* $2^{(1-2^{-r})(n+2)-r}$.

1.6 Subsequent Works and Future Directions

In the preliminary version of this work, we asked whether one can obtain an lower bound for ROLBPs with negligible correlation that is greater than $2^{n/3}$. This problem is recently solved by Li and Zhong [43]: they showed how to construct a directional affine extractor DAExt with $2^{-n^{\Omega(1)}}$ for $o(n)$ entropy. As proved in [29], this implies an average-case lower bound of size $2^{n-o(n)}$ and exponentially small correlation, i.e. $\mathsf{ROLBP}_{2^{-n^{\Omega(1)}}}(\mathrm{DAExt}) \geq 2^{n-o(n)}$.

In addition, an amazing recent work by Li [42] showed how to improve the entropy requirement of explicit sumset extractors to $O(\log(n))$ in the constant error regime. By Theorem 8, such an extractor implies a $2^n/\operatorname{poly}(n)$ average-case lower bound with constant correlation.

Another natural open problem raised in this work is to construct improved hitting set generators (and more ambitiously pseudorandom generators) for oblivious ROLBPs. As discussed above, one way to make progress on this question would be to show that sum of small-biased distributions are pseudorandom against (standard) oblivious ROBPs. Another direction is to see if objects from linear algebraic pseudorandomness [26] can be leveraged for derandomization in this setting.

1.7 Organization

We introduce preliminaries in Section 2. We prove Theorem 8 (and Theorem 11, which is a stronger version of Theorem 8) in Section 3. We discuss average-case lower bound results based on Theorem 11 in Section 4. We prove our results about HSGs and Kakeya sets (Theorem 18 and Theorem 20) in Section 5.

2 Preliminaries

2.1 Notation

Distributions and random variables

We sometimes abuse notation and treat distributions and random variables as the same. We always write a random variable/distribution in boldface font. Every log in this paper is of base 2 unless specified. We use $\mathrm{Supp}(\mathbf{X})$ to denote the support of a distribution. We use $\mathbf{U}_n$ to denote the uniform distribution on $\{0,1\}^n$. When $\mathbf{U}_n$ appears with other random variables in the same joint distribution, $\mathbf{U}_n$ is considered to be independent of other random variables. When there is a sequence of random variables $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_t$ in the context, for every set $S \subseteq [t]$ we use $\mathbf{X}_S$ to denote the sequence of random variables which use indices in S as subscript, i.e. $\mathbf{X}_S := \{\mathbf{X}_i\}_{i\in S}$.

Notation for $\mathbb{F}_2^n$

Throughout this paper, we treat $\mathbb{F}_2^n$ and $\{0,1\}^n$ as the same. We use $e_i \subseteq \mathbb{F}_2^n$ to denote the i-th standard basis vector, which as its i-th coordinate being 1 and other coordinates being 0. We sometimes use a vector $\ell \in \mathbb{F}_2^n$ to represent a function $f : \mathbb{F}_2^n \to \mathbb{F}_2$ defined as $f(x) = \langle \ell, x \rangle$.

2.2 Statistical Distance

▶ **Definition 21.** *Let $\mathbf{D}_1, \mathbf{D}_2$ be two distributions on the same universe Ω. The* statistical distance *between $\mathbf{D}_1$ and $\mathbf{D}_2$ is*

$$\begin{aligned}\Delta\left(\mathbf{D}_1;\mathbf{D}_2\right) &:= \max_{T\subseteq\Omega}\Big(\Pr\left[\mathbf{D}_1 \in T\right] - \Pr\left[\mathbf{D}_2 \in T\right]\Big)\\ &= \frac{1}{2}\sum_{s\in\Omega}\left|\mathbf{D}_1(s) - \mathbf{D}_2(s)\right|.\end{aligned}$$

We say $\mathbf{D}_1$ is ε-close to $\mathbf{D}_2$ if $\Delta(\mathbf{D}_1;\mathbf{D}_2) \le \varepsilon$, which is also denoted by $\mathbf{D}_1 \approx_\varepsilon \mathbf{D}_2$. When there are two joint distributions $(\mathbf{X},\mathbf{Z})$ and $(\mathbf{Y},\mathbf{Z})$ such that $(\mathbf{X},\mathbf{Z}) \approx_\varepsilon (\mathbf{Y},\mathbf{Z})$, we write $(\mathbf{X} \approx_\varepsilon \mathbf{Y}) \mid \mathbf{Z}$ for short.

Throughout this paper, we frequently use the following standard properties without explicit referencing.

▶ **Lemma 22.** *For every distribution $\mathbf{D}_1, \mathbf{D}_2, \mathbf{D}_3$ on the same universe, the following properties hold:*

- *For every function f, $\Delta\left(f(\mathbf{D}_1); f(\mathbf{D}_2)\right) \le \Delta\left(\mathbf{D}_1;\mathbf{D}_2\right)$.*
- *(Triangle inequality) $\Delta\left(\mathbf{D}_1;\mathbf{D}_3\right) \le \Delta\left(\mathbf{D}_1;\mathbf{D}_2\right) + \Delta\left(\mathbf{D}_2;\mathbf{D}_3\right)$.*
- *For any distribution $\mathbf{Z}$,*

$$\Delta\left((\mathbf{D}_1,\mathbf{Z});(\mathbf{D}_2,\mathbf{Z})\right) = \mathop{\mathbb{E}}_{z\sim\mathbf{Z}}\left[\Delta\left(\mathbf{D}_1|_{\mathbf{Z}=z};\mathbf{D}_2|_{\mathbf{Z}=z}\right)\right].$$

- *(Markov argument) For any distribution* $\mathbf{Z}$*, if* $(\mathbf{D}_1 \approx_\varepsilon \mathbf{D}_2) \mid \mathbf{Z}$*, then*

$$\Pr_{z\sim\mathbf{Z}}\left[\mathbf{D}_1|_{\mathbf{Z}=z} \approx_{\sqrt{\varepsilon}} \mathbf{D}_2|_{\mathbf{Z}=z}\right] \geq 1-\sqrt{\varepsilon}$$

2.3 Conditional Min-entropy

In this work we use a fine-grained definition of conditional min-entropy called *average conditional min-entropy* which was introduced in [22].

▶ **Definition 23** ([22])**.** *For a joint distribution* $(\mathbf{X},\mathbf{Z})$*, the* average conditional min-entropy *of* $\mathbf{X}$ *given* $\mathbf{Z}$ *is*

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X}\mid\mathbf{Z}) := -\log\left(\mathop{\mathbb{E}}_{z\sim\mathbf{Z}}\left[\max_x(\Pr\left[\mathbf{X}=x\mid\mathbf{Z}=z\right])\right]\right).$$

For average conditional min-entropy we have the following nice property called *chain rule*:

▶ **Lemma 24** ([22])**.** *Let* $\mathbf{X},\mathbf{Y},\mathbf{Z}$ *be (correlated) random variables such that* $\mathrm{Supp}(\mathbf{Y}|_{\mathbf{Z}=z}) \leq 2^\lambda$ *for every* $z\in\mathrm{Supp}(\mathbf{Z})$*. Then*

$$\widetilde{\mathrm{H}}_\infty(\mathbf{X}\mid(\mathbf{Y},\mathbf{Z})) \geq \widetilde{\mathrm{H}}_\infty((\mathbf{X},\mathbf{Y})\mid\mathbf{Z}) - \lambda \geq \widetilde{\mathrm{H}}_\infty(\mathbf{X}\mid\mathbf{Z}) - \lambda.$$

The average conditional min-entropy can be converted into worst-case conditional min-entropy with the following lemma.

▶ **Lemma 25** ([22, 44])**.** *Let* $\mathbf{X},\mathbf{Z}$ *be (correlated) random variables. For every* $\varepsilon>0$*,*

$$\Pr_{z\sim\mathbf{Z}}\left[\mathrm{H}_\infty(\mathbf{X}|_{\mathbf{Z}=z}) \geq \widetilde{\mathrm{H}}_\infty(\mathbf{X}\mid\mathbf{Z}) - \log(1/\varepsilon)\right] \geq 1-\varepsilon.$$

2.4 Extractors

First we define a more general form of seeded extractors. (In the standard definition of seeded extractor, we consider $\mathbf{Y}$ to be the uniform distribution over $\mathcal{S}$.)

▶ **Definition 26.** *Let* $\mathcal{X},\mathcal{S}$ *be two finite sets. Let* $\mathbf{Y}$ *be a distribution over* $\mathcal{S}$*. We say* $\mathrm{Ext}:\mathcal{X}\times\mathcal{S}\to\{0,1\}^m$ *is a* (k,ε)*-extractor with seed* $\mathbf{Y}$ *if for every distribution* $\mathbf{X}\in\mathcal{X}$ *independent of* $\mathbf{Y}$ *such that* $\mathrm{H}_\infty(\mathbf{X})\geq k$*,*

$$\mathrm{Ext}(\mathbf{X},\mathbf{Y}) \approx_\varepsilon \mathbf{U}_m.$$

Furthermore, we say Ext *is strong in* $g(\mathbf{Y})$ *for some deterministic function* g *if*

$$(\mathrm{Ext}(\mathbf{X},\mathbf{Y}) \approx_\varepsilon \mathbf{U}_m) \mid g(\mathbf{Y}).$$

When Ext *is strong in* $\mathbf{Y}$ *we simply say* Ext *is strong.*

For strong seeded extractor we have the following standard lemma.

▶ **Lemma 27.** *Suppose* $\mathrm{Ext}:\mathcal{X}\times\mathcal{S}\to\{0,1\}^m$ *is a* (k,ε)*-strong extractor with seed* $\mathbf{Y}$*, where* $\mathbf{Y}$ *is the uniform distribution over a set* $S\subseteq\mathcal{S}$*. Then for every* $\mathbf{Y}'$ *such that* $\mathrm{Supp}(\mathbf{Y}')\subseteq S$ *and* $\mathrm{H}_\infty(\mathbf{Y}')\geq\mathrm{H}_\infty(\mathbf{Y})-\Delta$*,* Ext *is a* $(k,2^\Delta\varepsilon)$*-strong extractor with seed* $\mathbf{Y}'$*.*

We need the following form of *leftover hash lemma*. This is more general than the original lemma in [33], but is also standard in the literature. (See, e.g., [53, Problem 6.3].)

▶ **Lemma 28** (Leftover Hash Lemma [33]). *Consider any $h : \{0,1\}^n \times \mathcal{S} \to \{0,1\}^m$ and any distribution $\mathbf{Y} \in \mathcal{S}$ such that for every distinct $x_1, x_2 \in \{0,1\}^n$, $\Pr_{y \sim \mathbf{Y}}[h(x_1, y) = h(x_2, y)] \leq (1+\varepsilon)2^{-m}$. (We say h is ε-almost universal over randomness $\mathbf{Y}$ if h and $\mathbf{Y}$ satisfy the condition above.) Then h is a strong $(m + \log(1/\varepsilon), \sqrt{\varepsilon/2})$-extractor with seed $\mathbf{Y}$.*

We will also use the following lemma for seeded extractors on conditional min-entropy from [53, Problem 6.8]. We need a more general form which works for the general seeded extractors defined above. We include a proof in Appendix C for completeness. (In the standard form of the following lemma, $\mathbf{Y}$ is a uniform over $\mathcal{S}$, and $\mathcal{X}_e = \mathcal{X}$ for every e.)

▶ **Lemma 29.** *Let $(\mathbf{X}, \mathbf{Y}, \mathbf{E})$ be a joint distribution such that $\mathbf{X} \in \mathcal{X}$ and $\mathbf{Y} \in \mathcal{S}$ are independent conditioned on $\mathbf{E}$, and $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{E}) \geq k$. Let $\mathrm{Ext} : \mathcal{X} \times \mathcal{S} \to \{0,1\}^m$ be a function which satisfies the following conditions for an error parameter $\varepsilon > 0$ and a deterministic function g: for every $e \in \mathrm{Supp}(\mathbf{E})$, there exists a set $\mathcal{X}_e \subseteq \mathcal{X}$ with size at least 2^{k+1} such that Ext when restricted to the domain $\mathcal{X}_e \times \mathcal{S}$ is a (k, ε)-extractor with seed $\mathbf{Y}|_{\mathbf{E}=e}$ and is strong in $g(e, \mathbf{Y})$. Then*

$$(\mathrm{Ext}(\mathbf{X}, \mathbf{Y}) \approx_{3\varepsilon} \mathbf{U}_m) \mid (\mathbf{E}, g(\mathbf{E}, \mathbf{Y})).$$

3 Linear BP lower bounds based on sumset extractors

In this section, we prove Theorem 11 that we restate below. We note that Theorem 8 follows as a special case of this theorem.

▶ **Theorem 11** (restated). *Let $\mathrm{SumExt}' : \mathbb{F}_2^n \to \{0,1\}$ be a function such that $\mathrm{SumExt}'(\mathbf{A} + \mathbf{B}) \approx_\varepsilon \mathbf{U}_1$ for any independent distributions $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$ which satisfy $\mathrm{H}_\infty(\mathbf{A}) \geq k_1, \mathrm{H}_\infty(\mathbf{B}) \geq k_2$ and the following two additional properties.*

- *$\mathbf{B}$ is* almost affine*: the span of $\mathrm{Supp}(\mathbf{B})$ is of dimension $\leq k_2 + 1$.*
- *$\mathbf{A}$ and $\mathbf{B}$ have* non-intersecting span*: $\mathrm{span}(\mathrm{Supp}(\mathbf{A})) \cap \mathrm{span}(\mathrm{Supp}(\mathbf{B})) = \{0\}$.*

Then

$$\mathsf{ROLBP}_{9\varepsilon}(\mathrm{SumExt}') > 2^{n-k_1-k_2-2}.$$

We first discuss the main ideas behind the proof before formally proving it. Given a read-once linear BP $P : \mathbb{F}_2^n \to \{0,1\}$ and any $b \in \{0,1\}$, the uniform distribution over the pre-image $P^{-1}(b)$ corresponds to the uniform distribution over all the computation path from the source s to the sink labeled b. For every edge e, whether a computation path pass goes through e and ends at a sink labeled b can be divided into two events: whether a path starting from s would reach e, and whether a path starting from e would end at a sink labeled b. The strongly read-once property guarantees that we can divide $\mathbb{F}_2^n$ into two complemented subspaces V_A, V_B such that the first event is determined by the projection of the input $x \in \mathbb{F}_2^n$ on V_A, and the second event is determined by the projection of x on V_B. Given a uniform input $\mathbf{X} \in \mathbb{F}_2^n$, the two projections are independent. Therefore, conditioned on the computation path passing through e and end at a sink labeled b, $\mathbf{X}$ can be written as the sum of two independent sources $\mathbf{A} + \mathbf{B}$, where $\mathrm{Supp}(\mathbf{A}) \subseteq V_A$ and $\mathrm{Supp}(\mathbf{B}) \subseteq V_B$. It remains to choose a cut E such that for every choice of $e \in E$, the two sources $\mathbf{A}, \mathbf{B}$ stated above both have enough entropy.

We formalize the ideas above as the following structural lemma:

▶ **Lemma 30.** *Let $\mathbf{X}$ be a uniform random variable over $\mathbb{F}_2^n$. For every strongly read-once linear BP $f : \mathbb{F}_2^n \to \{0,1\}$ of size s and every $d \in [n]$, there exists a random variable $\mathbf{E}$, and random variables $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$, s.t.*

- $\mathbf{E}$ *has support size at most* $2s$.
- $\mathbf{X} = \mathbf{A} + \mathbf{B}$
- *For every* $e \in \mathrm{Supp}(\mathbf{E})$, *define* $\mathbf{A}_e = \mathbf{A}|_{\mathbf{E}=e}$ *and* $\mathbf{B}_e = \mathbf{B}|_{\mathbf{E}=e}$. *Then we have*
  - $\mathbf{A}_e$ *and* $\mathbf{B}_e$ *are independent.*
  - $\mathbf{B}_e$ *is uniform over an affine subspace* V_e^B *of dimension* d
  - *There exists a complemented subspace* V_e^A *of* V_e^B *such that* $\mathbf{A}_e \in V_e^A$
- *There is a deterministic function* g *s.t.* $g(\mathbf{E}, \mathbf{B}) = f(\mathbf{X})$.

Proof. We show that there exist some functions E, A, B s.t. $\mathbf{E} = E(\mathbf{X}), \mathbf{A} = A(\mathbf{X}), \mathbf{B} = B(\mathbf{X})$ satisfy the above claim. Fix any $x \in \mathbb{F}_2^n$. Consider the computation path of x, and let v be the first node on this path which satisfies that $\dim(\mathsf{Post}_v) \le d$. Note that v is well-defined because the last node w on this path satisfies $\dim(\mathsf{Post}_w) = 0 \le d$. Then we define $E(x) := (u \to v)$ to be the edge right before v in this path. (If v is the source, we define u to be a dummy node $\perp$, and define $\mathsf{Pre}_\perp = \{0\}$.) First we claim that $\dim(\mathsf{Pre}_u) \le n - d$. If $u = \perp$ then the claim is trivially true. Otherwise, observe that $\dim(\mathsf{Post}_u) \ge d + 1$ by the definition of v, and by the strongly read-once property we have

$$\dim(\mathsf{Pre}_u) \le n + \dim(\mathsf{Pre}_u \cap \mathsf{Post}_u) - \dim(\mathsf{Post}_u) \le n + 1 - (d+1) = n - d.$$

Observe that $\mathsf{Pre}_u \cap \mathsf{Post}_v = \{0\}$ by the strongly read-once property. Now we choose an arbitrary basis $(b_1, b_2, \ldots, b_n)$ of $\mathbb{F}_2^n$ such that $\mathrm{span}(\{b_i\}_{1 \le i \le \dim(\mathsf{Pre}_u)}) = \mathsf{Pre}_u$ and $\mathrm{span}(\{b_{n-i}\}_{0 \le i < \dim(\mathsf{Post}_v)}) = \mathsf{Post}_v$. Define $\mathsf{Pre}'_u = \mathrm{span}(\{b_i\}_{1 \le i \le n-d})$ and $\mathsf{Post}'_v = \mathrm{span}(\{b_i\}_{n-d < i \le n})$. Note that $\mathsf{Pre}_u \subseteq \mathsf{Pre}'_u$, $\mathsf{Post}_v \subseteq \mathsf{Post}'_v$ and Pre'_u and Post'_v are complemented subspaces. Then define $(A(x), B(x))$ to be the unique pair in $(\mathsf{Post}'_v)^\perp \times (\mathsf{Pre}'_u)^\perp$ s.t. $A(x) + B(x) = x$. It remains to prove that $\mathbf{E} = E(\mathbf{X}), \mathbf{A} = A(\mathbf{X}), \mathbf{B} = B(\mathbf{X})$ satisfy our claim.

First it's easy to see that the support size of $\mathbf{E}$ is upper bounded by $2s$: if the source s satisfies $\dim(\mathsf{Post}_s) \le d$, v is always the source s and $\mathbf{E}$ has support size 1; otherwise $\mathbf{E}$ is an edge in the branching program, and there are at most $2s$ choices. Moreover, $\mathbf{X} = \mathbf{A} + \mathbf{B}$ by definition of A and B. To prove the remaining two claims, consider any possible fixing $\mathbf{E} = e := (u \to v)$. Let $(A_e(x), B_e(x))$ denote the unique pair in $(\mathsf{Post}'_v)^\perp \times (\mathsf{Pre}'_u)^\perp$ s.t. $A_e(x) + B_e(x) = x$. We claim that there exists a set $S \subseteq (\mathsf{Post}'_v)^\perp$ so that $E(x) = e$ if and only if $A_e(x) \in S$. This implies that $(\mathbf{A}, \mathbf{B})|_{\mathbf{E}=e}$ is exactly the uniform distributions over $S \times (\mathsf{Pre}'_u)^\perp$, which satisfies the third claim by taking $V_e^B = (\mathsf{Pre}'_u)^\perp$ and $V_e^A = (\mathsf{Post}'_v)^\perp$. To prove this claim, observe that whether $E(x) = e$ can be decided by the following procedure. We follow the computation path of x, but stop and answer "NO" if we reach any node w such that either w cannot reach u (so that $E(x)$ can never be e regardless of the remaining queries) or $\dim(\mathsf{Post}_w) \le d$ (so that $E(x)$ would be the edge ending at w instead of e). Otherwise, if we reach the edge e we stop and answer "YES" . Observe that every linear query ℓ we made in this procedure is in Pre_u. Moreover, for every such query, $\ell(x) = \ell(A_e(x)) + \ell(B_e(x)) = \ell(A_e(x))$ because $B_e(x) \in (\mathsf{Pre}'_u)^\perp \subseteq (\mathsf{Pre}_u)^\perp$. Therefore, the event $E(x) = e$ is completely determined by $A_e(x)$, which proves our claim. Finally, observe that conditioned on $E(x) = e$, the value of $f(x)$ is determined by queries in Post_v, and every such query ℓ satisfies that $\ell(x) = \ell(A_e(x)) + \ell(B_e(x)) = \ell(B_e(x)) = \ell(B(x))$ because $A_e(x) \in (\mathsf{Post}'_v)^\perp \subseteq (\mathsf{Post}_v)^\perp$. Therefore by choosing $g(e, \cdot)$ to be the subprogram of f starting at v, the last condition is also satisfied. ◀

Now we are ready to prove Theorem 11.

Proof of Theorem 11. Let SumExt' be a function which satisfies the conditions in Theorem 11 with parameters (k_1, k_2, ε), and let $f : \mathbb{F}_2^n \to \{0,1\}$ be any strongly read-once linear BP of size $s = 2^{n-k_1-k_2-2}$. Let $\mathbf{X}$ be a uniform random variable over $\mathbb{F}_2^n$. We want to show that

$$(\mathrm{SumExt}'(\mathbf{X}), f(\mathbf{X})) \approx_{9\varepsilon} (\mathbf{U}_1, f(\mathbf{X})), \tag{1}$$

which would imply $\Pr_{x\sim\mathbf{X}}\left[f(x) = \mathrm{SumExt}'(x)\right] \leq \frac{1}{2} + 9\varepsilon$ for every f of size s, and hence $\mathsf{ROLBP}_{9\varepsilon}(\mathrm{SumExt}') > s$.

Let $\mathbf{E}, \mathbf{A}, \mathbf{B}$ be the random variables depending on $\mathbf{X}$ as in Lemma 30, by taking $d = k_2+1$. Recall that $\mathbf{E}, \mathbf{A}, \mathbf{B}$ have the following properties:

- $\mathbf{E}$ has support size at most $2s$.
- $\mathbf{X} = \mathbf{A} + \mathbf{B}$
- For every $e \in \mathrm{Supp}(\mathbf{E})$, define $\mathbf{A}_e = \mathbf{A}|_{\mathbf{E}=e}$ and $\mathbf{B}_e = \mathbf{B}|_{\mathbf{E}=e}$. Then we have
  - $\mathbf{A}_e$ and $\mathbf{B}_e$ are independent.
  - There exist complemented subspaces V_e^A, V_e^B of dimension $n-d$ and d such that $\mathbf{B}_e$ is uniform over V_e^B and $\mathbf{A}_e \in V_e^A$.
- There is a deterministic function g s.t. $g(\mathbf{E}, \mathbf{B}) = f(\mathbf{X})$.

Therefore we can rewrite Equation (1) as

$$(\mathrm{SumExt}'(\mathbf{A}+\mathbf{B}) \approx_{9\varepsilon} \mathbf{U}_1) \mid g(\mathbf{E}, \mathbf{B}). \tag{2}$$

Consider the function $\mathrm{Ext} : (\mathbb{F}_2^n)^2 \to \{0,1\}$ defined as $\mathrm{Ext}(a,b) = \mathrm{SumExt}'(a+b)$. We claim that for every $e \in \mathrm{Supp}(\mathbf{E})$, Ext restricted on the domain $V_e^A \times \mathbb{F}_2^n$ is a $(k_1, 3\varepsilon)$-extractor with seed $\mathbf{B}_e$ and is strong in $g(e, \mathbf{B}_e)$. This would imply Equation (2) because of the following. Observe that

$$\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid \mathbf{E}) = \widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid (\mathbf{B}, \mathbf{E})) \geq \widetilde{\mathrm{H}}_\infty((\mathbf{A}, \mathbf{B}) \mid \mathbf{E}) - d \geq (n - \log(2s)) - d \geq k_1,$$

where the first equality is by the fact that $\mathbf{A}$ and $\mathbf{B}$ are independent conditioned on $\mathbf{E}$, and the first and second inequalities are by chain rule (Lemma 24). Furthermore, we can w.l.o.g. assume that $k_1 + k_2 \leq n-2$ (since otherwise the bound is trivial), and this would imply $|V_e^A| = 2^{n-d} \geq 2^{k_1+1}$. Therefore we can apply Lemma 29 on Ext to get Equation (2).

Next we prove the claim. Let $\mathbf{A}' \in V_e^A$ be any distribution such that $\mathrm{H}_\infty(\mathbf{A}') \geq k_1$. By definition of SumExt', we have that for every random variable $\mathbf{B}' \in V_e^B$ such that $\mathrm{H}_\infty(\mathbf{B}') \geq \dim(V_e^B) - 1 = k_2$,

$$\mathrm{SumExt}'(\mathbf{A}' + \mathbf{B}') \approx_\varepsilon \mathbf{U}_1.$$

In other words, the function $\mathrm{Ext}' : V_e^B \times \mathbb{F}_2^n \to \{0,1\}$ defined as $\mathrm{Ext}'(b,a) = \mathrm{SumExt}'(a+b)$ is a (k_2, ε)-extractor with seed $\mathbf{A}'$. By chain rule, $\widetilde{\mathrm{H}}_\infty(\mathbf{B}_e \mid g(e, \mathbf{B}_e)) \geq \mathrm{H}_\infty(\mathbf{B}_e) - 1 = k_2$. Therefore, by Lemma 29 we can conclude that

$$(\mathrm{SumExt}'(\mathbf{A}' + \mathbf{B}_e) \approx_{3\varepsilon} \mathbf{U}_1) \mid g(e, \mathbf{B}_e),$$

and this is exactly what we claimed. ◀

4 Average-case lower bound with negligible error

As we discussed in the introduction, Theorem 8 only implies average-case lower bound with polynomially small error because it is not known how to construct a (k_1, k_2, ε)-sumset extractor for entropy $k_1 + k_2 < n$ with negligible error ε. However, we proved a stronger theorem, Theorem 11, which says that we only need an extractor for sumset sources $\mathbf{A} + \mathbf{B}$ with two additional properties:

- $\mathbf{B}$ is almost affine: $\mathrm{Supp}(\mathbf{B})$ is contained in a linear subspace of dimension $\mathrm{H}_\infty(\mathbf{B}) + 1$, and
- $\mathbf{A}$ and $\mathbf{B}$ have non-intersecting span: $\mathrm{span}(\mathrm{Supp}(\mathbf{A})) \cap \mathrm{span}(\mathrm{Supp}(\mathbf{B})) = \{0\}$.

In this section we will see that we only need either of the two properties to prove a $2^{(1/3-\gamma)n}$ average-case lower bound with exponentially small error.

4.1 Sumset extractors for almost affine source

In this section, we show that a directional affine extractor can work for a sumset source $\mathbf{A} + \mathbf{B}$ as long as $\mathbf{B}$ is almost affine. The proof is simply an application of leftover hash lemma (Lemma 28).

▶ **Lemma 31.** *Let* $\mathrm{DAExt} : \mathbb{F}_2^n \to \{0,1\}$ *be any* $(d, \varepsilon/2)$*-directional affine extractor. Then for any* $\mathbf{B} \in \mathbb{F}_2^n$ *which is uniform over an affine subspace of dimension* d*, and any* $\mathbf{A} \in \mathbb{F}_2^n$ *independent of* $\mathbf{B}$ *such that* $\mathrm{H}_\infty(\mathbf{A}) \geq \log(1/\varepsilon) + 1$*,*

$$(\mathrm{DAExt}(\mathbf{A} + \mathbf{B}) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid \mathbf{B}.$$

Proof. Observe that for every distinct $a_1, a_2 \in \mathbb{F}_2^n$,

$$\Pr_{b \sim \mathbf{B}}[\mathrm{DAExt}(a_1 + b) = \mathrm{DAExt}(a_2 + b)] = \Pr_{b \sim \mathbf{B}}[(\mathrm{DAExt}(a_1 + b) + \mathrm{DAExt}(a_2 + b)) = 0] \leq \frac{1+\varepsilon}{2},$$

by definition of $(d, \varepsilon/2)$-directional affine extractor. This means the function $h(a, b) = \mathrm{DAExt}(a + b)$ is ε-almost universal over randomness $\mathbf{B}$. By leftover hash lemma (Lemma 28), h is a $(\log(1/\varepsilon)+1, \sqrt{\varepsilon/2})$-strong extractor with seed $\mathbf{B}$. In other words, for every distribution $\mathbf{A} \in \mathbb{F}_2^n$ independent of $\mathbf{B}$ such that $\mathrm{H}_\infty(\mathbf{A}) \geq \log(1/\varepsilon) + 1$,

$$(\mathrm{DAExt}(\mathbf{A} + \mathbf{B}) \approx_{\sqrt{\varepsilon/2}} \mathbf{U}_1) \mid \mathbf{B}.$$ ◀

▶ **Corollary 32.** *Let* $\mathrm{DAExt} : \mathbb{F}_2^n \to \{0,1\}$ *be any* $(d, \varepsilon/2)$*-directional affine extractor. Then for any independent distributions* $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$ *such that* $\mathrm{H}_\infty(\mathbf{A}) \geq \log(1/\varepsilon)+1$*,* $\mathrm{H}_\infty(\mathbf{B}) \geq d-1$ *and* $\dim(\mathrm{span}(\mathrm{Supp}(\mathbf{B}))) \leq d$*,*

$$\mathrm{DAExt}(\mathbf{A} + \mathbf{B}) \approx_{3\sqrt{\varepsilon/2}} \mathbf{U}_1.$$

Proof. Let V be a linear subspace of dimension d such that $\mathrm{Supp}(\mathbf{B}) \subseteq V$, and let $\mathbf{B}'$ denote the uniform distribution over V. Define $\mathrm{Ext} : (\mathbb{F}_2^n)^2 \to \{0,1\}$ to bet $\mathrm{Ext}(a, b) = \mathrm{DAExt}(a+b)$. By Lemma 31, Ext is a strong $(\log(1/\varepsilon) + 1, \sqrt{\varepsilon/2})$-extractor with seed $\mathbf{B}'$. Since $\mathrm{H}_\infty(\mathbf{B}) \geq d - 1 = \mathrm{H}_\infty(\mathbf{B}') - 1$, by Lemma 27, Ext is a strong $(\log(1/\varepsilon) + 1, 3\sqrt{\varepsilon/2})$-extractor with seed $\mathbf{B}$, which is exactly what we want to prove. ◀

Apply Theorem 11 on Corollary 32 by taking $k_1 = \log(1/\varepsilon) + 1$ and $k_2 = d - 1$, we get an alternative proof of [29, Theorem 17].

▶ **Theorem 33.** *If* DAExt *is a* $(d, \varepsilon/2)$*-directional affine extractor, then*

$$\mathsf{ROLBP}_{27\sqrt{\varepsilon/2}}(\mathrm{DAExt}) \geq \varepsilon 2^{n-d-1}.$$

▶ Remark 34. The error in the above theorem is worse than [29, Theorem 17] by a constant factor 27, but we note that our proof above is just a modular presentation of the proof in [29, Theorem 17], and the factor 27 can be removed by a more careful analysis of this specific construction. That is, in the proof of Theorem 11 we actually need an affine source with 1-bit leakage instead of an almost affine source, so a factor 9 incurred by arguments related to average conditional min-entropy is unnecessary. Second, a seeded extractor based on leftover hash lemma can in fact work for average conditional min-entropy without any loss (see [22]), so we can remove another factor 3.

Recall that [29, Theorem 15] shows that there is an explicit $(d + \log(1/\varepsilon), \varepsilon/2)$-directional affine extractor. This implies the following corollary:

▶ **Corollary 35.** *For every constant* $\gamma > 0$*, there exists an explicit function* $\mathrm{DAExt} : \mathbb{F}_2^n \to \{0,1\}$ *such that*

$$\mathsf{ROLBP}_{2^{-\gamma n}}(\mathrm{DAExt}) > 2^{(1/3-2\gamma)n-O(1)}.$$

▶ Remark 36. We note that while directional affine extractors imply sumset extractors with the additional "almost affine" restriction, the converse is not true. For example, if we take any sumset extractor Ext on n-bit input, and construct a new function Ext' on $(n+1)$-bit input which simply ignore the first bit and compute Ext on the last n bits, then Ext' is still a sumset extractor, but Ext' cannot be a directional affine extractor, because the shift $a = (1, 0, \ldots, 0, 0) \in \mathbb{F}_2^{n+1}$ would make $\mathrm{Ext}'(\mathbf{X} + a) + \mathrm{Ext}'(\mathbf{X}) = 0$ for every source $\mathbf{X}$.

4.2 Sumset extractors for non-intersecting span

To utilize the non-intersecting span property, we show that the interleaved-source extractor in [12] can be extended to work for the sum of two independent sources $\mathbf{A}, \mathbf{B}$ as long as both $\mathbf{A}, \mathbf{B}$ has entropy rate greater than $1/3$ and $\mathbf{A} + \mathbf{B}$ has entropy rate greater than $2/3$. Formally, we prove the following theorem which extends Theorem 8.1 in [12].[7]

▶ **Theorem 12** (restated). *For every constant* $\delta > 0$*, there exists constants* $\gamma, \tau > 0$ *and an explicit function* $\mathrm{ILExt} : \mathbb{F}_2^n \to \{0,1\}^m$*,* $m = \gamma n$*, such that for any two independent sources* $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$ *which satisfies that*

- $\mathrm{H}_\infty(\mathbf{A}), \mathrm{H}_\infty(\mathbf{B}) \geq (\frac{1}{3} + \delta)n$
- $\mathrm{H}_\infty(\mathbf{A} + \mathbf{B}) \geq (\frac{2}{3} + 2\delta)n$

we have

$$\mathrm{ILExt}(\mathbf{A} + \mathbf{B}) \approx_{2^{-\tau n}} \mathbf{U}_m.$$

This theorem also implies a roughly $2^{n/3}$ average-case lower bound:

▶ **Corollary 37.** *For every constant* $\delta > 0$*, there exists a constant* $\tau > 0$ *and an explicit function* $\mathrm{ILExt} : \mathbb{F}_2^n \to \{0,1\}$ *such that*

$$\mathsf{ROLBP}_{2^{-\tau n}}(\mathrm{ILExt}) > 2^{(1/3-2\delta)n}.$$

[7] Note that we also improve the error from $2^{n^{-\Omega(1)}}$ in [12] to $2^{-\Omega(n)}$. This improvement comes from a better construction of affine correlation breakers in more recent works [7, 13].

Proof. Let ILExt be the extractor in Theorem 12 with parameter $\delta > 0$, and let $\tau > 0$ be the corresponding constant in Theorem 12. (The output of ILExt is truncated to 1 bit.) Observe that given any two independent sources $\mathbf{A}, \mathbf{B} \in \mathbb{F}_2^n$, $\mathrm{span}(\mathrm{Supp}(\mathbf{A})) \cap \mathrm{span}(\mathrm{Supp}(\mathbf{B})) = \{0\}$ implies that for every $x \in \mathrm{Supp}(\mathbf{A}+\mathbf{B})$, there is a unique pair $(a,b) \in \mathrm{Supp}(\mathbf{A}) \times \mathrm{Supp}(\mathbf{B})$ such that $a+b=x$, where a is the projection of x on $\mathrm{span}(\mathrm{Supp}(\mathbf{A}))$ and b is the projection of x on $\mathrm{span}(\mathrm{Supp}(\mathbf{B}))$. This implies $\mathrm{H}_\infty(\mathbf{A}+\mathbf{B}) = \mathrm{H}_\infty(\mathbf{A}) + \mathrm{H}_\infty(\mathbf{B})$. Therefore, we can apply Theorem 11 on ILExt by taking $k_1 = k_2 = (1/3+\delta)n$ and conclude that

$$\mathsf{ROLBP}_{2^{-\tau n}}(\mathrm{ILExt}) > 2^{(1/3-2\delta)n}.$$ ◀

Before we formally prove Theorem 12, first we recall the construction of the interleaved-source extractor in [12]. The construction can be viewed as an affine variant of the three-source extractor in [18], which is as follows. Suppose we have three independent sources $\mathbf{X}, \mathbf{Y}, \mathbf{Z} \in \{0,1\}^n$ with min-entropy δn. The first step is to apply a *somewhere random condenser* on $\mathbf{Z}$ to get $t = O(1)$ correlated sources $(\mathbf{S}_1, \ldots, \mathbf{S}_t) \in (\{0,1\}^{d_1})^t$ such that there exists an unknown $i^* \in [t]$ for which $\mathbf{S}_{i^*}$ is guaranteed to have min-entropy $(1-\beta)d_1$, for some small enough constant $\beta > 0$. The second step is to compute $\mathbf{R}_i = \mathrm{Ext}(\mathbf{Y}, \mathbf{S}_i)$ for every $i \in [t]$ with some strong seeded extractor Ext. This makes sure that $\mathbf{R}_{i^*}$ is close to uniform, but we still don't know i^*, and $\mathbf{R}_{i^*}$ is correlated with other $\mathbf{R}_i$. To fix this problem, the final step is to apply a *correlation breaker* to "break the correlation" between $(\mathbf{R}_1, \ldots, \mathbf{R}_t)$ with the help of the remaining independent source $\mathbf{X}$, and merge them into a single uniform string by computing their parity.

In the interleaved source/sumset source setting, we are only given one source $\mathbf{A}+\mathbf{B}$. To apply the above three-source extractor construction, [12] takes a prefix of $\mathbf{A}+\mathbf{B}$ of length n_1, denoted by $\mathbf{A}_0+\mathbf{B}_0$, to play the role of $\mathbf{Z}$ in the above construction. Then $\mathbf{A}$ and $\mathbf{B}$ would play the roles of $\mathbf{X}$ and $\mathbf{Y}$ in the above construction respectively. In fact, since we do not have access to $\mathbf{A}$ and $\mathbf{B}$ separately, we would actually use $\mathbf{A}+\mathbf{B}$ to play the role of both $\mathbf{X}$ and $\mathbf{Y}$. We would take Ext to be a *strong linear seeded extractor*, and the correlation breaker to be an *affine correlation breaker*, so that $\mathbf{A}+\mathbf{B}$ can play the role of $\mathbf{B}$ and $\mathbf{A}$ respectively in the analysis. We will see the definitions of these primitives later.

To see why taking $\mathbf{Z}$ to be the prefix $\mathbf{A}_0+\mathbf{B}_0$ could possibly work, first observe that in the above construction, we only need a *block source* $(\mathbf{Z}, \mathbf{X})$ and another independent source $\mathbf{Y}$, instead of three independent sources. That is, we only need $(\mathbf{X}, \mathbf{Z})$ to be independent of $\mathbf{Y}$, and $\mathbf{X}$ to have enough entropy conditioned on $\mathbf{Z}$, because we would fix $\mathbf{Z}$ after the first step in the analysis. Therefore, as long as $\mathbf{A}_0$ has enough entropy, we can fix $\mathbf{B}_0$ in the first step, and $(\mathbf{A}, \mathbf{A}_0+\mathbf{B}_0)$ would become independent of $\mathbf{B}$. For the analysis to work, we need to make sure that after fixing both $\mathbf{A}_0$ and $\mathbf{B}_0$, both $\mathbf{A}$ and $\mathbf{B}$ still have enough entropy. Therefore, we need $\mathrm{H}_\infty(\mathbf{A}), \mathrm{H}_\infty(\mathbf{B})$ to be greater than n_1. At the same time, we also need n_1 to be large enough so that $\mathbf{A}_0$ contains enough entropy. (Note that $\mathbf{A}, \mathbf{B}$ are symmetric in the construction, so the analysis can also work if $\mathbf{B}_0$ contains enough entropy instead.) It turns out that it suffices to take $n_1 = n/3$ if $\mathbf{A}+\mathbf{B}$ is an interleaved source, and this is the only place where [12] needs $\mathbf{A}+\mathbf{B}$ to be an interleaved source. We observe that what we actually need in the analysis is that $\mathrm{H}_\infty(\mathbf{A}+\mathbf{B})$ is larger than $2n/3$.

Next we introduce the primitives that we mentioned in the above construction. First we define somewhere random sources and somewhere random condenser.

▶ **Definition 38.** *We say* $(\mathbf{R}_1, \ldots, \mathbf{R}_t) \in (\{0,1\}^n)^t$ *is an* elementary somewhere random k-source *if there exists* $i \in [t]$ *s.t.* $\mathrm{H}_\infty(\mathbf{R}_i) \geq k$. *A* somewhere random k-source *is a convex combination of elementary somewhere random k-sources.*

▶ **Definition 39.** *We say* $\mathrm{SRCon} : \{0,1\}^n \to (\{0,1\}^m)^t$ *is a* $(\alpha_1 \to \alpha_2, \varepsilon)$-somewhere random condenser *if for every* $\mathbf{X} \in \{0,1\}^n$ *such that* $\mathrm{H}_\infty(\mathbf{X}) \geq \alpha_1 n$, $\mathrm{SRCon}(\mathbf{X})$ *is* ε*-close to a somewhere random* $(\alpha_2 m)$*-source.*

▶ **Lemma 40** ([2, 48, 55])**.** *For every constants* $\delta, \beta > 0$*, there exist constants* $t \in \mathbb{N}$ *and* $\gamma_1, \gamma_2 > 0$ *such that the following holds. For every large enough* $n \in \mathbb{N}$*, there exists an explicit* $(\delta \to 0.99, \varepsilon)$*-somewhere random condenser* $\mathrm{SRCon} : \{0,1\}^n \to (\{0,1\}^{\gamma_1 n})^t$ *where* $\varepsilon = 2^{-\gamma_2 n}$.

The second primitive we need is a strong linear seeded extractors. We say a seeded extractor $\mathrm{Ext} : \mathcal{X} \times \mathcal{S} \to \{0,1\}^n$ is linear if for every $s \in \mathcal{S}$, $\mathrm{Ext}(\cdot, s)$ is a linear function. We need a linear seeded extractor with good dependence on the error, which can be constructed with a composition of GUV condenser [30] and leftover hash lemma [33]. (See, e.g., [7] for a proof.)

▶ **Lemma 41.** *For every* m *and* $\varepsilon > 0$*, and every* $d \geq 2m + 8\log(n/\varepsilon) + O(1)$*, there is an explicit* (k, ε)*-strong linear extractor* $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *with seed* $\mathbf{U}_d$*, where* $k \geq m + 2\log(1/\varepsilon)$.

Specifically, we want to choose ε small enough to get a seeded extractor that works for high-entropy seed.

▶ **Lemma 42.** *For every* $d \geq 200\log(n)$*, there is an explicit function* $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d/3}$*, such that for every distribution* $\mathbf{Y} \in \{0,1\}^d$ *which satisfies* $\mathrm{H}_\infty(\mathbf{Y}) \geq 0.99d$*,* LExt *is a* $(0.5d, 2^{-0.02d})$*-strong extractor with seed* $\mathbf{Y}$*.*

Proof. We claim that we can take LExt to be the (k, ε)-extractor in Lemma 41, where $\varepsilon = 2^{-0.03d}$ and $k = 0.5d$. Note that the restriction on k and d is satisfied by our choice of parameters. Since LExt is a strong-$(k, 2^{-0.03d})$ extractor with seed $\mathbf{U}_d$, Lemma 27 implies that for every distribution $\mathbf{Y} \in \{0,1\}^d$ with min-entropy $0.99d$, LExt is a $(k, 2^{-0.02d})$-strong extractor with seed $\mathbf{Y}$. ◀

Finally we introduce (a special case of) affine correlation breakers. Roughly speaking, if we are given correlated random variables $(\mathbf{Y}_1, \ldots, \mathbf{Y}_t)$ where $\mathbf{Y}_i$ is uniform, we can feed $(\mathbf{Y}_1, \ldots, \mathbf{Y}_t)$ into a correlation breaker, and break the correlation of the i-th output from the other output, with the help of an extra independent source $\mathbf{X}$. We say a correlation breaker is an affine correlation breaker if we allow the extra source to be in the form $\mathbf{X} = \mathbf{A} + \mathbf{B}$ where $\mathbf{A}$ is an independent source but $\mathbf{B}$ can be correlated with $(\mathbf{Y}_1, \ldots, \mathbf{Y}_t)$.

▶ **Definition 43** ([41, 10])**.** *We say* $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times [t] \to \{0,1\}^m$ *is a* (t, k, ε)*-affine correlation breaker if for every distribution* $\mathbf{A}, \mathbf{B} \in \{0,1\}^n$*,* $\mathbf{Y}_1, \ldots, \mathbf{Y}_t \in \{0,1\}^d$ *and every* $i^* \in [t]$ *such that*

- $\mathrm{H}_\infty(\mathbf{A}) \geq k$*,*
- $\mathbf{A}$ *is independent of* $(\mathbf{B}, \mathbf{Y}_1, \ldots, \mathbf{Y}_t)$*,*
- $\mathbf{Y}_{i^*} = \mathbf{U}_d$*,*

it holds that

$$(\mathrm{ACB}(\mathbf{A} + \mathbf{B}, \mathbf{Y}_{i^*}, i^*) \approx_\varepsilon \mathbf{U}_m) \mid \{\mathrm{ACB}(\mathbf{A} + \mathbf{B}, \mathbf{Y}_i, i)\}_{i \in [t] \setminus \{i^*\}}$$

We need the following construction of affine correlation breaker which can work for $\varepsilon = 2^{-\Omega(n)}$.

▶ **Lemma 44** ([7, 13])**.** *For every* $t = O(1)$*, there exists a universal constant* C *such that for* $\varepsilon > 0$ *and* $m \in \mathbb{N}$*, there exists an explicit* (t, k, ε)*-affine correlation breaker* $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^d \times [t] \to \{0,1\}^m$ *such that* $d = C\log(n/\varepsilon)$ *and* $k = C(m + \log(n/\varepsilon))$*.*

Now we are ready to prove Theorem 12.

Proof of Theorem 12. The construction of ILExt is as follows.

1. Take $\mathbf{X}_1$ to be a length-$(n/3)$ prefix of $\mathbf{X}$.
2. Compute $(\mathbf{S}_1, \mathbf{S}_2, \ldots, \mathbf{S}_t) = \mathrm{SRCon}(\mathbf{X}_1)$, where $\mathrm{SRCon} : \{0,1\}^{n/3} \to (\{0,1\}^{\gamma_1 n})^t$ is the $(3\delta \to 0.99, 2^{-\gamma_2 n})$ somewhere random condenser from Lemma 40. ($t \in \mathbb{N}, \gamma_1 > 0, \gamma_2 > 0$ are constants depending on δ. Specifically, we can make $\gamma_1 < \delta$.)
3. Define $\gamma_3 = \min(\delta/3t, \gamma_1/3)$, and let $\mathrm{LExt} : \{0,1\}^n \times \{0,1\}^{\gamma_1 n} \to \{0,1\}^{\gamma_3 n}$ be the $(0.5\gamma_1 n, 2^{-0.01\gamma_1 n})$-strong linear extractor from Lemma 42 which can work for any seed with $0.99\gamma_1 n$ min-entropy. Note that for every constant $\gamma_1 > 0$ we can guarantee that $\gamma_1 n \geq 200 \log(n)$ for large enough n.[8]
   For every $i \in [t]$, compute $\mathbf{R}_i = \mathrm{LExt}(\mathbf{X}, \mathbf{S}_i)$.
4. Output $\mathrm{ILExt}(\mathbf{X}) := \bigoplus_{i \in [t]} \mathrm{ACB}(\mathbf{X}, \mathbf{R}_i, i)$, where $\mathrm{ACB} : \{0,1\}^n \times \{0,1\}^{\gamma_3 n} \times [t] \to \{0,1\}^{\gamma n}$ is the $(t, (\delta/2)n, 2^{-\gamma_4 n})$-affine correlation breaker from Lemma 44, where $\gamma_4, \gamma > 0$ are small enough constants that satisfy the constraints $\gamma_3 n \geq C \log(n/\varepsilon)$ and $(\delta/2)n \geq C(\log(n/\varepsilon) + \gamma n)$ in Lemma 44. (C is a constant depending on t.) It suffices to choose $\gamma_4 = \min(\gamma_3/2C, \delta/4C)$ and $\gamma = \delta/8C$.

Next we prove the correctness of this construction. Let $\mathbf{A}_0$ be the prefixes of $\mathbf{A}$ of length $(1/3)n$ respectively, and $\mathbf{B}_0$ be the prefixed of $\mathbf{B}$ of length $(1/3)n$. First observe that either $\mathrm{H}_\infty(\mathbf{A}_0) \geq \delta n$ or $\mathrm{H}_\infty(\mathbf{B}_0) \geq \delta n$, since

$$\mathrm{H}_\infty(\mathbf{A}_0) + \mathrm{H}_\infty(\mathbf{B}_0) \geq \mathrm{H}_\infty(\mathbf{A}_0 + \mathbf{B}_0) \geq \mathrm{H}_\infty(\mathbf{A} + \mathbf{B}) - (2/3)n \geq 2\delta n.$$

Note that $\mathbf{A}$ and $\mathbf{B}$ are symmetric in this theorem, so without loss of generality we assume that $\mathrm{H}_\infty(\mathbf{A}_0) \geq \delta n$. By Lemma 24 and Lemma 25, we have $\mathrm{H}_\infty(\mathbf{B}|_{\mathbf{B}_0 = b_0}) \geq (\delta/2)n$ with probability $1 - 2^{-(\delta/2)n}$ over the fixing $\mathbf{B}_0 = b_0$. For the rest of the proof we fix $\mathbf{B}_0 = b_0$ and only consider b_0 which makes $\mathrm{H}_\infty(\mathbf{B}) \geq (\delta/2)n$, and add back the $2^{-(\delta/2)n} = 2^{-\Omega(n)}$ error in the end.

Observe that $\mathrm{H}_\infty(\mathbf{X}_0) = \mathrm{H}_\infty(\mathbf{A}_0 + b_0) \geq \delta n$. Therefore $\mathbf{S}_{[t]}$ is $2^{-\gamma_2 n}$-close to a somewhere random $0.99\gamma_1 n$-source. For every $i \in [t]$, define $\mathbf{R}_{A,i} = \mathrm{LExt}(\mathbf{A}, \mathbf{S}_i)$ and $\mathbf{R}_{B,i} = \mathrm{LExt}(\mathbf{B}, \mathbf{S}_i)$. Note that $\mathbf{R}_i = \mathbf{R}_{A,i} + \mathbf{R}_{B,i}$. Now assume that there exists $i \in [t]$ such that $\mathbf{S}_i$ has min-entropy $0.99\gamma_1 n$. Because $\mathbf{S}_i$ is independent of $\mathbf{B}$, and $\mathrm{H}_\infty(\mathbf{B}) \geq 0.5\delta n \geq 0.5\gamma_1 n$, we have then $\mathbf{R}_{B,i} \approx_{2^{-\Omega(n)}} \mathbf{U}_{\gamma_3 n}$ with probability $1 - 2^{-\Omega(n)}$ over the fixing of $\mathbf{S}_i$ by our choice of parameters of LExt and Markov argument. Moreover, after fixing $\mathbf{S}_i$, $\mathbf{R}_{B,i}$ is independent of $\mathbf{A}_0$. Therefore, with probability $1 - 2^{-\Omega(n)}$ over the fixing of $\mathbf{A}_0$ (which would also fix $\mathbf{S}_i$), $\mathbf{R}_{B,i} \approx_{2^{-\Omega(n)}} \mathbf{U}_{\gamma_3 n}$. Then observe that we can remove the assumption and use the fact that $\mathbf{S}_{[t]}$ is $2^{-\gamma_2 n}$-close to a somewhere random $0.99\gamma_1 n$-source to conclude that with probability $1 - 2^{-\Omega(n)}$ over the fixing of $\mathbf{A}_0$, there exists $i^* \in [t]$ such that $\mathbf{R}_{B,i^*} \approx_{2^{-\Omega(n)}} \mathbf{U}_{\gamma_3 n}$. Moreover, since $\mathbf{R}_{A,[t]}$ is independent of $\mathbf{R}_{B,i^*}$ after fixing $\mathbf{A}_0$, we have $\mathbf{R}_{i^*} \approx_{2^{-\Omega(n)}} \mathbf{U}_{\gamma_3 n}$ over any further fixing of $\mathbf{R}_{A,[t]}$.

Next, observe that by Lemma 24,

$$\widetilde{\mathrm{H}}_\infty(\mathbf{A} \mid (\mathbf{A}_0, \mathbf{R}_{A,1}, \ldots, \mathbf{R}_{A,t})) \geq \mathrm{H}_\infty(\mathbf{A}) - (1/3)n - (t\gamma_3)n \geq (2/3)\delta n.$$

By Lemma 25 and union bound, we can conclude that with probability $1 - 2^{-\Omega(n)}$ over the fixing of $\mathbf{A}_0, \mathbf{R}_{A,1}, \ldots, \mathbf{R}_{A,t}$, we have $\mathbf{R}_{i^*} \approx_{2^{-\Omega(n)}} \mathbf{U}_{\gamma_3 n}$ and $\mathrm{H}_\infty(\mathbf{A}) \geq \delta/2n$. Moreover, observe that under any such fixing, $\mathbf{A}$ is independent of $(\mathbf{B}, \mathbf{R}_{[t]})$. Therefore, by Lemma 44 we can conclude that

[8] If $\gamma_3 < 1/3$ we can simply take the prefix of length $\gamma_3 n$ of the output. The output is still uniform, and LExt is still linear.

$$(\mathrm{ACB}(\mathbf{A}+\mathbf{B},\mathbf{R}_{i^*},i^*) \approx_{2^{-\gamma_4 n}} \mathbf{U}_{\gamma n}) \mid \{\mathrm{ACB}(\mathbf{A}+\mathbf{B},\mathbf{R}_i,i)\}_{i\in[t]\setminus\{i^*\}},$$

which implies

$$\mathrm{ILExt}(\mathbf{A}+\mathbf{B}) = \bigoplus_{i\in[t]} \mathrm{ACB}(\mathbf{A}+\mathbf{B},\mathbf{R}_{i^*},i^*) \approx_{2^{-\gamma_4 n}} \mathbf{U}_{\gamma n}.$$

Finally, after adding back all the $2^{-\Omega(n)}$ error that we mentioned above, the error is still $2^{-\Omega(n)}$. ◀

5 Kakeya sets and HSGs for regular ROLBPs

In this section, we prove Theorem 18, which says that rank-r Kakeya set is a hitting set for oblivious ROLBPs of width $(r+1)$, and Theorem 20, a size lower bound for rank-r Kakeya set over $\mathbb{F}_2^n$.

In [6], it was proved that a Hamming ball of radius $(w-1)$ is a hitting set for regular read-once branching program of width w.[9] Their proof relies on the fact that there are only $(w-1)$ "crucial layers" such that we can only make a "fatal decision" which goes from a "possibly accept" node to an "always reject" node in these layers. The formal statement is as follows.

▶ **Lemma 45** ([6]). *For a ROBP f on $\mathbb{F}_2^n$ with layers $L_0, L_1, \ldots, L_n$, we say a layer L_i is crucial if there exists $v \in L_i$ and an edge $(u \to v)$ such that u can reach an accepting state but v cannot.*[10] *Then for every $w \in \mathbb{N}$, a regular ROBP of width w has at most $(w-1)$ crucial layers.*

Based on this lemma, [6] observed that in order to find an input x of which the computation path reaches an accepting state, we only need to make sure that we do not make any fatal decision in the crucial layers, and the bits read in the other layer can simply be set to 0. Therefore, the Hamming ball of radius $(w-1)$ centered around 0 is a hitting set for regular ROBPs of width w, because the Hamming ball covers every possible decision in the crucial layers, no matter where the crucial layers are. This makes sure that we can find a string which does not make any fatal decision, and this string would reach the sink labeled with 1 in the end.

To generalize this argument to the setting of regular ROLBPs, we want to find a set H such that for every possible rotation R of $\mathbb{F}_2^n$, the rotation of H (denoted by $R(H)$) contains a string which does not make any fatal decision. A naive idea is to find a set which contains every possible rotation of Hamming balls centered at 0. However, this contains exactly the whole set $\mathbb{F}_2^n$. To deal with this issue, we observe that for the argument in [6] to work, we only need to make sure that for every possible choices of crucial layers $L_{i_1}, \ldots, L_{i_{w-1}}$, where $I = \{i_1, \ldots, i_{w-1}\} \subseteq [n]$, there exists a fixing of the bits outside the crucial layer, such that we enumerate over every possible choice of bits in the crucial layers. Note that the fixing does not need to be 0 and can depend on the choice of crucial layers I. That is, for every set $I \subseteq [n]$ of size at most $(w-1)$, we need to enumerate over a subcube with free bits in I and arbitrary fixing outside I. To ensure this for every possible rotation, what we need is exactly a *Kakeya set*. Next we give a formal proof of our argument.

[9] A Hamming ball of radius r centered around $c \in \{0,1\}^n$ is the set of all the strings which are different from c in at most r bits.

[10] Accepting states are the sinks with label 1.

▶ **Lemma 46.** *Let $H \subseteq \mathbb{F}_2^n$ be a set which satisfies the following: for every $I \subseteq [n]$ of size $(w-1)$, there exists $b \in \mathbb{F}_2^n$ such that $b + \mathrm{span}(\{e_i\}_{i \in I}) \subseteq H$. Then H is a hitting set for regular branching programs of width w.*

Proof. Let f be a regular branching program of width w which accepts at least one string. By Lemma 45, there are at most $(w-1)$ crucial layers in f. Let I denote the set of indices of these crucial layers. By assumption there exists $b \in \mathbb{F}_2^n$ such that $b + \mathrm{span}(\{e_i\}_{i \in I}) \subseteq H$. Now we define a string $b' \in \mathbb{F}_2^n$ inductively as follows. Let v_0 be the source of f, and for every non-sink node v and every $b \in \{0,1\}$ let $\mathsf{next}(v, b)$ denote the node which v connects to with an edge of label b. For i from 1 to n, we define b'_i (the i-th bit of b') as follows:

- If $i \notin I$, then set $b'_i = b_i$.
- If $i \in I$, then set $b'_i = 0$ if $\mathsf{next}(v_{i-1}, 0)$ can reach a accepting state. Otherwise set $b'_i = 1$.

Then we define $v_i = \mathsf{next}(v_{i-1}, b'_i)$. First observe that b' only differ from b on the bits with indices in I. Therefore $b' \in H$. It remains to prove that $f(b') = 1$. Next we prove by induction that every v_i can reach a accepting state. This means v_n is a accepting state, i.e. $f(b') = 1$. For the base case, note that v_0 is the source and hence can reach a accepting state by assumption. To prove that v_i can reach an accepting state assuming that v_{i-1} can reach an accepting state, consider two cases. If $i \notin I$, then the i-th layer is not crucial, which means v_i can reach a accepting state. If $i \in I$, observe that at least one node in $\{\mathsf{next}(v_{i-1}, 0), \mathsf{next}(v_{i-1}, 1)\}$ should be able to reach a accepting state, because they are the only nodes that v_{i-1} can connect to, and v_{i-1} can reach a accepting state. Therefore v_i can also reach a accepting state by definition of b'_i. ◀

Now we are ready to prove Theorem 18.

Proof of Theorem 18. Let K be a rank-r Kakeya set, and f be any oblivious ROLBP of width $(r+1)$ that accepts at least one string. Observe that there exists a full-rank matrix $R \in \mathbb{F}_2^{n \times n}$ and a read-once regular BP f' of width $(r+1)$ such that for every $x \in \mathbb{F}_2^n$ we have $f(x) = f'(Rx)$. We claim that f' accepts at least one string in $H = \{Rx : x \in K\}$, which implies that f accepts at least one string in K.

For every $I \subseteq [n]$ of size r, observe that there exists $b \in \mathbb{F}_2^n$ such that

$$b + \mathrm{span}(\{R^{-1}e_i\}_{i \in I}) \subseteq K,$$

by definition of Kakeya set. This implies that $Rb + \mathrm{span}(\{e_i\}_{i \in I}) \subseteq H$. By Lemma 46, H is a hitting set for regular branching programs of width $(r+1)$. Therefore f' accepts at least one string in H. ◀

▶ **Corollary 47.** *For every $r, n \in \mathbb{N}$ s.t. $r \leq n$, there is an explicit hitting set $K \subseteq \mathbb{F}_2^n$ for oblivious read-once regular linear BP of width $(r+1)$ such that $|K| \leq 2^{\lceil (1-2^{-r})n \rceil + r}$.*

5.1 Limitation to our approach

Next we prove Theorem 20, which proves a lower bound on rank-r Kakeya sets and implies that the seed length of hitting set generator based on our approach cannot be improved by much.

▶ **Theorem 20** (restated). *Every rank-r Kakeya set over $\mathbb{F}_2^n$ has size at least $2^{(1-2^{-r})(n+2)-r}$.*

Proof. Let $s_{n,r}$ denote the minimum size of rank-r Kakeya set over $\mathbb{F}_2^n$. Clearly $S_{n,0} = 1$ for every $n \in \mathbb{N}$. We will show that for every n, r we have $S_{n,r}^2 \geq 2^{n+1} S_{n-1,r-1}$, and then the claimed bound easily follows by induction.

To prove this claim, consider any rank-r Kakeya set over $\mathbb{F}_2^n$, denoted by K, and for every non-zero $a \in \mathbb{F}_2^n$ define $K_a = \{v \in \mathbb{F}_2^n : v \in K \wedge v + a \in K\}$. We claim that for every a we have $|K_a| \geq 2S_{n-1,r-1}$. (Note that this also implies $|K| \geq 2S_{n-1,r-1}$ because every K_a is a subset of K.) To prove this, first we assume w.l.o.g. that the n-th bit of a is 1, and define $K'_a = \{v' \in \mathbb{F}_2^{n-1} : v' \circ 0 \in K_a\}$. Note that $|K'_a| = |K_a|/2$ because for every $v \in \mathbb{F}_2^n$ we have $v \in K_a$ if and only if $v + a \in K_a$, and exactly one of $\{v, v + a\}$ has the last bit being 0.

We claim that K'_a is a rank-$(r-1)$ Kakeya set over size $\mathbb{F}_2^{n-1}$, and hence has size at least $S_{n-1,r-1}$. To prove this, consider any subspace $V' \subseteq \mathbb{F}_2^{n-1}$ of dimension $(r-1)$, and let V denote the subspace of $\mathbb{F}_2^n$ which consists of vectors in V' padded with a 0 in the last bit. Since K is a rank-r Kakeya set, there exists $b \in \mathbb{F}_2^n$ such that $b + V + \{0^n, a\} \subseteq K$. W.l.o.g. we can assume that the last bit of b is 0, i.e. $b = b' \circ 0$ for some $b' \in \mathbb{F}_2^{n-1}$. Then observe that $V' + b' \subseteq K'_a$, because for every $v' \in V'$ we have that $(v' \circ 0) + (b' \circ 0) \in K$ and $(v' \circ 0) + (b' \circ 0) + a \in K$, which implies that $v' + b' \in K'_a$.

Since the same argument works for every subspace V' of dimension $(r-1)$, this means K'_a is a rank-r Kakeya set. Finally, consider the bijective function $f : \mathbb{F}_2^n \times \mathbb{F}_2^n \to \mathbb{F}_2^n \times \mathbb{F}_2^n$ defined as $f(v_1, v_2) = (v_1, v_2 - v_1)$. Observe that the image of f on $K \times K$ is exactly $(K \times \{0^n\}) \cup \bigcup_{a \in \mathbb{F}_2^n, a \neq 0^n} K_a \times \{a\}$. This implies $|K|^2 \geq 2^{n+1} S_{n-1,r-1}$, which is exactly the bound we want. ◀

References

1. Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992.
2. Boaz Barak, Guy Kindler, Ronen Shaltiel, Benny Sudakov, and Avi Wigderson. Simulating independence: New constructions of condensers, ramsey graphs, dispersers, and extractors. *J. ACM*, 57(4):20:1–20:52, 2010. doi:10.1145/1734213.1734214.
3. Andrej Bogdanov, Zeev Dvir, Elad Verbin, and Amir Yehudayoff. Pseudorandomness for width-2 branching programs. *Theory of Computing*, 9(1):283–293, 2013.
4. Andrej Bogdanov, William M Hoza, Gautam Prakriya, and Edward Pyne. Hitting sets for regular branching programs. In *37th Computational Complexity Conference (CCC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.
5. Andrej Bogdanov, Periklis A. Papakonstantinou, and Andrew Wan. Pseudorandomness for read-once formulas. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 240–246, 2011. doi:10.1109/FOCS.2011.57.
6. Mark Braverman, Anup Rao, Ran Raz, and Amir Yehudayoff. Pseudorandom generators for regular branching programs. *SIAM Journal on Computing*, 43(3):973–986, 2014.
7. Eshan Chattopadhyay, Jesse Goodman, and Jyun-Jie Liao. Affine extractors for almost logarithmic entropy. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021*, pages 622–633, 2021. doi:10.1109/FOCS52979.2021.00067.
8. Eshan Chattopadhyay, Jesse Goodman, and David Zuckerman. The space complexity of sampling. In *13th Innovations in Theoretical Computer Science Conference (ITCS 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.
9. Eshan Chattopadhyay, Pooya Hatami, Omer Reingold, and Avishay Tal. Improved pseudorandomness for unordered branching programs through local monotonicity. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 363–375, 2018.
10. Eshan Chattopadhyay and Xin Li. Extractors for sumset sources. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016*, pages 299–311, 2016. doi:10.1145/2897518.2897643.
11. Eshan Chattopadhyay and Xin Li. Non-malleable codes and extractors for small-depth circuits, and affine functions. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1171–1184, 2017.

12 Eshan Chattopadhyay and Xin Li. Non-malleable codes, extractors and secret sharing for interleaved tampering and composition of tampering. In *Theory of Cryptography - 18th International Conference, TCC 2020*, volume 12552 of *Lecture Notes in Computer Science*, pages 584–613, 2020. doi:10.1007/978-3-030-64381-2_21.

13 Eshan Chattopadhyay and Jyun-Jie Liao. Extractors for sum of two sources. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1584–1597, 2022. doi:10.1145/3519935.3519963.

14 Eshan Chattopadhyay and David Zuckerman. New extractors for interleaved sources. In *31st Conference on Computational Complexity, CCC 2016*, volume 50 of *LIPIcs*, pages 7:1–7:28, 2016. doi:10.4230/LIPIcs.CCC.2016.7.

15 Kuan Cheng and William M Hoza. Hitting sets give two-sided derandomization of small space. In *35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

16 Mahdi Cheraghchi and Venkatesan Guruswami. Non-malleable coding against bit-wise and split-state tampering. In *Theory of Cryptography Conference*, pages 440–464. Springer, 2014.

17 Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. Comput.*, 17(2):230–261, 1988. doi:10.1137/0217015.

18 Gil Cohen. Local correlation breakers and applications to three-source extractors and mergers. *SIAM J. Comput.*, 45(4):1297–1338, 2016. doi:10.1137/15M1029837.

19 Gil Cohen and Igor Shinkar. The complexity of DNF of parities. In *Proceedings of the 2016 ACM Conference on Innovations in Theoretical Computer Science, ITCS 2016*, pages 47–58, 2016. doi:10.1145/2840728.2840734.

20 Evgeny Demenkov and Alexander S. Kulikov. An elementary proof of a 3n - o(n) lower bound on the circuit complexity of affine dispersers. In *Mathematical Foundations of Computer Science 2011 - 36th International Symposium, MFCS 2011*, volume 6907 of *Lecture Notes in Computer Science*, pages 256–265, 2011. doi:10.1007/978-3-642-22993-0_25.

21 Yevgeniy Dodis, Xin Li, Trevor D. Wooley, and David Zuckerman. Privacy amplification and non-malleable extractors via character sums. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011*, pages 668–677, 2011. doi:10.1109/FOCS.2011.67.

22 Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam D. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008. doi:10.1137/060651380.

23 Yevgeniy Dodis and Daniel Wichs. Non-malleable extractors and symmetric key cryptography from weak secrets. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 601–610, 2009. doi:10.1145/1536414.1536496.

24 Jordan S Ellenberg, Richard Oberlin, and Terence Tao. The kakeya set and maximal conjectures for algebraic varieties over finite fields. *Mathematika*, 56(1):1–25, 2010.

25 Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than-3n lower bound for the circuit complexity of an explicit function. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 89–98, 2016. doi:10.1109/FOCS.2016.19.

26 Michael A. Forbes and Venkatesan Guruswami. Dimension expanders via rank condensers. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2015*, volume 40 of *LIPIcs*, pages 800–814, 2015.

27 Michael A. Forbes and Zander Kelley. Pseudorandom generators for read-once branching programs, in any order. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018*, pages 946–955, 2018. doi:10.1109/FOCS.2018.00093.

28 Uma Girish, Avishay Tal, and Kewen Wu. Fourier growth of parity decision trees. *arXiv preprint*, 2021. arXiv:2103.11604.

29 Svyatoslav Gryaznov, Pavel Pudlák, and Navid Talebanfard. Linear branching programs and directional affine extractors. In *37th Computational Complexity Conference, CCC 2022*, volume 234 of *LIPIcs*, pages 4:1–4:16, 2022. doi:10.4230/LIPIcs.CCC.2022.4.

30 Venkatesan Guruswami, Christopher Umans, and Salil P. Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *J. ACM*, 56(4):20:1–20:34, 2009. doi:10.1145/1538902.1538904.

31 John Hastad. Almost optimal lower bounds for small depth circuits. *Adv. Comput. Res.*, 5:143–170, 1989.

32 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for xor functions. *SIAM Journal on Computing*, 47(1):208–217, 2018.

33 Russell Impagliazzo, Leonid A. Levin, and Michael Luby. Pseudo-random generation from one-way functions (extended abstracts). In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, STOC 1989*, pages 12–24, 1989. doi:10.1145/73007.73009.

34 Stasys Jukna. *Boolean Function Complexity – Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-24508-4.

35 Jesse Kamp, Anup Rao, Salil P. Vadhan, and David Zuckerman. Deterministic extractors for small-space sources. *J. Comput. Syst. Sci.*, 77(1):191–220, 2011. doi:10.1016/j.jcss.2010.06.014.

36 Swastik Kopparty, Vsevolod F Lev, Shubhangi Saraf, and Madhu Sudan. Kakeya-type sets in finite vector spaces. *Journal of Algebraic Combinatorics*, 34(3):337–355, 2011.

37 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993. doi:10.1137/0222080.

38 Chin Ho Lee, Edward Pyne, and Salil P. Vadhan. Fourier growth of regular branching programs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022*, volume 245 of *LIPIcs*, pages 2:1–2:21, 2022. doi:10.4230/LIPIcs.APPROX/RANDOM.2022.2.

39 Chin Ho Lee and Emanuele Viola. Some limitations of the sum of small-bias distributions. *Theory of Computing*, 13(1):1–23, 2017.

40 Jiatu Li and Tianqi Yang. $3.1n - o(n)$ circuit lower bounds for explicit functions. In *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1180–1193, 2022. doi:10.1145/3519935.3519976.

41 Xin Li. Improved two-source extractors, and affine extractors for polylogarithmic entropy. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 168–177, 2016. doi:10.1109/FOCS.2016.26.

42 Xin Li. Two source extractors for asymptotically optimal entropy, and (many) more. *CoRR*, abs/2303.06802, 2023. doi:10.48550/arXiv.2303.06802.

43 Xin Li and Yan Zhong. Explicit directional affine extractors and improved hardness for linear branching programs. *CoRR*, abs/2304.11495, 2023. doi:10.48550/arXiv.2304.11495.

44 Ueli M. Maurer and Stefan Wolf. Privacy amplification secure against active adversaries. In *Advances in Cryptology – CRYPTO '97, 17th Annual International Cryptology Conference*, volume 1294 of *Lecture Notes in Computer Science*, pages 307–321, 1997. doi:10.1007/BFb0052244.

45 Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993. doi:10.1137/0222053.

46 Noam Nisan. Pseudorandom generators for space-bounded computation. *Comb.*, 12(4):449–461, 1992. doi:10.1007/BF01305237.

47 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of computer and System Sciences*, 49(2):149–167, 1994.

48 Ran Raz. Extractors with weak random seeds. In *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, STOC 2005*, pages 11–20, 2005. doi:10.1145/1060590.1060593.

49 Omer Reingold, Thomas Steinke, and Salil Vadhan. Pseudorandomness for regular branching programs via fourier analysis. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 655–670. Springer, 2013.

50 Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017*, pages 238–251, 2017.

51 Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier sparsity, spectral norm, and the log-rank conjecture. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 658–667. IEEE, 2013.

52 Yoav Tzur. Notions of weak pseudorandomness and gf (2n)-polynomials. *Master's thesis, Weizmann Institute of Science*, 2009.

53 Salil P. Vadhan. Pseudorandomness. *Found. Trends Theor. Comput. Sci.*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.

54 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, pages 1–10, 1985. doi:10.1109/SFCS.1985.49.

55 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory Comput.*, 3(1):103–128, 2007. doi:10.4086/toc.2007.v003a006.

A Definitions of strongly read-once linear branching programs

The difference between the definition of strongly read-once in [29] and our definition (Definition 2) is as follows. First, let Pre'_u denote the span of all the linear queries on a path to u, *excluding the query* ℓ_u *on* u. The definition of strongly read-once in [29] is that $\mathsf{Pre}'_u \cap \mathsf{Post}_u = \{0\}$ for every node u. First we show that the definition in [29] implies Definition 2.

▷ Claim 48. In a linear branching program, if for every node v it holds that $\mathsf{Pre}'_v \cap \mathsf{Post}_v = \{0\}$, then

- For every edge (u, v), $\mathsf{Pre}_u \cap \mathsf{Post}_v = \{0\}$.
- For every node v, $\mathsf{Pre}_v \cap \mathsf{Post}_v = \{0, \ell_v\}$.

Proof. Let $P(v)$ denote the set of all nodes u such that there is an edge $(u \to v)$. Observe that $\mathsf{Pre}'_v = \mathrm{span}(\bigcup_{u \in P(v)} \mathsf{Pre}_u)$. Therefore, if $\mathsf{Pre}'_v \cap \mathsf{Post}_v = \{0\}$, then $(\mathsf{Pre}_u \cap \mathsf{Post}_v) \subseteq (\mathsf{Pre}'_v \cap \mathsf{Post}_v)$ for every $u \in P(v)$, which implies $\mathsf{Pre}_u \cap \mathsf{Post}_v = \{0\}$ for every $u \in P(v)$. To prove the second property, note that $\mathsf{Pre}_v = \mathsf{Pre}'_v \cup (\mathsf{Pre}'_v + \ell_v)$. Because Post_v is a subspace that contains ℓ_v, we have $(\mathsf{Pre}'_v + \ell_v) \cap \mathsf{Post}_v = (\mathsf{Pre}'_v + \ell_v) \cap (\mathsf{Post}_v + \ell_v) = (\mathsf{Pre}'_v \cap \mathsf{Post}_v) + \ell_v = \{\ell_v\}$, which implies $\mathsf{Pre}_v \cap \mathsf{Post}_v = \{0, \ell_v\}$. ◁

Next we show that our definition is strictly more general.

▷ Claim 49. There exists a linear branching program which satisfies the strongly read-once definition in Definition 2, but contains some node w such that $\mathsf{Pre}'_w \cap \mathsf{Post}_w \neq \{0\}$.

Proof. To see why this is the case, consider a linear branching programs with four non-sink nodes, s, v_1, v_2, w, and the two edges of w connect to two sink labeled with 0 and 1 respectively. Furthermore, we choose the queries on these nodes to be $\ell_s = e_3$, $\ell_{v_1} = e_1$, $\ell_{v_2} = e_2$ and $\ell_w = e_1 + e_2$. Then observe that both Pre'_w and Post_w contain $e_1 + e_2$, but this linear branching program satisfies the definition in Definition 2. ◁

In fact, from the example above, one can see that our definition of strongly read-once is technically incomparable with the "weakly read-once" model defined in [29], which requires that $\ell_v \notin \mathsf{Pre}'_v$ for every v. However, our definition is closer to strongly read-once in [29] because we still need the fact that the queries before v and the queries after v do not affect each other.

Finally let us elaborate what the second property in our definition means, because it might seem less intuitive. Given the first definition, we see that every edge $e = (u \to v)$ decompose $\mathbb{F}_2^n$ into two complemented subspace. The second property is to make sure that the dimension of both subspaces in this decomposition change by at most 1 when we move one step from an edge $(u \to v)$ to another edge $(v \to w)$. This is to make sure that for every path we can find an edge e such that the dimension of Pre_u and Post_v is exactly what we want. Without this property, the size lower bound in Theorem 11 would become roughly $2^{n-k_1-2k_2}$ because the dimension of Post_v can drop down to half after one step, and the directional affine extractor in [29] would no longer work.

B Directional affine extractors are non-malleable

Recently, stronger variants of seeded and seedless extractors, called non-malleable extractors have been studied, with motivations from cryptography and pseudorandomness [23, 16]. In this section we show that directional affine extractors are equivalent to affine extractors that are non-malleable against tampering functions that are constant shift.

We refer the reader to [16] for the general definition of seedless non-malleable extractors, and present the definition specialized to our setting below.

▶ **Definition 50.** *We say* $\mathrm{Ext} : \mathbb{F}_2^n \to \{0,1\}$ *is a* (d,ε)-non-malleable affine extractor against shifts *if for every source* $\mathbf{X} \in \mathbb{F}_2^n$ *which is uniform over an affine subspace of dimension* d*, and every non-zero shift* $a \in \mathbb{F}_2^n$*,*

$$(\mathrm{Ext}(\mathbf{X}) \approx_\varepsilon \mathbf{U}_1) \mid \mathrm{Ext}(\mathbf{X}+a).$$

It's easy to see that a (d,ε)-non-malleable affine extractor against shifts is also a (d,ε)-directional affine extractor. We prove the converse below.

▶ **Theorem 51.** *For every* $d \in \mathbb{N}, \varepsilon > 0$ *such that* $d \geq \log(1/\varepsilon)$*, a* (d,ε)*-directional affine extractor is also a* $(d, O(\sqrt{\varepsilon}))$*-non-malleable affine extractor.*

Proof. To prove this theorem, we need an extension of Vazirani's XOR lemma, which can be found in [21, Lemma 3.8]. We only state the special case we need here.

▶ **Lemma 52.** *Let* $(\mathbf{W}, \mathbf{W}')$ *be a random variable over* $(\mathbb{F}_2)^2$*. If* $\mathbf{W} \approx_\varepsilon \mathbf{U}_1$ *and* $(\mathbf{W}+\mathbf{W}') \approx_\varepsilon \mathbf{U}_1$*, then*

$$(\mathbf{W} \approx_{4\varepsilon} \mathbf{U}_1) \mid \mathbf{W}'.$$

With this lemma, it suffices to prove that for every (d,ε)-directional affine extractor $\mathrm{DAExt} : \mathbb{F}_2^n \to \{0,1\}$ the following holds. for every source $\mathbf{X} \in \mathbb{F}_2^n$ which is uniform over an affine subspace of dimension d, and every non-zero shift $a \in \mathbb{F}_2^n$,

- $\mathrm{DAExt}(\mathbf{X}) \approx_{\sqrt{\varepsilon}} \mathbf{U}_1$, and
- $\mathrm{DAExt}(\mathbf{X}) + \mathrm{DAExt}(\mathbf{X}+a) \approx_{\sqrt{\varepsilon}} \mathbf{U}_1$.

The second condition is directly implied by the definition of DAExt. It remains to prove the first condition. Let V be the linear subspace which is a shift of the affine subspace $\mathrm{Supp}(\mathbf{X})$, and let $\mathbf{V}$ denote the uniform distribution over V which is independent of $\mathbf{X}$. Observe that $\mathbf{V}+\mathbf{X}$ is the same distribution as $\mathbf{X}$, and $\mathrm{H}_\infty(\mathbf{V}) \geq d \geq \log(1/\varepsilon)$. Then, by Lemma 31 we have $\mathrm{DAExt}(\mathbf{X}+\mathbf{V}) \approx_{O(\sqrt{\varepsilon})} \mathbf{U}_1$. ◀

We note that Chattopadhyay and Li [11] considered the problem of constructing non-malleable extractors against the more general class of all linear functions, but their results requires to the affine source to have dimension $0.99n$. However, it appears difficult to extend their techniques to handle smaller min-entropy, even against the weaker class of shifts.

C Extractors for average conditional min-entropy, generalized

In this section we prove the following lemma.

▶ **Lemma 29** (restated). *Let $(\mathbf{X}, \mathbf{Y}, \mathbf{E})$ be a joint distribution such that $\mathbf{X} \in \mathcal{X}$ and $\mathbf{Y} \in \mathcal{S}$ are independent conditioned on $\mathbf{E}$, and $\widetilde{\mathrm{H}}_\infty(\mathbf{X} \mid \mathbf{E}) \geq k$. Let $\mathrm{Ext} : \mathcal{X} \times \mathcal{S} \to \{0,1\}^m$ be a function which satisfies the following conditions for an error parameter $\varepsilon > 0$ and a deterministic function g: for every $e \in \mathrm{Supp}(\mathbf{E})$, there exists a set $\mathcal{X}_e \subseteq \mathcal{X}$ with size at least 2^{k+1} such that Ext when restricted to the domain $\mathcal{X}_e \times \mathcal{S}$ is a (k, ε)-extractor with seed $\mathbf{Y}|_{\mathbf{E}=e}$ and is strong in $g(e, \mathbf{Y})$. Then*

$$(\mathrm{Ext}(\mathbf{X}, \mathbf{Y}) \approx_{3\varepsilon} \mathbf{U}_m) \mid (\mathbf{E}, g(\mathbf{E}, \mathbf{Y})).$$

The proof follows the outline in [53, Problem 6.8], but each step in the proof needs to be extended to our more general definition of seeded extractors. First we need the following lemma.

▶ **Lemma 53.** *Let $\mathrm{Ext} : \mathcal{X} \times \mathcal{S} \to \{0,1\}^m$ be a (k, ε)-extractor with seed $\mathbf{Y}$, where $k \leq \log(|\mathcal{X}|) - 1$, and is strong in $g(\mathbf{Y})$ for some deterministic function g. Then for every $0 < t \leq k$, $\mathrm{Ext} : \mathcal{X} \times \mathcal{S} \to \{0,1\}$ is also a $(k - t, 2^{t+1}\varepsilon)$-extractor with seed $\mathbf{Y}$ that is strong in $g(\mathbf{Y})$.*

Proof. Let $\mathcal{G} = \mathrm{Supp}(g(\mathbf{Y}))$. It suffices to prove that for every $T \subseteq \{0,1\}^m \times \mathcal{G}$ and every $\mathbf{X}$ such that $\mathrm{H}_\infty(\mathbf{X}) \geq k - t$, it holds that

$$\Pr\left[(\mathrm{Ext}(\mathbf{X}, \mathbf{Y}), g(\mathbf{Y})) \in T\right] - \Pr\left[(\mathbf{U}_m, g(\mathbf{Y})) \in T\right] \leq (2^{t+1} - 1)\varepsilon$$

For every $x \in \mathcal{X}$, define $\delta(x) = \Pr\left[(\mathrm{Ext}(x, \mathbf{Y}), g(\mathbf{Y})) \in T\right] - \Pr[(\mathbf{U}_m, g(\mathbf{Y})) \in T]$. Let $N = |\mathcal{X}|$, and consider an ordering of the elements in $\mathcal{X}$, $x_1, \ldots, x_N$ such that $\delta(x_1) \geq \delta(x_2) \geq \ldots \geq \delta(x_N)$. Define a step function $f : (0, N] \to \mathbb{R}$ to be $f(r) = \delta(x_{\lceil r \rceil})$. Note that f is decreasing. Since Ext is a (k, ε) extractor, observe that for every $0 \leq m \leq N - 2^k$ it holds that $-\varepsilon \leq 2^{-k} \int_m^{m+2^k} f(t)\, dt \leq \varepsilon$, because $2^{-k} \int_m^{m+2^k} f(t)\, dt$ corresponds to $\Pr\left[(\mathrm{Ext}(\mathbf{X}', \mathbf{Y}), g(\mathbf{Y})) \in T\right] - \Pr\left[(\mathbf{U}_m, g(\mathbf{Y})) \in T\right]$ for some $\mathbf{X}'$ of min-entropy k. Then observe that

$$\begin{aligned}
&\Pr\left[(\mathrm{Ext}(\mathbf{X}, \mathbf{Y}), g(\mathbf{Y})) \in T\right] - \Pr\left[(\mathbf{U}_m, g(\mathbf{Y})) \in T\right] \\
&\leq 2^{t-k} \int_0^{2^{k-t}} f(t)\, dt \\
&= 2^{t-k} \left(\int_0^{2^k} f(t)\, dt - \int_{2^{k-t}}^{2^k} f(t)\, dt \right) \\
&\leq 2^{t-k} \left(\int_0^{2^k} f(t)\, dt - \frac{2^k - 2^{k-t}}{2^k} \int_{N-2^k}^{N} f(t)\, dt \right) \quad (2^k \leq N - 2^k \text{ and } f \text{ is decreasing}) \\
&\leq (2^{t+1} - 1)\varepsilon \\
&\leq 2^{t+1}\varepsilon.
\end{aligned}$$

◀

Next we prove Lemma 29.

Proof of Lemma 29. For every $e \in \mathrm{Supp}(\mathbf{E})$, write $\mathbf{X}_e = \mathbf{X}|_{\mathbf{E}=e}$ and $\mathbf{Y}_e = \mathbf{Y}|_{\mathbf{E}=e}$ for short. Note that $(\mathbf{X}, \mathbf{Y}) \mid (\mathbf{E} = e)$ is equivalent to independent distributions $(\mathbf{X}_e, \mathbf{Y}_e)$. Therefore,

$$
\begin{aligned}
&\Delta\left((\mathrm{Ext}(\mathbf{X},\mathbf{Y}),\mathbf{E},g(\mathbf{E},\mathbf{Y}));(\mathrm{Ext}(\mathbf{X},\mathbf{Y}),\mathbf{E},g(\mathbf{E},\mathbf{Y}))\right)\\
&= \mathop{\mathbb{E}}_{e\sim\mathbf{E}}\left[\Delta\left((\mathrm{Ext}(\mathbf{X}_e,\mathbf{Y}_e),g(e,\mathbf{Y}_e));(\mathrm{Ext}(\mathbf{X}_e,\mathbf{Y}_e),g(e,\mathbf{Y}_e))\right)\right]\\
&= \sum_{e:\mathrm{H}_\infty(\mathbf{X}_e)\geq k} \Pr[\mathbf{E}=e]\cdot\Delta\left((\mathrm{Ext}(\mathbf{X}_e,\mathbf{Y}_e),g(e,\mathbf{Y}_e));(\mathrm{Ext}(\mathbf{X}_e,\mathbf{Y}_e),g(e,\mathbf{Y}_e))\right)\\
&+ \sum_{e:\mathrm{H}_\infty(\mathbf{X}_e)<k} \Pr[\mathbf{E}=e]\cdot\Delta\left((\mathrm{Ext}(\mathbf{X}_e,\mathbf{Y}_e),g(e,\mathbf{Y}_e));(\mathrm{Ext}(\mathbf{X}_e,\mathbf{Y}_e),g(e,\mathbf{Y}_e))\right)\\
&= \sum_{e:\mathrm{H}_\infty(\mathbf{X}_e)\geq k} \Pr[\mathbf{E}=e]\cdot\varepsilon\\
&+ \sum_{e:\mathrm{H}_\infty(\mathbf{X}_e)\geq k} \Pr[\mathbf{E}=e]\cdot 2^{k+1-\mathrm{H}_\infty(\mathbf{X}_e)}\varepsilon \text{ (by Lemma 53)}\\
&\leq \sum_e \Pr[\mathbf{E}=e]\cdot\varepsilon + \sum_e \Pr[\mathbf{E}=e]\cdot 2^{k+1-\mathrm{H}_\infty(\mathbf{X}_e)}\varepsilon\\
&= \varepsilon + 2^{-k}\cdot 2^{1+\widetilde{\mathrm{H}}_\infty(\mathbf{X}|\mathbf{E})}\cdot\varepsilon\\
&\leq 3\varepsilon.
\end{aligned}
$$

◀

An Improved Trickle down Theorem for Partite Complexes

Dorna Abdolazimi
University of Washington, Seattle, WA, USA

Shayan Oveis Gharan
University of Washington, Seattle, WA, USA

Abstract

We prove a strengthening of the trickle down theorem for partite complexes. Given a $(d+1)$-partite d-dimensional simplicial complex, we show that if "on average" the links of faces of co-dimension 2 are $\frac{1-\delta}{d}$-(one-sided) spectral expanders, then the link of any face of co-dimension k is an $O(\frac{1-\delta}{k\delta})$-(one-sided) spectral expander, for all $3 \leq k \leq d+1$. For an application, using our theorem as a black-box, we show that links of faces of co-dimension k in recent constructions of bounded degree high dimensional expanders have spectral expansion at most $O(1/k)$ fraction of the spectral expansion of the links of the worst faces of co-dimension 2.

2012 ACM Subject Classification Theory of computation → Expander graphs and randomness extractors; Theory of computation → Error-correcting codes; Theory of computation → Random walks and Markov chains

Keywords and phrases Simplicial complexes, High dimensional expanders, Trickle down theorem, Bounded degree high dimensional expanders, Locally testable codes, Random walks

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.10

Funding *Dorna Abdolazimi*: Research supported by NSF grant CCF-1907845 and Air Force Office of Scientific Research grant FA9550-20-1-0212.
Shayan Oveis Gharan: Research supported by Air Force Office of Scientific Research grant FA9550-20-1-0212, Simons Investigator grant.

Acknowledgements The discussion that initiated this work took place at the DIMACS Workshop on Entropy and Maximization. We would like to thank the DIMACS center and the workhop organizers for making this happen. In particular, we would like to thank Tali Kaufman for raising the question of an improved trickle down theorem for sparse simplical complexes in that workshop. We also would like to thank Ryan O'Donnell and Kevin Pratt for helpful discussions on high dimensional expanders based on Chevalley groups.

1 Introduction

A simplicial complex X on a finite ground set $[n] = \{0, \ldots, n\}$ is a downwards closed collection of subsets of $[n]$, i.e. if $\tau \in X$ and $\sigma \subset \tau$, then $\sigma \in X$. The elements of X are called *faces*, and the maximal faces are called *facets*. We say that a face τ is of dimension k if $|\tau| = k+1$ and write $\dim(\tau) = k$. A simplicial complex X is a *pure* d-dimensional complex if every facet has dimension d. In this paper, all simplicial complexes are assumed to be pure. Given a d-dimensional complex X, for any $0 \leq i \leq d$, define $X(i) = \{\tau \in X : \dim(\tau) = i\}$. Moreover, the co-dimension of a face $\tau \in X$ is defined as $\mathrm{codim}(\tau) = d - \dim(\tau)$. For a face $\tau \in X$, define the *link* of τ as the simplicial complex $X_\tau = \{\sigma \setminus \tau : \sigma \in X, \sigma \supset \tau\}$. Note that X_τ is a $(\mathrm{codim}(\tau) - 1)$- dimensional complex.

A $(d+1)$-partite complex is a a d-dimensional complex such that $X(0)$ can be (uniquely) partitioned into sets $T_0 \cup \cdots \cup T_d$ such that for every facet $\tau \in X(d)$, we have $|\tau \cap T_i| = 1$ for all $i \in [d]$. The type of any face $\tau \in X$ is defined as $\mathrm{type}(\tau) = \{i \in [d] : |\tau \cap T_i| = 1\}$.

We equip X with a probability distribution π supported on all *facets* of X and we denote this pair by (X, π). For a face $\tau \in X$, π induces a conditional distribution π_τ on facets of X_τ where for each facet $\sigma \in X_\tau$,

$$\pi_\tau(\sigma) = \frac{\pi(\sigma \cup \tau)}{\sum_{\text{facet } \sigma' \in X_\tau} \pi(\sigma' \cup \tau)}.$$

For each face τ of co-dimension at least 2 the 1-skeleton of (X_τ, π_τ) is a weighted graph with vertices $X_\tau(0)$, edges $X_\tau(1)$, and edge weights given by $\mathbb{P}_{\sigma \sim \pi_\tau}[\{x, y\} \subseteq \sigma]$ for each edge $\{x, y\}$. Note that when τ is of co-dimension 2, the complex (X_τ, π_τ) is itself a weighted graph. We say that a complex X is *totally connected* if the 1-skeleton of the link of any face τ of co-dimension at least 2 is connected.

▶ **Definition 1** (Local Spectral High Dimensional Expander). *We say that the link of a face τ of co-dimension at least 2 of a d-dimensional (weighted) complex (X, π) is a λ-(one sided) spectral expander if the second largest eigenvalue of the simple random walk on the 1-skeleton of (X_τ, π_τ) is at most λ. We say that (X, π) is a $(\gamma_2, \gamma_3, \ldots, \gamma_{d+1})$-local spectral expander if the link of any face τ of co-dimension at least 2 is a $\gamma_{\mathrm{codim}(\tau)}$-spectral expander. When the complex (X, π) is clear in the context, for an integer $2 \le k \le d+1$, we write γ_k to denote the largest 2nd eigenvalue of the simple random walk on the 1-skeleton of all links of faces of co-dimension k of the complex.*

Over the last few years, the study of local spectral high dimensional expanders (HDX) has revolutionized several areas of Math and theoretical computer science, namely in analysis of Markov chains [4, 3, 6, 1], coding theory [9], and elsewhere [2, 11, 10]. One can generally divide the family of HDXes studied in recent works into two groups: (i) Dense Complexes. Here, we have a HDX with exponentially large number of facets, i.e., $|X(0)|^d$. One typically encounters these objects in studying Markov Chain Monte Carlo technique where we use a Markov Chain to sample from an exponentially large probability distribution. Perhaps the simplest such family is the complex of all independent sets of a matroid. (ii) Sparse/Ramanujan Complexes. Here we have a HDX where every vertex (of $X(0)$) only appear in constant number of facets, independent of $|X(0)|$. See, [15, 13, 17] for explicit constructions. These objects have been useful in constructing double samplers [11], agreement testers [8, 7], or locally testable codes [9].

One of the main aspects of local spectral expanders is their "local to global phenomenon", often referred to as the Garland's method or the trickle down theorem [18].

▶ **Theorem 2** (Trickle Down Theorem). *Given a totally connected complex (X, π), if $\gamma_2 \le \frac{1-\delta}{d}$ for some $0 < \delta \le 1$, then $\gamma_k \le \frac{1-\delta}{d-(k-2)(1-\delta)} \le \frac{1-\delta}{d\delta}$ for all $2 \le k \le d$.*

The trickle down theorem has found numerous applications in proving bounds on local spectral expansion of simplicial complexes. To invoke the theorem one needs to inspect all faces of co-dimension 2 to find the worst 2nd eigenvalue. If we get lucky and this number is below $1/d$, then, the trickle down theorem kicks in and inductively bounds the spectral expansion of all links of the complex.

There are, however, two pitfalls for the theorem: i) The required bound on γ_2 is too small and often not satisfiable. In particular, for many dense complexes in counting and sampling applications that satisfy $\gamma_k = O(1/k)$ for $k \ge \Omega(d)$ (see e.g., [3, 6]), the links of faces of co-dimension 2 are only $\Theta(1)$-spectral expanders. ii) Even if $\gamma_2 \ll 1/d$, the trickle down theorem only implies $\gamma_k \simeq \gamma_2$, i.e., γ_k does not increase too much as k increases. This is in contrast with the fact that, for many dense complexes, one can observe a steep decrease in spectral expansion as the co-dimension increases, i.e., $\gamma_k \lesssim \gamma_2/k$.

Such a decrease has not been known for any sparse complex. This led some experts to conjecture that, perhaps, dense and sparse complexes exhibit a different pattern of local spectral expansion; in particular, unlike dense HDX, local spectral expansion does not decay for sparse complexes.

In this paper, we prove a generalization of the trickle down theorem for *partite complexes* that shows that even if $\gamma_2 = \Theta(1)$, but "on average" the links of faces of co-dimension 2 are $< 1/d$-spectral expanders, then we have $\gamma_k \leq O(1/k)$ for all $3 \leq k \leq d+1$. Surprising to us, our average condition is satisfied by some recent construction of (sparse) bounded degree high dimensional expanders [13, 17]. In particular, as we explain below, one can use our theorem to prove a significantly better local spectral expansion for the Kaufman-Opennheim construction in a black-box manner.

1.1 Main Contribution

We start by stating two special cases of our theorem. We need the following definition.

▶ **Definition 3.** *Given a $(d+1)$-partite complex (X, π) with parts $[d]$, for every $i \in [d]$, define*

$$\Delta_{(X,\pi)}(i) = |\{j \in [d] \setminus i : \exists \tau \text{ of } \mathrm{type}(\tau) = [d] \setminus \{i,j\} \text{ s.t. } \lambda_2(P_\tau) > 0\}|,$$

i.e. $\Delta_{(X,\pi)}(i)$ is the number of parts $j \neq i$ for which there exists a face of type $[d] \setminus \{i,j\}$ whose link is not a 0-spectral expander. Moreover, define $\Delta_{(X,\pi)} = \max_{i\in[d]} \Delta_{(X,\pi)}(i)$. We drop the subscripts (X,π) when the complex is clear in the context.

▶ **Theorem 4.** *Let (X,π) be a $(d+1)$-partite (weighted) totally connected complex. For some $0 < \delta < 1$, assume that*

$$\gamma_2 \leq \frac{\delta^2}{10(1+\ln \Delta)} \quad \text{and} \quad \gamma_2 \leq \frac{1-\delta}{\Delta + \ln \Delta}.$$

Then, the link of any face τ of co-dimension k of X has spectral expansion

$$\begin{cases} \frac{c(1-\delta)}{k\delta} & \text{if } k \geq \Delta, \\ \frac{c(1-\delta)}{k\delta}\frac{k+\ln k}{\Delta+\ln\Delta} & \text{if } k < \Delta, \end{cases}$$

for some constant $c \leq 2$ that depends on δ. .

Note that, for $\Delta = d$, this theorem retrieves Theorem 2 up to a lower order term in the condition on γ_2 and a constant in the bounds on local spectral expansions.

When $\Delta \ll d$, this theorem is a significant improvement over Theorem 2. Roughly speaking, this theorem says that, if the complex has many faces of co-dimension 2 whose links are 0-expanders, one needs to satisfy a much weaker condition on γ_2 to get $O(1/k)$-spectral expansion for faces of co-dimension k. In other words, the faces of co-dimension 2 that have perfect spectral expansion can compensate for faces of co-dimension 2 that have bad spectral expansion.

Next, we state the second special case of our theorem. For every integers $1 \leq n$, let $H_n = \sum_{i=1}^{n} \frac{1}{i}$ be the n-th harmonic number. Moreover, for any $1 \leq i \leq n$ define $H_n(i) = \sum_{j=i}^{n} \frac{1}{j}$ and let $H_n(0) = H_n(1)$.

▶ **Theorem 5.** *Let (X,π) be a $(d+1)$-partite (weighted) totally connected complex. For any distinct $i,j\in[d]$, let $\epsilon_{\{i,j\}}=\max_{\tau:\mathrm{type}(\tau)=[d]\setminus\{i,j\}}\lambda_2(P_\tau)$ be the 2nd largest eigenvalue of the simple random walk matrices on (X_τ,π_τ) for all τ of type $[d]\setminus\{i,j\}$. For some $0<\delta<1$, assume that for every $i\in[d]$,*

$$\begin{aligned}\epsilon_{\{i,j\}}\cdot H_d &\le \frac{\delta^2}{10},\forall j\neq i \quad \textit{and}\\ \sum_{\ell=1}^{d}\epsilon_{\{i,j_\ell\}}\cdot\frac{H_d(\ell-1)}{d} &\le \frac{1-\delta}{d},\end{aligned}$$

where $j_0\ldots,j_d$ is an ordering of $[d]\setminus i$ such that $\epsilon_{\{i,j_0\}}\le\cdots\le\epsilon_{\{i,j_d\}}$. Then, X is $(\frac{c(1-\delta)}{\delta},\ldots,\frac{c(1-\delta)}{d\delta})$-local spectral expander for some constant $c\le 2$ that depends on δ.

We remark that for every any $i\in[d]$, $1\le\frac{\sum_{\ell=1}^{d}H_d(\ell-1)}{d}\le 1+\frac{\ln d}{d}$. So, roughly speaking, the latter condition can be seen as $\mathbb{E}_j[\epsilon_{\{i,j\}}]\le\frac{1-\delta}{d}$ for every $i\in[d]$, where the expectation is weighted according to $\frac{H_d(.)}{d}$. This is an improvement over the stronger condition in Theorem 2. Now, we state the main theorem.

▶ **Theorem 6** (Main)**.** *Let (X,π) be a $(d+1)$-partite (weighted) totally connected complex. For any distinct $i,j\in[d]$, let $\epsilon_{\{i,j\}}=\max_{\tau:\mathrm{type}(\tau)=[d]\setminus\{i,j\}}\lambda_2(P_\tau)$ be the 2nd largest eigenvalue of the simple random walk matrices on (X_τ,π_τ) for all τ of type $[d]\setminus\{i,j\}$. For some $0<\delta<1$, assume that for every $i\in[d]$,*

$$\epsilon_{\{i,j\}}\cdot H_{\Delta-1} \le \frac{\delta^2}{10},\forall j\neq i \quad \textit{and} \tag{1}$$

$$\sum_{\ell=1}^{\Delta(i)}\epsilon_{\{i,j_\ell\}}\cdot H_{\Delta(i)-1}(\ell-1) \le 1-\delta, \tag{2}$$

where $j_0\ldots,j_d$ is an ordering of $[d]\setminus i$ such that $\epsilon_{\{i,j_0\}}\le\cdots\le\epsilon_{\{i,j_d\}}$. Then, (the link of the emptyset of) X is a $\frac{c(1-\delta)}{d\delta}$-expander for $c=\frac{2(1+\frac{\delta^2}{10})}{(1+\delta)}$.

▶ Remark 7. If, for some $\delta>0$, the conditions of the above theorem hold for a complex (X,π), then the conditions also hold for the same δ for all links (X_τ,π_τ) (of faces of co-dimension at least 2). Therefore, this theorem implies that X is $(\frac{c(1-\delta)}{\delta},\ldots,\frac{c(1-\delta)}{d\delta})$-local spectral expander for $c=\frac{2(1+\frac{\delta^2}{10})}{(1+\delta)}$. One can prove tighter bounds if they apply this theorem to any link (X_τ,π_τ) individually and possibly use better bounds on $\Delta_{(X_\tau,\pi_\tau)}(i)$.

Proof of Theorem 4. Fix a face τ of co-dimension k. For brevity we abuse notation and write Δ_τ denote $\Delta_{(X_\tau,\pi_\tau)}$. If $k\ge\Delta$ the statement follows from the above remark. In particular, for any $i,j\in[d]$

$$\begin{aligned}&\epsilon_{\{i,j\}}\cdot H_{\Delta_\tau-1}\le\gamma_2\cdot H_{\Delta-1}\le\gamma_2\cdot(1+\ln\Delta)\le\frac{\delta^2}{10},\\ &\sum_{\ell=1}^{\Delta_\tau(i)}\epsilon_{\{i,j_\ell\}}\cdot H_{\Delta_\tau(i)-1}(\ell-1)\le\gamma_2(\Delta+\ln\Delta)\le 1-\delta.\end{aligned}$$

So, we can apply Theorem 6.

Otherwise, to bound the spectral expansion of (X_τ, π_τ), let $\delta_k = 1-(1-\delta)\frac{k+\ln k}{\Delta+\ln\Delta} \geq \delta$. For $i,j \in [d]$

$$\epsilon_{\{i,j\}} \cdot H_{\Delta_\tau - 1} \leq \gamma_2 \cdot H_{k-1} \leq \frac{\delta^2 \cdot H_{k-1}}{10(1+\ln\Delta)} \underset{\delta\leq\delta_k}{\leq} \frac{\delta_k^2}{10},$$

$$\sum_{\ell=1}^{\Delta_\tau(i)} \epsilon_{\{i,j_\ell\}} \cdot H_{\Delta_\tau(i)-1} \underset{\epsilon_{i,j_\ell}\leq\gamma_2, \Delta_\tau(i)\leq k}{\leq} \gamma_2(k+\ln k) \leq \frac{(1-\delta)(k+\ln k)}{\Delta+\ln\Delta} = 1-\delta_k.$$

Therefore, applying Theorem 6 to (X_τ, π_τ), we obtain that (X_τ, π_τ) is a $\frac{c(1-\delta_k)}{k\delta}$-expander. ◀

Applications to Graph Coloring

Consider a graph $G = ([n], E)$ with degree function $\Delta : [n] \to \mathbb{Z}_{\geq 0}$ and maximum degree Δ, paired with a collection of color lists $\{L(i)\}_{i\in[n]}$ satisfying $L(i) \geq \Delta(i) + (1+\eta)\Delta$ for all $i \in [n]$ and for some $0 < \eta \leq 0.9$ such that $\frac{1+\ln\Delta}{\Delta} \leq \frac{\eta^2}{40}$. We define the $(n+1)$-partite coloring complex $X(G,L)$ specified by the following facets: $\{i, \sigma(i)\}_{i\in[n]}$ is a facet if and only if σ is a proper L-coloring of G, i.e. $\sigma(i) \in L(i)$ for each $i \in [n]$ and $\sigma(i) \neq \sigma(j)$ if $\{i,j\} \in E$. It is not hard to see that if $\{i,j\} \notin E$, then $\epsilon_{\{i,j\}} = 0$. Moreover, if $\{i,j\} \in E$, then $\epsilon_{\{i,j\}} \leq \frac{1}{(1+\eta)\Delta} + \frac{1}{(1+\eta)^2\Delta^2}$ (see Theorem 4.4 in [1]). Once can verify that if we apply the above theorem to the coloring complex $X(G,L)$ with $\delta = \frac{\eta}{2}$, we get that $X(G,L)$ is a $\left(\frac{4}{\eta}, \frac{4}{2\eta}, \ldots, \frac{4}{(|V|-1)\cdot\eta}\right)$-local spectral expander, and thus the Glauber dynamics for sampling a random proper coloring mixes in polynomial time. This retrieves (up to constants) a theorem proved in [1].

Applications to Sparse High Dimensional Expanders

Kaufman and Oppenheim [13] obtained a simple construction of sparse $(d+1)$-partite complexes with $|X(0)| \geq p^s$ for any integer $s > d$ and prime power p such that every $x \in X(0)$ is in at most $p^{O(d^3)}$ many facets (hence the degree is independent of s). They argued that for any non-consecutive pair of parts $i,j \in [d]$, i.e., $j \neq i+1$ and $i \neq j+1$ (mod $d+1$), we have $\epsilon_{\{i,j\}} = 0$ but $\epsilon_{\{i,i+1\}} \leq \frac{1}{\sqrt{p}}$ for any $i \in [d]$ ($i+1$ is taken modulo $d+1$). Consequently, $\Delta(i) = 2$ for any $i \in [d]$. Then, using Theorem 2, they show that the complex is a $(\frac{1}{\sqrt{p}-(d-2)}, \ldots, \frac{1}{\sqrt{p}-d-2})$-local spectral expander for $p > (d-2)^2$. Simply plugging in these values into the above theorem, for $\delta = 1 - \frac{2}{\sqrt{p}}$ and $p \geq 193$ (independent of d) the assumptions of the theorem are satisfied. The resulting complex is $(\frac{2c}{\sqrt{p}\delta}, \ldots, \frac{2c}{d\sqrt{p}\delta})$-local spectral expander for $c \approx 1.15$. In other words, not only does the Kaufman-Opennheim construction give a HDX for constant values of p independent of d, but also its local spectral expansion improves inverse linearly with the co-dimension.

O'Donnell and Pratt [17] constructed $(d+1)$-partite (sparse) high-dimensional expanders, with unbounded dimension d, via root systems of simple Lie Algebras, namely families A_d for $d \geq 1$, B_d for $d \geq 2$, C_d for $d \geq 3$ and D_d for $d \geq 4$. For explicit descriptions of these root systems, see e.g. [5, Sec. 3.6]. O'Donnell and Pratt [17] showed that, similar to the Kaufman-Oppenheim construction, the resulting d-dimensional complex X satisfies $|X(0)| \geq p^{\Theta(m)}$ whereas every vertex is only in $p^{\Theta(d^2)}$ many facets and for any $i,j \in [d]$, $\epsilon_{i,j} \leq \sqrt{2/p}$. Then, using Theorem 2 they concluded that the complex is a $(\frac{1}{\sqrt{p/2}-d+1}, \ldots, \frac{1}{\sqrt{p/2}-d+1})$-local spectral expander. Upon further inspection of the explicit set of roots, one can verify that $\Delta \leq 2$ for complexes based on A_d, B_d, C_d root systems and $\Delta \leq 3$ for the D_d root system. Plugging in these values in the above theorem and setting $\delta = 1 - 2\sqrt{2/p}$ for A_d, B_d, C_d

complexes and $\delta = 1 - 3.5\sqrt{2/p}$ for the D_d complex, if $p \geq 376$ for A_d, B_d, C_d complexes and $p \geq 729$ for the D_d complex, we get that these complexes are $(\frac{c'}{\sqrt{p}\delta}, \ldots, \frac{c'}{d\sqrt{p}\delta})$-local spectral expander for some constant $c' > 1$.

The well known Ramanujan complexes, also known as LSV complexes, are generalizations of Ramanujan graphs that were introduced by Lubotsky, Samuels, and Vishne in [14] and explicitly constructed in [16]. Any d-dimenssional LSV complex X that is q-thick for some fixed prime power q and $d \geq 2$ has a bounded degree (the number of facets that contain each $x \in X(0)$ only deponents on q and d, and is constant in the size of the ground set n which can be arbitrarily large) (e.g. see [12]). Moreover, the link of every proper face of type S is a spherical building complex in which $\Delta(i) = |\{j \neq i : \epsilon_{\{i,j\}} > 0\}|$ is at most 2 for every $i \in [d] \setminus S$. Furthermore, the worst expansion among links co-dimension 2 is $\frac{c}{\sqrt{q}}$, for some constant c independent of q, d, n. So, there is a constant q_0 such that if $q \geq q_0$, Theorem 6 implies that the link of any (proper) face of X of co-dimension k is a $\frac{c'}{(k-1)\sqrt{q}}$-spectral expander for some constant $c' > 0$ independent of q, d, n. This improves over the bound $\frac{C(d)}{\sqrt{q}}$ proved in [12], where $C(d) \geq 2^d(d+1)!$.

1.2 Proof Overview

At a high-level, our proof builds on the matrix trickle down framework introduced in the work of the authors with Liu [1]. The Oppenheim's trickle down theorem follows from an inductive argument that derives a bound on the second eigenvalue of the simple walk on 1-skeleton of each link (X_τ, π_τ) using the largest second eigenvalue of the simple walk on the 1-skeleton of links $(X_{\tau'}, \pi_{\tau'})$ for all faces $\tau' \supset \tau$ of size $|\tau| + 1$. The reason that one has to take the largest 2nd eigenvalue as opposed to the average in each inductive step is that the eigenspaces of these simple walks are very different. The matrix trickle down framework overcomes this issue by substituting the scalar bounds on the second eigenvalues with matrices that upper bound the transition probability matrices of the simple walks on the 1-skeletons of links. However, as opposed to Oppenhiem's trickle down theorem, the matrix trickle down framework cannot be applied in a black-box manner to bound the spectral expansion of the 1-skeletons of all links only by bounding the spectral expansion of the 1-skeletons of links of faces of co-dimension 2. The main result of this paper can be seen as applying the matrix trickle down framework with a carefully chosen set of upper-bound matrices to prove an improved trickle down theorem for partite complexes that can be applied in the same black-box fashion, just known an "average" second eigenvalue.

Our technical contribution in this paper are twofold: First, we observe that for any two disjoint sets of parts $S, T \subseteq [d]$, if the links of all faces of co-dimension 2 whose types intersect with both S, T are 0-spectral expanders, then for any $\sigma \in X$ of type S and τ of type T we get

$$\mathbb{P}_{\eta\sim\pi}[\sigma \subset \eta | \tau \subset \eta] = \mathbb{P}_{\eta\sim\pi}[\sigma \subset \eta] \quad \text{and} \quad \mathbb{P}_{\eta\sim\pi}[\tau \subset \eta | \sigma \subset \eta] = \mathbb{P}_{\eta\sim\pi}[\tau \subset \eta],$$

namely, the conditional distributions on these types are independent (see Lemma 18 for details). This observation significantly simplifies invoking the Matrix trickle down framework. Armed with this tool, we invoke the matrix trickle down theorem using a carefully chosen family of (diagonal) matrices as the matrix bounds. These matrices are recursively defined based on an "average" of the spectral expansions of the links of all faces of co-dimension 2, See the proof of Theorem 6 for the construction of these matrices.

2 Preliminaries

For any integer $n \geq 0$, let $[n] = \{0, \ldots, n\}$. When it is clear from context, we write x to denote a singleton $\{x\}$. Given a set S, we write $v \in \mathbb{R}^S$ and $M \in \mathbb{R}^{S \times S}$ to respectively denote a vector and a matrix indexed by S. Given a probability distributions μ over a set S, we may view μ as a vector in $\mathbb{R}^S_{\geq 0}$. For a $n \times n$ matrix M with eigenvalues $\lambda_1, \ldots, \lambda_n$, define $\rho(A) = \max_{1 \leq i \leq n} |\lambda_i|$.

Graphs

Given a graph $G = (V, E)$, for any $v \in V$, let $\Delta_G(v)$ be the degree of v in G, and let Δ_G be the maximum degree of G. Moreover, given a subset $S \subseteq V$, $G[S]$ denotes the induced subgraph of G on the set of vertices S. For any $S \subseteq V$, define $G_S = G[V \setminus S]$. For simplicity of notation, when G is clear from context, we denote $\Delta_G(v)$ by $\Delta(v)$ for any $v \in V$, and for any $S \subseteq V$, we denote $\Delta_{G_S}(v)$ by $\Delta_S(v)$ for any $v \in V \setminus S$. Similarly, we denote the maximum degree of G and G_S by Δ and Δ_S respectively. Moreover, when G is clear from context, we write $u \sim v$ if u, v are adjacent vertices in G and $u \sim_S v$ if $u, v \in V \setminus S$ and $u \sim v$.

We say that a graph $G = (V, E)$ paired with a weight function $w : E \to \mathbb{R}_{\geq 0}$ is ϵ-expander if $\lambda_2(P) \leq \epsilon$, where $P \in \mathbb{R}^{V \times V}$ is the transition probability matrix of the simple random walk on (G, w) defined as $P(x, y) = \frac{w(\{x,y\})}{\sum_z w(\{x,z\})}$ for any $x, y \in V$. For such a graph we write $d_w(x) = \sum_{y \sim x} w(\{x, y\})$ to denote the weighted degree of a vertex x and $\mathrm{vol}(S) = \sum_{x \in S} d_w(x)$ to denote the volume of a set $S \subseteq V$.

2.1 Linear Algebra

▶ **Lemma 8** (Cheeger's Inequality). *For any graph $G = (V, E)$ with weights $w : E \to \mathbb{R}_{\geq 0}$ and any $S \subseteq V$,*

$$\frac{w(E(S, \overline{S}))}{\min\{\mathrm{vol}(S), \mathrm{vol}(\overline{S})\}} \leq \sqrt{2(1 - \lambda_2)}$$

where λ_2 is the second largest eigenvalue of the simple random walk on G

▶ **Lemma 9** (Expander Mixing Lemma). *Given a (weighted) graph $G = (V, E, w)$, for any set $S \subseteq V$,*

$$\left| w(E(S)) - \frac{\mathrm{vol}(S)^2}{\mathrm{vol}(V)} \right| \leq \lambda_2 \mathrm{vol}(S),$$

where λ_2 is the second largest eigenvalue of the simple random walk on G.

2.2 Simplicial Complexes

We say that a simplicial complex X is *gallery connected* if for any face τ of co-dimension at least 2 and any pair of facets σ, σ' of X_τ there is a sequence of facets of X_τ, $\sigma = \sigma_0, \sigma_1, \ldots, \sigma_\ell = \sigma'$, such that for all $0 \leq i < \ell$, $|\sigma_i \Delta \sigma_{i+1}| = 2$. It is shown in [18, Prop 3.6] that if X is totally connected, then it is gallery connected.

▶ **Lemma 10.** *Consider a totally connected $(d+1)$-partite complex X with parts indexed by $[d]$. For any $S \subseteq [d]$, The induced subgraph of the 1-skeleton of X on vertices of type S is connected.*

Proof. Take x, y of type $i, j \in S$ and facets η, η' such that $x \in \eta$, $y \in \eta'$. Total connectivity implies that there is a sequence $\eta = \eta_1, \ldots, \eta_t = \eta'$ such that $\eta_i \cap \eta_{i+1} \neq \emptyset$ for all $1 \leq i \leq t-1$. Let $\sigma_1 \subseteq \eta_1, \ldots, \sigma_t \subseteq \eta_t$ be faces of type $\{i, j\}$. Then $\sigma_1, \ldots, \sigma_t$ gives a path between x, y. ◀

Given a (weighted) complex (X, π), for integer $-1 \leq i \leq \dim(X) - 1$, π induces a distribution π_i on $X(i)$,

$$\pi_i(\sigma) = \frac{1}{\binom{\dim(X)+1}{i+1}} \Pr_{\tau \sim \pi}[\sigma \subset \tau] \quad \forall \sigma \in X(i).$$

Let $P_{(X,\pi),\tau} \in \mathbb{R}^{X(0) \times X(0)}$ denote the transition probability matrix of the simple random walk on the 1-skeleton of (X_τ, π_τ) padded with zeros outside the $X_\tau(0) \times X_\tau(0)$ block, i.e. $P_{(X,\pi),\tau}(x, y) = \frac{\mathbb{P}_{\sigma \sim \pi_\tau}[\{x,y\} \subset \sigma]}{\sum_{z \in x_\tau(0)} \mathbb{P}_{\sigma \sim \pi_\tau}[\{x,z\} \subset \sigma]}$ for $x, y \in X_\tau(0)$, and $P_\tau(x, y) = 0$ otherwise. When the weighted complex (X, π) is clear from context, we write P_τ to denote $P_{(X,\pi),\tau}$. For any τ of co-dimension at least 2, we define the diagonal matrix $\Pi_{(X,\pi),\tau} \in \mathbb{R}^{X(0) \times X(0)}$ as follows: $\Pi_{(X,\pi),\tau}(x, x) = \pi_{\tau,0}(x)$ for $x \in X_\tau(0)$, and $\Pi_{(X,\pi),\tau}(x, x) = 0$ otherwise. When (X, π) is clear from context, we write Π_τ to denote $\Pi_{(X,\pi),\tau}$. Note that $\Pi_\tau P_\tau$ is a symmetric matrix.

Given a $(d+1)$-partite complex,

we say that an $x \in X(0)$ is of type i and write $\text{type}(x) = i$ if $x \in T_i$. Similarly, the type of a face $\tau \in X$ is defined as $\text{type}(\tau) = \{i \in [d] : |\tau \cap T_i| = 1\}$. The following facts hold for weighted partite complexes.

▶ **Observation 11.** *Consider a weighted $(d+1)$ partite complex (X, π) and a face τ of co-dimension $k \geq 1$. We have $k\pi_{\tau,0}(x) = \Pr_{\sigma \sim \pi_\tau}[x \in \sigma]$ for all $x \in X_\tau(0)$.*

▶ **Observation 12.** *Consider a weighted $(d+1)$ partite complex (X, π) with parts indexed by $[d]$ and a face τ of co-dimension $k \geq 1$. For any $i \in [d]$, $\sum_{x:\text{type}(x)=i} \Pr_{\sigma \sim \pi_\tau}[x \in \sigma] = 1$.*

The following definition is useful for proving the main theorem.

▶ **Definition 13.** *For any $(d+1)$-partite complex (X, π) with parts indexed by $[d]$, define a graph $G_{(X,\pi)}$ on the set of vertices $[d]$, where any distinct $i, j \in [d]$ are adjacent in $G_{(X,\pi)}$ if there exists τ of type $[d] \setminus \{i, j\}$ such that the second eigenvalue of (X_τ, π_τ) is positive.*

▶ Remark 14. For any $(d+1)$-partite complex (X, π) with parts indexed by $[d]$, for every $i \in [d]$, $\Delta(i)$ (see Definition 3) is the degree of i in graph $G_{(X,\pi)}$ and Δ is the maximum degree of $G_{(X,\pi)}$.

Note that if $\text{codim}(\tau) = k$, the link X_τ is a k-partite complex with parts indexed by $[d] \setminus S$. One can verify that given a face τ of type S, the set of edges of $G_{(X_\tau,\pi_\tau)}$ is a subset of the edges of $(G_{(X,\pi)})_S$, i.e., the induced subgraph of $G_{(X,\pi)}$ on $[d] \setminus S$. When (X, π) is clear from context, we write G for $G_{(X,\pi)}$ and G_S for $(G_{(X,\pi)})_S$.

Product of Weighted Complexes

Given weighted complexes $(Y_1, \mu_1), \ldots, (Y_\ell, \mu_\ell)$ defined on disjoint ground sets and of dimensions $d_1, \ldots, d_\ell$ respectively, and a weighted complexes (X, π) of dimension d, we write $(X, \pi) = (Y_1, \mu_1) \times \cdots \times (Y_\ell, \mu_\ell)$ if $X(d) = \{\cup_{i \in [\ell]} \tau_i : \tau_1 \in Y_1(d_1), \ldots, \tau_\ell \in Y_\ell(d_\ell)\}$ and $\pi(\cup_{i \in [\ell]} \tau_i) = \prod_{i \in [\ell]} \mu_i(\tau_i)$ for all $\tau_1 \in Y_1(d_1), \ldots, \tau_\ell \in Y_\ell(d_\ell)$. We denote the generating polynomial of (X, π) by $g_{(X,\pi)}$, i.e. $g_{(X,\pi)} = \sum_{\tau \in X(d)} \pi(\tau) \prod_{x \in \tau} z_x$. One can verify that $(X, \pi) = (X_1, \mu_1) \times \cdots \times (X_\ell, \mu_\ell)$ if and only if $g_{(X,\pi)} = g_{(X_1,\mu_1)} \times \cdots \times g_{(X_\ell,\mu_\ell)}$. Note that this is true because we assume that for any weighted simplicial complex, the given distribution on facets is non-zero on all facets.

Matrix Trickle Down Theorem

We use the following theorem which is the main technical theorem in [1].

▶ **Theorem 15** ([1, Thm III.5]). *Let (X,π) be a totally connected weighted complex. Suppose $\{M_\tau \in \mathbb{R}^{X(0)\times X(0)}\}_{\tau\in X(\le d-2)}$ is a family of symmetric matrices satisfying the following:*

1. **Base Case:** *For every τ of co-dimension 2, we have the spectral inequality*

$$\Pi_\tau P_\tau - 2\pi_{\tau,0}\pi_{\tau,0}^\top \preceq M_\tau \preceq \frac{1}{5}\Pi_\tau.$$

2. **Recursive Condition:** *For every τ of co-dimension at least $k\ge 3$, at least one of the following holds: M_τ satisfies*

$$M_\tau \preceq \frac{k-1}{3k-1}\Pi_\tau \quad \text{and} \quad \mathbb{E}_{x\sim\pi_\tau} M_{\tau\cup\{x\}} \preceq M_\tau - \frac{k-1}{k-2}M_\tau\Pi_\tau^{-1}M_\tau. \tag{3}$$

Or, $(X_\tau,\pi_{\tau,k-1})$ is a product of weighted simplicial complexes $(Y_1,\mu_1),\dots,(Y_t,\mu_t)$ and for every $\eta\in X_\tau(k-1)$,

$$M_\tau = \bigoplus_{1\le i\le t:d_{Y_i}\ge 1} \frac{d_{Y_i}(d_{Y_i}+1)}{k(k-1)} M_{\tau\cup\eta_{-i}},$$

where $\eta_{-i}=\eta\setminus Y_i(0)$.

Then for every $\tau\in X(\le d-2)$, we have the bound $\lambda_2(\Pi_\tau P_\tau)\le \rho(\Pi_\tau^{-1}M_\tau)$.

3 Simplifying Matrix Trickle Down's Conditions to Scalar Inequalities

In this section, given a $(d+1)$-partite complex (X,π), we apply the matrix trickle down theorem to derive a set of conditions on a family of vectors $\{f_S\in\mathbb{R}^{[d]}\}_{S\subset[d],|S|<d}$ that will guarantee that $\lambda_2(P_\tau)\le \frac{\max_{i\in[d]} f_S(i)}{k-1}$ for all $k\ge 2$ and τ of co-dimension k and type S. We prove the following theorem.

▶ **Theorem 16.** *Consider a totally connected $(d+1)$-partite complex (X,π) with parts indexed by $[d]$ and graph $G=G_{(X,\pi)}$. Suppose we are given a family of vectors $\{f_S\in\mathbb{R}^{[d]}\}_{S\subset[d],|S|<d}$ such that for all $S\subset[d]$ of size $(d+1)-k$, the support of f_S is a subset of $[d]\setminus S$, and the following holds:*

- *If G_S is disconnected, then $f_S=\sum_{1\le i\le \ell:|I_i|\ge 2} f_{[d]\setminus I_i}$, where $I_1\cup\dots\cup I_\ell$ are the vertices of the connected components of G_S. Note that if all connected components are of size 1, then $f_S=0$.*
- *Otherwise if G_S is connected, we have $\max_{i\in[d]} f_S(i)\le \frac{(k-1)^2}{3k-1}$ and*

(i) *Base Case: If $k=2$, then for every face τ of type S, $\lambda_2(P_\tau)\le \max_{i\in[d]\setminus S} f_S(i)$.*

(ii) *Recursive Condition: If $k\ge 3$, then*

$$\sum_{j\in[d]\setminus(S\cup i)} f_{S\cup j}(i) \le (k-2)f_S(i) - f_S^2(i),$$

for all $i\in[d]\setminus S$.

Then, for all $k\ge 2$ and τ of co-dimension k and type S, $\lambda_2(P_\tau)\le \frac{\max_{i\in[d]} f_S(i)}{k-1}$.

The main sets of conditions in the above theorem are the inequalities in Item i and Item ii. To get some intuition about these conditions, it is helpful to compare the above with the standard trickle down theorem (Theorem 2). There, one shows that if $\lambda_2(P_{\tau\cup\{x\}})\le\lambda$ for all $x\in X_\tau(0)$, then $\lambda_2(P_\tau)\le\alpha$, where satisfies

$$\lambda \le \alpha - \alpha^2(1-\lambda). \tag{4}$$

Then, Theorem 2 follows by recursively applying this inequalities.

In the above theorem, instead of a single upper bound on $\lambda_2(P_\tau)$ for faces τ of co-dimension 2, one bounds the expansion of the links of all faces of co-dimension 2 of each type separately, allowing higher degrees of freedom. For any face τ of type S and co-dimension $k = |S|$, the function $\frac{f_S(.)}{k-1}$ will serve as the digonal entries of a matrix upper-bound P_τ.

Then, the inequality $\frac{\sum_{j\in[d]\setminus(S\cup i)} f_{S\cup j}(i)}{k-2} \leq f_S(i) - \frac{f_S^2(i)}{k-2}$ is the natural analogue of (4) which requires f_S to be at least "the average" of $f_{S\cup j}$ for all $j \in [d] \setminus S$ plus an square error term.

Before proving the above theorem, we show that if G_S is disconnected with parts $G[I_1], \ldots, G[I_\ell]$ for some $S \subset [d]$ of size at most $d-1$, then for any τ of type S, $(X_\tau, \pi_{\tau,k-1})$ can be written as a product of family of its links of types $[d] \setminus I_i$ for all $1 \leq i \leq \ell$. This allows us to prove a better upper-bound on $\lambda_2(P_\tau)$ for such faces τ by simply "concatenating" upper-bounds on each connected component of G_S.

▶ **Lemma 17.** *Consider a 2-partite complex (X, π) with parts S, T. If (X, π) is 0-expander, then $(X, \pi) = (X_z, \pi_z) \times (X_y, \pi_y)$ for any $y \in S$ and $z \in T$.*

Proof. Note that (X, π) is a weighted bipartite graph with parts S, T. Let $A \in \mathbb{R}^{X(0)\times X(0)}$ be the adjacency matrix of (X, π). Let $A_{S,T}(y, z) = A(y, z)$ for $y \in S$, $z \in T$ and 0 on other entries. Moreover, let $A_{T,S} = A - A_{S,T}$. Then, for any vector $v \in \mathbb{R}^{X(0)}$, we get $A = A_{S,T}v_T + A_{T,S}v_S$, where v_S, v_T are respectively supported on S, T and $v = v_S + v_T$. Thus, if $Av = \lambda v$, then $Av' = -\lambda v'$, for $v' = (-v_S + v_T)$. So if μ is an eigenvalue of A, then $-\mu$ is also an eigenvalue of A. Thus, if (X, π) is 0-expander, the rank of A is 2. This implies that there are vectors $w_S \in \mathbb{R}^S$ and $w_T \in \mathbb{R}^T$ such that $\pi(\{y, z\}) = A(y, z) = A(z, y) = w_S(y)w_T(z)$ for $y \in S$, $z \in T$. Without loss of generality, assume $\|w_S\|_1 = \|w_T\|_1 = 1$. Then, for any $y \in S$ and $z \in T$, we have $\pi_z(y) = \frac{\pi(\{y,z\})}{\sum_{x\in S}\pi(\{x,z\})} = w_S(y)$. Similarly $\pi_y(z) = w_T(z)$. Thus $\pi(\{y, z\}) = \pi_y(z)\pi_z(y)$. This finishes the proof. ◀

▶ **Lemma 18.** *Consider a totally connected $(d+1)$-partite complex (X, π) with parts indexed by $[d]$ and its associated graph $G = G_{(X,\pi)}$. Let $I_1 \cup \cdots \cup I_\ell$ be a partition of $[d]$ such that for any $1 \leq i \leq \ell$ the induced graph $G[I_i]$ is a connected component or the union of several connected components of G. Then $(X, \pi) = (X_{\sigma_{-1}}, \pi_{\sigma_{-1}}) \times \cdots \times (X_{\sigma_{-\ell}}, \pi_{\sigma_{-\ell}})$, where σ_{-i} is an arbitrary face of type $[d] \setminus I_i$ for any $1 \leq i \leq \ell$.*

Proof. We prove the statement by induction on d. For $d = 1$, the statement simply follows from Lemma 17. Now, assume that $d > 1$. If $|I_i| = 1$ for all $1 \leq i \leq \ell$, then $\ell \geq 3$. In this case, let $S = I_1 \cup I_2$. Otherwise, WLOG assume that $|I_1| \geq 2$ and let $S = I_1$. First, we show that $g_{(X,\pi)}$ can be written as $g_{(X,\pi)} = h \cdot h'$, where h is a polynomial in $\{z_y : \text{type}(y) \in I \setminus S\}$ and h' is a polynomial in terms of variables in $\{z_y : \text{type}(y) \in S\}$. By induction hypothesis, for any $i \in S$, $x \in T_i$, and any face $\sigma \in X$ of type S such that $x \in \sigma$

$$\partial_{z_x} g_{(X,\pi)} = f^x \cdot g^x \tag{5}$$

where f^x is a polynomial in terms of variables in $\{z_y : \text{type}(y) \in S \setminus i\}$ and g^x is a polynomial in terms of variables in $\{z_y : \text{type}(y) \in I \setminus S\}$. Now, take arbitrary $i, j \in S$ such that $i \neq j$. Then, (5) implies that for any face $\{x, y\}$ of type $\{i, j\}$

$$\partial_{z_x}\partial_{z_y} g_{(X,\pi)} = (\partial_{z_y} f^x)g^x = (\partial_{z_x} f^y)g^y$$

It thus follows that g^x is a multiple of g^y. One can see this simply by substituting 1 for all variables in $\{z_y : \text{type}(y) \in S \setminus \{i, j\}\}$. Moreover, since g^x and g^y are generating polynomials of distributions, i.e. the coefficients sum up to 1, we get $g^x = g^y$. Therefore, we get that

for any distinct x, y such that $\text{type}(x), \text{type}(y) \in S$ and $\{x, y\}$ is a face, $g^x = g^y$. Applying Lemma 10, we get $g^x = g^y$ for all $x, y \in \cup_{i\in S} T_i$. Thus, there exist a polynomial h in variables $\{z_y : \text{type}(y) \in I \setminus S\}$ such that we can rewrite (5) for any x with $\text{type}(x) \in S$ as

$$\partial_{z_x} g_{(X,\pi)} = f^x \cdot h,$$

where f^x is a polynomial in terms of variables in $\{z_y : \text{type}(y) \in S \setminus i\}$. Finally, since X is a partite complex,

$$|S| g_{(X,\pi)} = \sum_{i\in S} \sum_{x\in T_i} z_x \partial_{z_x} g_{(X,\pi)} = h \cdot \sum_{i\in S} \sum_{x\in T_i} z_x f^x = h \cdot h', \tag{6}$$

where $h' = \sum_{i\in S} \sum_{x\in T_i} z_x f^x$ is a polynomial in $\{z_y : \text{type}(y) \in S\}$. It remains to show that for any face σ of type S, we have $h = g_{(X_\sigma,\pi_\sigma)}$, and for any τ of type $[d] \setminus S$, we have $h' = g_{(X_\tau,\pi_\tau)}$. Fix arbitrary faces σ of type S and τ of type $[d] \setminus S$. Noting that $g_{(X,\pi)}$ is a multiple of $h \cdot h'$, and that h' is in variables associated to elements whose types are in S and h is in variables associated to elements whose types are in $[d] \setminus S$, we conclude that h' has a monomial that is a multiple of $\prod_{x\in\sigma} z_x$ and h has a monomial that is a multiple of $\prod_{x\in\tau} z_x$. First, take $(\prod_{x\in\sigma} \partial_{z_x})$ from both sides of (6). We get that $g_{(X_\sigma,\pi_\sigma)}$ is a positive multiple of h. Similarly, taking $(\prod_{x\in\tau} \partial_{z_x})$ from both sides of (6), we get that $g_{(X_\tau,\pi_\tau)}$ is a positive multiple of h'. Thus, noting that the coefficients of generating polynomials sum up to 1, we get $h = g_{(X_\sigma,\pi_\sigma)}$ and $h' = g_{(X_\tau,\pi_\tau)}$ as desired. Repeating the same argument inductively on the complex (X_σ, π_σ) proves the claim. ◀

Now we are ready to prove Theorem 16.

Proof of Theorem 16. We apply Theorem 15. For every $S \subset [d]$ such that $|S| < d$, define a diagonal matrix $D_S \in \mathbb{R}^{X(0)\times X(0)}$ as $D_S(x,x) = f_S(\text{type}(x))$ for all $x \in X(0)$. We prove that the conditions of Theorem 15 hold for $M_\tau = \frac{\Pi_\tau D_S}{k-1}$ for an arbitrary face $\tau \in X$ of co-dimension at least $k \geq 2$ and type S. If G_S is connected, $\max_{i\in[d]} f_S(i) \leq \frac{(k-1)^2}{3k-1}$ holds by assumption. If G_S is disconnected, $\max_{i\in[d]} f_S(i) \leq \frac{(k-1)^2}{3k-1}$ follows from the assumptions that $f_S = \sum_{1\leq i\leq \ell:|I_i|\geq 2} f_{[d]\setminus I_i}$, where $I_1 \cup \cdots \cup I_\ell$ are the vertices of connected components of G_S. That is because the supports of vectors $f_{[d]\setminus I_i}$ are disjoint by assumption and $\frac{(k-1)^2}{3k-1}$ is an increasing function for $k \geq 2$. So, we get $D_\tau \preceq \frac{(k-1)^2}{3k-1} I$, and thus, $M_\tau \preceq \frac{k-1}{3k-1}\Pi_\tau$. To prove the rest of the conditions hold, first assume that $k = 2$. If G_S is two disconnected vertices, we get $f_S = 0$, and therefore, $D_S = 0$. Thus, we get $\Pi_\tau P_\tau - \pi_{\tau,0}\pi_{\tau,0}^\top \preceq 0 = \Pi_\tau D_S = M_\tau$, as desired. If G_S is connected, the base case assumption (Item i) implies that $\lambda_2(P_\tau) \leq D_S(x,x)$ for all $x \in X_\tau(0)$. Therefore, $\Pi_\tau P_\tau - \pi_{\tau,0}\pi_{\tau,0}^\top \preceq \Pi_\tau D_S = M_\tau$. Now, assume $k \geq 3$. First assume that G_S is disconnected and $G[I_1], \ldots, G[I_\ell]$ are its connected components for some partition $I_1 \cup \cdots \cup I_\ell$ of $[d] \setminus S$. Fix any $\sigma \in X_\tau(k-1)$. By Lemma 18, $(X_\tau, \pi_\tau) = (X_{\tau\cup\sigma_{-1}}, \pi_{\tau\cup\sigma_{-1}}) \times \cdots \times (X_{\tau\cup\sigma_{-\ell}}, \pi_{\tau\cup\sigma_{-\ell}})$ where for every $1 \leq j \leq \ell$, σ_{-j} is a subset of σ that has type $[d] \setminus (S \cup I_j)$. Therefore, we get $\Pr_{\eta\sim\pi_{\tau\cup\sigma_{-j}}}[x \in \eta] = \Pr_{\eta\sim\pi_\tau}[x \in \eta]$ for all $1 \leq j \leq \ell$ and $x \in X_{\tau\cup\sigma_{-j}}(0)$. Combining this with Observation 11, we get $k_j \cdot \pi_{\tau\cup\sigma_{-j},0}(x) = k \cdot \pi_{\tau,0}(x)$, where $k_j = |I_j|$ for all $1 \leq j \leq \ell$. Thus we can write

$$\begin{aligned}
\sum_{1\leq j\leq \ell:|I_j|\geq 2} \frac{(k_j-1)k_j}{(k-1)k} M_{\tau\cup\sigma_{-j}} &\overset{\text{def of } M_{\tau\cup\sigma_{-j}}}{=} \sum_{1\leq j\leq \ell:|I_j|\geq 2} \frac{(k_j-1)k_j}{(k-1)k} \frac{\Pi_{\tau\cup\sigma_{-j}} D_{[d]\setminus I_j}}{k_j-1} \\
&= \sum_{1\leq j\leq \ell:|I_j|\geq 2} \frac{k_j}{k(k-1)} \frac{k}{k_j} \Pi_\tau D_{[d]\setminus I_j} \\
&= \frac{\Pi_\tau}{k-1} \sum_{1\leq j\leq \ell:|I_j|\geq 2} D_{[d]\setminus I_j} = \frac{\Pi_\tau D_S}{k-1} = M_\tau,
\end{aligned}$$

where in the second to last equality, we used the fact that $\sum_{1\le j\le \ell:|I_j|\ge 2} f_{[d]\setminus I_j} = f_S$, and thus $\sum_{1\le j\le \ell:|I_j|\ge 2} D_{[d]\setminus I_j} = D_S$ by definition of D_S. Now, assume that G_S is connected. It is enough to show that $\mathbb{E}_{x\sim\pi_{\tau,0}} M_{\tau\cup x} \preceq M_\tau - M_\tau \Pi_\tau^{-1} M_\tau$. This is equivalent to showing that for any $x \in X_\tau(0)$

$$\mathbb{E}_{y\sim\pi_{\tau,0}}\left[\frac{(\Pi_\tau^{-1}\Pi_{\tau\cup y} D_{S\cup \mathrm{type}(y)})(x,x)}{k-2}\right] \le \frac{D_S(x,x)}{k-1} - \frac{D_S^2(x,x)}{(k-2)(k-1)} \tag{7}$$

One can check that for any $x \in X_\tau(0)$ of type i

$$\begin{aligned}
\mathbb{E}_{y\sim\pi_{\tau,0}}\left[\frac{\Pi_\tau^{-1}\Pi_{\tau\cup y} D_{S\cup \mathrm{type}(y)}(x,x)}{k-2}\right] &= \frac{\sum_{y\in X_{\tau\cup x}(0)} \Pr_{\sigma\sim\pi_{\tau\cup x}}[y\in\sigma] D_{S\cup\mathrm{type}(y)}(x,x)}{(k-1)(k-2)} \\
&= \sum_{j\in[d]\setminus S} \frac{f_{\tau\cup j}(i)}{(k-1)(k-2)} \sum_{y\in X} \sum_{\substack{\tau\cup x(0):\\ \mathrm{type}(y)=j}} \Pr_{\sigma\sim\pi_{\tau\cup x}}[y\in\sigma] \\
&= \frac{\sum_{j\in[d]\setminus S} f_{\tau\cup j}(i)}{(k-1)(k-2)},
\end{aligned}$$

where in the last equality, we used Observation 12. Thus, substituting $D_S(x,x) = f_S(\mathrm{type}(x))$ in the RHS of (7), it is enough to show that for any $i \in [d]\setminus S$

$$\frac{\sum_{j\in[d]\setminus S} f_{\tau\cup j}(i)}{(k-1)(k-2)} \le \frac{f_S(i)}{k-1} - \frac{f_S^2(i)}{(k-1)(k-2)},$$

which holds by assumption Item ii. ◀

4 Proof of Main Theorem

We are ready to prove Theorem 6.

Proof of Theorem 6. We find a family of vectors $\{f_S \in \mathbb{R}^{[d]}\}_{S\subset[d]:|S|<d}$ that satisfy the conditions of theorem Theorem 16. Let $G = G_{(X,\pi)}$. Based on the conditions of Theorem 16, vectors $\{f_S \in \mathbb{R}^{[d]}\}_{S\subset[d]:|S|<d}$ can be defined as functions of $\{\epsilon_{\{i,j\}}\}_{i,j\in[d],i\neq j}$. Recall that edges of G capture pairs $\{i,j\}$ for which $\epsilon_{\{i,j\}} > 0$. Assign every edge $\{i,j\}$ of G with weight $\epsilon_{\{i,j\}}$. We restrict our attention to functions that are very local with respect to G, i.e. for every S and $i \in [d]\setminus S$, we assume $f_S(i)$ only depends on $\Delta_S(i)$ and the weights of edges adjacent to i in G_S if $\Delta(i) > 1$. It turns out that if $\Delta(i) = 1$, we would need to also take into account the degree of the unique neighbor of i. More formally, consider the following family of vectors $\{f_S \in \mathbb{R}^{[d]}\}_{S\subset[d]:|S|<d}$: for any $S \subset [d]$ such that $|S| < d$, let f_S be of the following form: for any $i \in S$, let $f_S(i) = 0$, and for any $i \in [d]\setminus S$ define

$$f_S(i) = \begin{cases} 0 & \text{if } \Delta_S(i) = 0, \\ \epsilon_{\{i,j\}} \cdot g_{i,j}(\Delta_S(j)) & \text{if } \Delta_S(i) = 1 and i \sim_S j, \\ \sum_{j\sim_S i} \epsilon_{\{i,j\}} \cdot h_i\left(\Delta_S(i)\right) & \text{if } \Delta_S(i) \ge 2, \end{cases}$$

where for every $i \in [d]$ and $j \sim i$, functions $g_{i,j}, h_i : \{1,\dots,\Delta\} \to \mathbb{R}_{\ge 0}$ are defined later in a way that guarantees that $\{f_S\}_{S\subset[d]:|S|<d}$ satisfies the assumptions of Theorem 16 (see (10), (12)).

First, consider the case that G_S is disconnected. Note that for any $S, S' \subset [d]$ such that $|S|, |S'| < d$, if $\{j \in [d] : j \sim_S i\} = \{j \in [d] : j \sim_{S'} i\}$ for some $i \notin S, S'$, then $f_S(i) = f_{S'}(i)$. Let $I_1, \ldots, I_\ell$ be the vertices of connected components of G_S. Since the neighborhood of each vertex in any connected component of G_S is the same as its neighborhood in G_S, we get $f_S = \sum_{1 \le i \le \ell : |I_i| \ge 2} f_{[d]\setminus I_i}$.

Now, assume G_S is connected. Take an arbitrary $k \ge 2$ and $S \subset [d]$ of size $(d+1)-k$. First we verify the set of conditions given in Item i and Item ii. First, assume that $k = 2$. Let $[d] \setminus S = \{i, j\}$. By definition of $\epsilon_{\{i,j\}}$, for any τ of type S, $\lambda_2(P_\tau) \le \epsilon_{\{i,j\}}$. Thus, if we define $g_{\ell,t}(1) = 1$ for all distinct $\ell, t \in [d]$, then we get $\lambda_2(P_\tau) \le \epsilon_{\{i,j\}} = \epsilon_{\{i,j\}} g_{i,j}(1) = f_S(i) = f_S(j)$, as desired. Now, assume that $k \ge 3$. Fix an arbitrary $i \in [d] \setminus S$. Our goal is to define $g_{i,j}, h_i : \{1, \ldots, \Delta\} \to \mathbb{R}_{\ge 0}$ for all $j \sim i$ such that $g_{i,j}(1) = 1$ for all $j \sim i$ and the following inequality is satisfied:

$$\sum_{j \in [d]\setminus(S \cup i)} f_{S\cup j}(i) \le (k-2) f_S(i) - f_S^2(i). \tag{8}$$

To keep the notation concise, relabel the elements such that i is relabeled to 0 and $\epsilon_{\{0,1\}} \ge \cdots \ge \epsilon_{\{0,d\}}$. Moreover, define $\epsilon_j = \epsilon_{\{0,j\}}$ for any $j \in [d] \setminus 0$.

Case 1: $\Delta_S(0) = 1$, and $j \sim_S 0$. Since G_S is connected and $(d+1) - |S| \ge 3$, we have $\Delta_S(j) \ge 2$. Define $t = \Delta_S(j)$. We have

$$\begin{aligned}\sum_{\ell \in [d]\setminus(S\cup 0)} f_{S\cup\ell}(0) &= f_{S\cup j}(0) + \sum_{\ell \in [d]\setminus(S\cup 0) : \ell \sim_S j} f_{S\cup\ell}(0) + \sum_{\ell \in [d]\setminus(S\cup 0) : \ell \not\sim_S j, \ell \ne j} f_{S\cup\ell}(0) \\ &= 0 + (t-1)\cdot\epsilon_j \cdot g_{0,j}(t-1) + (k-t-1)\cdot\epsilon_j \cdot g_{0,j}(t).\end{aligned}$$

On the other hand, $(k-2) f_S(0) - f_S(0)^2 = (k-2)\cdot\epsilon_j \cdot g_{0,j}(t) - \epsilon_j^2 \cdot g_{0,j}^2(t)$. So it is enough to satisfy

$$(t-1)\cdot\epsilon_j \cdot (g_{0,j}(t) - g_{0,j}(t-1)) \ge \epsilon_j^2 \cdot g_{0,j}^2(t). \tag{9}$$

Now, define $g_{0,j} : \{1, \ldots, \Delta\} \to \mathbb{R}_{\ge 0}$ as follows: recall that we defined $g_{0,j}(1) = 1$. For any $2 \le \ell \le \Delta$, let

$$g_{0,j}(\ell) = 1 + 1.3 \cdot \epsilon_j \cdot H_{\ell-1}. \tag{10}$$

Using assumption (1), $\epsilon_j H_{\Delta-1} \le \frac{\delta^2}{10} \le \frac{1}{10}$. Thus

$$\epsilon_j^2 \cdot g_{0,j}^2(t) \le \epsilon_j^2 \left(1 + 1.3\epsilon_j \left(1 + H_{\Delta-1}\right)\right)^2 < 1.3\epsilon_j^2.$$

Substituting $g_{0,j}(t)$ according to (10) and using the above bound, one can verify that (9) holds.

Case 2: $\Delta_S(0) \ge 2$. For simplicity of notation, define $t = \Delta_S(0)$ and $\alpha = \sum_{j : j \sim_S 0} \epsilon_j$. Define $h_0(1) = \max_{j : j \sim 0} g_{0,j}(\Delta)$.

$$\begin{aligned}\sum_{j \in [d]\setminus(S\cup 0)} f_{S\cup j}(0) &= \sum_{j \in [d]\setminus(S\cup 0) : j \sim_S 0} f_{S\cup j}(0) + \sum_{j \in [d]\setminus(S\cup 0) : j \not\sim_S 0} f_{S\cup j}(0) \\ &\le \left(\sum_{j \in [d]\setminus(S\cup 0) : j \sim_S 0} (\alpha - \epsilon_{\{0,j\}})\right) \cdot h_0(t-1) + (k-t-1)\cdot\alpha\cdot h_0(t) \\ &= (t-1)\cdot\alpha\cdot h_0(t-1) + (k-t-1)\cdot\alpha\cdot h_0(t).\end{aligned}$$

Note that if $t \geq 3$, the first inequality is an equality by definition. If $t = 2$, the first inequality follows from the definition of $h_0(1)$. Thus, it is enough to satisfy

$$\sum_{j \in [d] \setminus (S \cup 0)} f_{S \cup j}(0) = (t-1) \cdot \alpha \cdot h_0(t-1) + (k-t-1) \cdot \alpha \cdot h_0(t)$$
$$\leq (k-2) \cdot \alpha \cdot h_0(t) - \alpha^2 \cdot h_0^2(t) = (k-2) f_S(0) - f_S^2(0).$$

Equivalently, it suffices to satisfy

$$(t-1)(h_0(t) - h_0(t-1)) \geq \alpha \cdot h_0^2(t). \tag{11}$$

Now, define $h_0 : \{1, \ldots, \Delta\} \to \mathbb{R}_{\geq 0}$ as follows: recall that we defined $h_0(1) = \max_{j : j \sim 0} g_{0,j}(\Delta)$. For any $2 \leq \ell \leq \Delta$, define

$$h_0(\ell) = \frac{h_0(1)}{1 - c\left(\sum_{j=1}^{\ell} \epsilon_j H_{\ell-1}(j-1)\right)}. \tag{12}$$

We need to prove (11) for a carefully chosen c. Let β be such that $h_0(t) = \frac{h_0(1)}{\beta}$. We get $h_0(t-1) = \frac{h_0(1)}{\beta + c(\sum_{j=1}^{t} \frac{\epsilon_j}{t-1})}$, and thus,

$$(t-1)(h_0(t) - h_0(t-1)) = \frac{h_0(1) \cdot c \sum_{j=1}^{t} \epsilon_j}{\beta \cdot (\beta + \frac{c \sum_{j=1}^{t} \epsilon_j}{t-1})}.$$

Note that $\alpha \cdot h_0^2(t) = \frac{\alpha \cdot h_0^2(1)}{\beta^2}$. Thus, to satisfy (11), it is enough to show that

$$\beta \cdot c \cdot \left(\sum_{j=1}^{t} \epsilon_j\right) \geq \alpha \cdot h_0(1) \cdot \left(\beta + \frac{c \sum_{j=1}^{t} \epsilon_j}{t-1}\right).$$

Note that

$$h_0(1) \leq \max_{j \sim i} g_{0,j}(\Delta) = 1 + 1.3\epsilon_1 H_{\Delta-1} \underset{\text{by (1)}}{\leq} 1 + 1.3\frac{\delta^2}{10}. \tag{13}$$

Moreover, $\sum_{j=1}^{t} \epsilon_j \geq \sum_{j : j \sim_S 0} \epsilon_j = \alpha$. Thus, letting $c = 1 + c'\delta$ for some $c' > 0$ that we choose later, it is enough to show that

$$\beta \cdot (c' - 0.13\delta)\delta \geq (1 + 0.13\delta) \cdot \frac{(1 + c'\delta) \sum_{j=1}^{t} \epsilon_j}{t-1}.$$

Using $\frac{\sum_{j=1}^{t} \epsilon_j}{t-1} \leq 2\epsilon_1 \underset{(1)}{\leq} \frac{\delta^2}{5}$, it is enough to show that

$$\beta \cdot (c' - 0.13\delta) \geq (1 + 0.13\delta)(1 + c'\delta)\frac{\delta}{5}. \tag{14}$$

On the other hand,

$$\beta \geq 1 - (1 + c'\delta)\left(\sum_{j=1}^{\Delta(0)} \epsilon_j H_{\Delta(0)-1}(j-1)\right) \underset{(2)}{\geq} 1 - (1 + c'\delta)(1 - \delta) = \delta(1 - c' + c'\delta), \tag{15}$$

Thus, to satisfy (14), it is enough to show that $(1-c'+c'\delta)(c'-0.13\delta) \geq (1+1.13\delta)(1+c'\delta)\frac{1}{5}$. Letting $c' = \frac{1}{2}$, this inequality holds for every $0 < \delta < 1$. This establishes *Equation* (8). So we verified conditions Item i and Item ii are satisfied.

To show that all conditions of Theorem 16 are satisfied, it remains to show that $\max_{i\in[d]} f_S(i) \leq \frac{(k-1)^2}{3k-1}$. Note that $\sum_{j:j\sim i} \epsilon_{\{i,j\}} \leq \Delta_S \cdot \epsilon_1 \underset{(1)}{\leq} \Delta_S \cdot \frac{\delta^2}{10}$ for all $i \in [d] \setminus S$. Thus, we get $\max_{i\in[d]} f_S(i) \leq \Delta_S \cdot \frac{\delta^2}{10} \max_{i\in[d]\setminus S} \cdot h_i(\Delta_S(i))$. Moreover, using (13) and (15) with $c' = \frac{1}{2}$ (we can write this inequality for every i), we get

$$h_i(\Delta_S(i)) \leq h_i(\Delta(i)) \leq \frac{1+\frac{\delta^2}{10}}{\delta(\frac{1}{2}+\frac{\delta}{2})}, \tag{16}$$

Thus, we can write

$$\max_{i\in[d]} f_S(i) \leq \Delta_S \cdot \frac{\delta^2}{10}\frac{1+\frac{\delta^2}{10}}{\delta(\frac{1}{2}+\frac{\delta}{2})} \leq \frac{\Delta_S}{5} \leq \frac{k-1}{5} \leq \frac{(k-1)^2}{3k-1},$$

as desired. So we proved that $\{f_S\}_{S\subset[d]:|S|<d}$ satisfies the conditions of Theorem 16. Now, we are ready to bound $\lambda_2(P_\tau)$ for any face τ of co-dimension $k \geq 2$ and type S. First, we show that for every $i \in [d] \setminus S$, $\sum_{j:j\sim_S i} \epsilon_{\{i,j\}} \leq 1-\delta$. Note that

$$\sum_{\ell=1}^{\Delta(i)} H_{\Delta(i)-1}(\ell-1) = \sum_{\ell=2}^{\Delta(i)} \frac{\ell}{\ell-1} = 2 + \sum_{\ell=3}^{\Delta(i)} \frac{\ell}{\ell-1} \geq \Delta(i).$$

Thus, we can write

$$\sum_{j:j\sim_S i} \epsilon_{\{i,j\}} \leq \left(\sum_{\ell=1}^{\Delta(i)} \frac{H_{\Delta(i)-1}(\ell-1)}{\Delta(i)}\right)\left(\sum_{j\sim i} \epsilon_{\{i,j\}}\right) \leq \sum_{\ell=1}^{\Delta(i)} H_{\Delta(i)-1}(\ell-1)\cdot\epsilon_{\{i,j_\ell\}} \leq 1-\delta. \tag{17}$$

where we assumed that $i_1, \ldots, j_d$ is an ordering of $[d] \setminus S$ such that $\epsilon_{j_1} \leq \cdots \leq \epsilon_{j_d}$. Using this inequality and (16), we get

$$\begin{aligned}\lambda_2(P_\tau) &\leq \frac{\max_{i\in[d]\setminus S} f_S(i)}{k-1} \leq \frac{\max_{i\in[d]}(\sum_{j\sim_S i}\epsilon_{\{i,j\}})\cdot h_i(\Delta_S(i))}{k-1} \\ &\leq \frac{(1-\delta)\cdot\max_{i\in[d]} h_i(\Delta(i))}{k-1} \leq \frac{(1-\delta)\cdot\frac{2(1+\frac{\delta^2}{10}\delta)}{\delta(\delta+1)}}{k-1},\end{aligned}$$

as desired. ◀

References

1 Dorna Abdolazimi, Kuikui Liu, and Shayan Oveis Gharan. A matrix trickle-down theorem on simplicial complexes and applications to sampling colorings. In *FOCS*, pages 161–172. IEEE, 2021.

2 Vedat Levi Alev, Fernando Granha Jeronimo, and Madhur Tulsiani. Approximating constraint satisfaction problems on high-dimensional expanders. In David Zuckerman, editor, *FOCS*, pages 180–201. IEEE Computer Society, 2019.

3 Nima Anari, Kuikui Liu, and Shayan Oveis Gharan. Spectral independence in high-dimensional expanders and applications to the hardcore model. *SIAM Journal on Computing*, FOCS20:1–37, 2020. doi:10.1137/20M1367696.

4 Nima Anari, Kuikui Liu, Shayan Oveis Gharan, and Cynthia Vinzant. Log-concave polynomials II: high-dimensional walks and an FPRAS for counting bases of a matroid. In Moses Charikar and Edith Cohen, editors, *STOC*, pages 1–12. ACM, 2019.

5 Roger Carter. *Simple groups of Lie type*. John Wiley & Sons, 1989.

6 Zongchen Chen, Kuikui Liu, and Eric Vigoda. Optimal mixing of glauber dynamics: entropy factorization via high-dimensional expansion. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC*, pages 1537–1550. ACM, 2021.

7 Yotam Dikstein and Irit Dinur. Agreement testing theorems on layered set systems. In David Zuckerman, editor, *FOCS*, pages 1495–1524. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00088.

8 I. Dinur and T. Kaufman. High dimensional expanders imply agreement expanders. In *FOCS*, pages 974–985, 2017.

9 Irit Dinur, Shai Evra, Ron Livne, Alexander Lubotzky, and Shahar Mozes. Locally testable codes with constant rate, distance, and locality. In Stefano Leonardi and Anupam Gupta, editors, *STOC*, pages 357–374. ACM, 2022.

10 Irit Dinur, Yuval Filmus, Prahladh Harsha, and Madhur Tulsiani. Explicit sos lower bounds from high-dimensional expanders. In James R. Lee, editor, *12th Innovations in Theoretical Computer Science Conference, ITCS 2021, January 6-8, 2021, Virtual Conference*, volume 185 of *LIPIcs*, pages 38:1–38:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

11 Irit Dinur, Prahladh Harsha, Tali Kaufman, Inbal Livni Navon, and Amnon Ta-Shma. List-decoding with double samplers. *SIAM J. Comput.*, 50(2):301–349, 2021.

12 Shai Evra and Tali Kaufman. Bounded degree cosystolic expanders of every dimension. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 36–48, 2016.

13 Tali Kaufman and Izhar Oppenheim. Construction of new local spectral high dimensional expanders. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *STOC*, pages 773–786. ACM, 2018. doi:10.1145/3188745.3188782.

14 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of ramanujan complexes of type ad. *European Journal of Combinatorics*, 26(6):965–993, 2005.

15 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Explicit constructions of ramanujan complexes of type Ãd. *European Journal of Combinatorics*, 26(6):965–993, 2005. Combinatorics and Representation Theory. doi:10.1016/j.ejc.2004.06.007.

16 Alexander Lubotzky, Beth Samuels, and Uzi Vishne. Ramanujan complexes of type a d. *Israel journal of Mathematics*, 149:267–299, 2005.

17 Ryan O'Donnell and Kevin Pratt. High-Dimensional Expanders from Chevalley Groups. In Shachar Lovett, editor, *37th Computational Complexity Conference (CCC 2022)*, volume 234 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 18:1–18:26, Dagstuhl, Germany, 2022. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

18 Izhar Oppenheim. Local spectral expansion approach to high dimensional expanders part i: Descent of spectral gaps. *Discrete and Computational Geometry*, 59(2):293–330, 2018.

Derandomization with Minimal Memory Footprint

Dean Doron
Ben-Gurion University of the Negev, Beer-Sheva, Israel

Roei Tell
The Institute for Advanced Study at Princeton, NJ, USA
DIMACS Center at Rutgers University, Piscataway, NJ, USA

Abstract

Existing proofs that deduce $\mathbf{BPL} = \mathbf{L}$ from circuit lower bounds convert randomized algorithms into deterministic algorithms with large constant overhead in space. We study space-bounded derandomization with minimal footprint, and ask what is the minimal possible space overhead for derandomization. We show that $\mathbf{BPSPACE}[S] \subseteq \mathbf{DSPACE}[c \cdot S]$ for $c \approx 2$, assuming space-efficient cryptographic PRGs, and, either: (1) lower bounds against bounded-space algorithms with advice, or: (2) lower bounds against certain *uniform* compression algorithms. Under additional assumptions regarding the power of catalytic computation, in a new setting of parameters that was not studied before, we are even able to get $c \approx 1$.

Our results are constructive: Given a candidate hard function (and a candidate cryptographic PRG) we show how to transform the randomized algorithm into an efficient deterministic one. This follows from new PRGs and targeted PRGs for space-bounded algorithms, which we combine with novel space-efficient evaluation methods. A central ingredient in all our constructions is hardness amplification reductions in logspace-uniform $\mathbf{TC}^0$, that were not known before.

2012 ACM Subject Classification Theory of computation → Complexity theory and logic; Theory of computation → Pseudorandomness and derandomization; Theory of computation → Error-correcting codes

Keywords and phrases derandomization, space-bounded computation, catalytic space

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.11

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2023/036/` [14]

Funding The second author is supported in part by the National Science Foundation under grant number CCF-1900460.

Acknowledgements We are grateful to Avi Wigderson for several useful conversations regarding the gap between double space blow-up and single space blow-up. We thank Lijie Chen for suggesting the idea of using catalytic space to save on complexity, early in this work, and for pointing out a gap in a previous version of the proof of Theorem 2. We also thank Ian Mertz for several useful conversations exploring the abilities of catalytic space. We are very grateful to an anonymous reviewer for a careful read of this paper and for many useful comments. Part of this work was done while the second author was visiting the Simons Institute for the Theory of Computing.

1 Introduction

One of the greatest challenges in complexity theory is the derandomization of efficient algorithms, or more broadly, understanding to what extent randomness is necessary or useful for algorithms. In the time-bounded setting, can we simulate any randomized algorithm by a deterministic one with a roughly similar runtime? In the space-bounded setting, can we derandomize with only a small factor blowup in space?

Classical hardness-to-pseudorandomness results tell us that under plausible circuit lower bounds, any randomized algorithm that runs in time T can be be simulated deterministically by an algorithm running in time T^c [37, 26], and any randomized algorithm that uses S

space can be simulated deterministically in space $c \cdot S$ [31], where c is a large constant. In the terminology of complexity classes, $\mathbf{BPP} = \mathbf{P}$ and $\mathbf{BPL} = \mathbf{L}$ follow from circuit lower bounds.[1]

The constant c in the foregoing classical results can indeed be large to the point of impracticality, for reasons that are inherent to the proof techniques. Therefore, a natural question is whether these results can be made *more efficient*, by providing an explicit small bound on the time or space overhead in derandomization. In other words, we ask what is the *precise, fine-grained value of randomness* in various computational settings. Note that this question is likely to be relevant even when the great goal of establishing $\mathbf{BPP} = \mathbf{P}$ and $\mathbf{BPL} = \mathbf{L}$ without relying on hardness assumptions is achieved.

In recent years, starting with the work of Doron, Moshkovitz, Oh, and Zuckerman [13] and continuing with the works of Chen and Tell [11, 10, 12], a study of *fine-grained derandomization* led to a series of results:[2]

- $\mathbf{BPTIME}[T] \subseteq \mathbf{DTIME}[T^{2+\alpha}]$ assuming there exists a language in $\mathbf{DTIME}[2^{(1+\alpha)n}]$ that is hard for certain randomized, non-deterministic circuits of size $2^{(1-\alpha)n}$ [13]. Chen and Tell [11] showed that one can get rid of the circuits' randomness by assuming that the language is batch-computable.
- $\mathbf{BPTIME}[T] \subseteq \mathbf{DTIME}[n^{1+\alpha} \cdot T]$, where n denotes the length of the input, assuming that one-way functions exist, and there exists a language in $\mathbf{DTIME}[2^{k \cdot n}]$ that is hard for $\mathbf{DTIME}[2^{(k-\alpha)\cdot n}]/2^{(1-\alpha)\cdot n}$ [11]. In a followup work, Chen and Tell [12] showed that one can forgo the cryptographic assumption and replace it with [13]-style ones.
- $\mathbf{BPTIME}[T] \subseteq \mathbf{heurDTIME}[n^{\alpha} \cdot T]$, meaning that the derandomization fails with negligible probability with respect to all efficiently-samplable distributions [11]. This result follows from uniform cryptographic assumptions and certain uniform hardness assumptions for multi-bit output functions.
- Derandomization of *interactive proof systems* with constantly many rounds, that either has a (bounded) polynomial time overhead that depends on the number of rounds, or has only n^{α} time overhead and yields a deterministic (**NP**-style) argument system [12].

These results are often complemented with nearly matching conditional lower bounds (i.e., lower bounds assuming certain complexity-theoretic assumptions). In addition to derandomization in nearly no cost, those results gave rise to new notions, tools, and techniques in derandomization.

The space-bounded setting

In this work, we study efficient *space*-bounded derandomization under hardness assumptions, asking what is the minimal possible *space* overhead for derandomization. That is, can we transform randomized algorithms into deterministic ones that use roughly the same amount of memory?

We note that unlike the time-bounded setting, wherein unconditional derandomization results would lead to lower bounds that currently seem out of reach (see, e.g., [25, 28, 30, 47, 34, 44, 7, 8]), in the space-bounded setting we do have unconditional partial derandomization results. Savitch's theorem [40] can be extended to show that $\mathbf{BPSPACE}[S] \subseteq \mathbf{DSPACE}[O(S^2)]$ (see also [5]). Nisan [35, 36] devised a time-efficient derandomization with

[1] We also have equality between the promise classes. In fact, all our results in this paper will hold for the corresponding promise classes as well, but for readability we will omit the promise problems notation.

[2] In what follows, $\alpha > 0$ is an arbitrarily small constant, but different appearances of α may (or should) not be the same. We refer the reader to the relevant papers for the precise statements.

a quadratic overhead in space, namely, $\mathbf{BPL} \subseteq \mathbf{DTISP}[\mathrm{poly}(n), O(\log^2 n)]$. Focusing solely on space, Saks and Zhou [39] cleverly built on Nisan's work to deterministically simulate space-S randomized algorithms in $\mathbf{DSPACE}[O(S^{2/3})]$. The state-of-the-art is a recent improvement by Hoza [23], giving a deterministic simulation in space $O(S^{2/3}/\sqrt{\log S})$.

Still, even *when* $\mathbf{BPL} = \mathbf{L}$ is proven, it is very likely that the minimal-footprint derandomization question would remain: What is the minimal c for which

$$\mathbf{BPSPACE}[S] \subseteq \mathbf{DSPACE}[c \cdot S]?$$

We will give assumptions under which c approaches 2, and further assumptions under which c approaches 1! Moreover, the results in this paper are *constructive*. Namely, given a candidate hard function (and a candidate cryptographic PRG), we show how to transform the randomized algorithm into an efficient deterministic one.

We proceed to give an overview of our results. Throughout the paper, when we refer to a nice space function, we mean a function $S(n) \geq c_0 \cdot \log(n)$ (where $c_0 \geq 1$ is a universal constant) such that there exists a Turing machine that gets input $(x, 1^t)$ where $t \leq \lceil \log(S(|x|)) \rceil$, runs in space $O(\log(|x|) + \log(S(|x|)))$, and accepts if and only if $t = \lceil \log(S(|x|)) \rceil$.

1.1 Setting the stage: A tighter hypothesis and improved local list decoding

We first revisit the Klivans–van-Malkebeek result [31] that establishes $\mathbf{BPL} = \mathbf{L}$ from standard, nonuniform hardness assumptions. The [31] result, which goes along the line of [37], states that given a language in $\mathbf{DSPACE}[O(n)]$ that is hard for circuits of size $2^{\varepsilon n}$, then $\mathbf{BPL} = \mathbf{L}$.[3] Can we do better? In particular, can we work with a more restricted class of circuits? We show:

▶ **Theorem 1** (see also [14, Theorem 5.2])**.** *Assume there exists a language $L \in \mathbf{DSPACE}[O(n)]$ that is hard for $\mathbf{TC}^0$ circuits of size $2^{\varepsilon n}$, for some $\varepsilon \in (0, 1)$, with oracle access to read-once branching programs of length and width $2^{\varepsilon n}$. Then, for any nice space function S,*

$$\mathbf{BPSPACE}[S] \subseteq \mathbf{DSPACE}[O(S)].$$

While Theorem 1 is not needed for our minimal-footprint results, the main ingredient that goes into the proof of Theorem 1 is a basic component in all of our results: We give a new *hardness amplification* result in $\mathbf{TC}^0$, or equivalently, a new locally list decodable code with $\mathbf{TC}^0$ decoding. We elaborate on it in Section 2.1.

1.2 Black-box derandomization with minimal footprint

Our first derandomization result follows from *worst-case nonuniform hardness assumptions* and *cryptographic assumptions*. We begin with our hardness assumption, which asserts that there is a language computable in linear space that is hard for algorithms that use smaller linear space as well as non-uniform advice.

[3] In [31] it is also stated that one can obtain $\mathbf{BPL} = \mathbf{L}$ from a size-$2^{\varepsilon n}$ lower bound on branching programs. The proof of this statement is not spelled out there in full detail, and as far as we understand, the branching programs referred to in the statement are non oblivious and non read once – a model that lies between $\mathbf{NC}^1$ and $\mathbf{AC}^1$. Theorem 1 gives a stronger statement.

▶ **Assumption 1** (nonuniform hardness assumption)**.** *For a sufficiently large constant C there exists a language L computable in deterministic space $(C+1)\cdot n$ that is hard, on all but finitely many input lengths, for algorithms that run in deterministic space $C\cdot n$ with $2^{n/2}$ bits of advice.*

We note that the gap of $C+1$ vs. C can be replaced by any constant gap (i.e., $C+k$ vs. C for any small constant k), at the cost of allowing a relatively minor additional overhead in the derandomization; for the precise statement, see [14, Theorem 5.5]. We note that a small gap between the space complexity of L and the space for which it is hard for, is inherent for "super efficient" derandomization results merely due to space hierarchy theorems.

We continue with our cryptographic PRG.

▶ **Assumption 2.** *There exists a polynomial-stretch PRG fooling circuits of arbitrary polynomial size, computable in logarithmic space.*[4]

One appealing candidate for a cryptographic PRG satisfying Assumption 2 is Goldreich's expander-based PRG [16], instantiated with expanders whose neighbor function is logspace-computable; we elaborate on this below. However, in the assumption we can use any cryptographic PRG with polynomial stretch, as long as its space complexity is as described.

Equipped with those two assumptions, we can state our efficient derandomization from worst case hardness assumptions.

▶ **Theorem 2** (see also [14, Theorem 5.5])**.** *Suppose that Assumption 1 and Assumption 2 hold. Then, for any nice space function S, we have that*

$$\mathbf{BPSPACE}[S] \subseteq \mathbf{DSPACE}\left[\left(2+\frac{c}{C}\right)S\right],$$

where $c>1$ is some fixed universal constant.

As the section's name suggests, the above result uses a space-efficient pseudorandom generator (along the lines of [11]), which we combine with a new method to space-efficiently evaluate a space-bounded machine over the PRG's image, utilizing the machine's own configuration. See Section 2.2 for a discussion about the techniques.

On the hardness assumptions

The combination of two assumptions – one asserting hardness for non-uniform machines, and an additional one that is either cryptographic or relies on hardness for non-deterministic non-uniform machines (as in [13]) – is in line with previous works in the area (see [11, 12]). However, previous works focused on time bounded algorithms, whereas the space bounded model turns out to be more subtle, posing several additional challenges. Thus, the particular hardness assumptions that we use above are more specialized. Let us elaborate.

First, note that the hypothesized lower bound in Assumption 1 is against space-bounded Turing machines (with advice). One could have hoped for a hardness assumption that is even closer to Theorem 1, namely, for *read-once branching programs* (or for $\mathbf{TC}^0$ circuits with oracle access to such programs). In the technical section we show that Assumption 1 can indeed be relaxed to a seemingly weaker, branching programs based assumption, which is a bit more involved to state (see [14, Section 5.3.2] for details).

---

[4] That is, for any constants η and k we have a PRG $C^{\mathsf{cry}}\colon \{0,1\}^{n^\eta} \to \{0,1\}^{n^k}$ computable in space $O(\log(n^\eta)+\log\log(n^k))$ (see [14, Section 3.1]).

Secondly, our cryptographic PRG is not just an arbitrary one, but has to be logspace-computable. However, as mentioned above, we propose Goldreich's PRG [16] as a natural and well-studied candidate, that works as follows. Let $\Gamma\colon [n^C] \times [d] \to [n^\eta]$ be the neighbor function of a suitable *lossless expander*, and let $P\colon \{0,1\}^d \to \{0,1\}$ be a predicate. Then, given $s \in \{0,1\}^{n^\eta}$ and $i \in [n^C]$, we define

$$G^{\mathsf{exp}}(s)_i = P\left(s{\restriction}_{\Gamma(i)}\right),$$

where $s{\restriction}_{\Gamma(i)}$ is the restriction of s to the set of right-neighbors of i the lossless expander. For Γ, we use an explicit, space-efficient expander [22, 29]. We further discuss Goldreich's PRG and its security, including possible choices for P, in [14, Section 5.3.1].

1.3 Non black-box derandomization with minimal footprint

Next, we turn to minimal-footprint derandomization under *uniform* hardness assumptions. Roughly speaking, we assume the existence of a function computable in space $(C+1)\cdot n$ that cannot be probabilistically "compressed" (even in slightly larger space) into a small Turing machine that uses only $C \cdot n$ space and computes the function. Formally:

▶ **Definition 1.** *We say that $P \in \{0,1\}^\star$ is an S-*space compressed version of *$f \in \{0,1\}^\star$ if P is a description, of length $\sqrt{|f|}$, of a Turing machine M that satisfies the following: On input $x \in [|f|]$, the machine M runs in space $S(\log(|f|))$ and outputs f_x.*

▶ **Assumption 3.** *For a sufficiently large constant C, there exists a function $f\colon \{0,1\}^\star \to \{0,1\}^\star$ mapping n bits to n^2 bits, that is computable in space $(C+1)\cdot\log n$, and satisfies the following. For every probabilistic algorithm R running in space $C\cdot\log n + O(\log n)$[5] there are at most finitely many $x \in \{0,1\}^\star$ for which*

$$\Pr\left[R(x) \text{ prints a } (C\cdot\log n)\text{-space compressed version of } f(x)\right] \geq \frac{2}{3}.$$

Again, similarly to our comments after Assumption 1, the precise difference of $C+1$ vs. C is not crucial (i.e., we can use $C+k$ vs. C for a fixed small universal k), and moreover the precise "amount of compression" can also be relaxed (e.g., compressing to $|f|^{0.01}$ instead of to $\sqrt{|f|}$); see [14, Section 6] for the precise details.

▶ **Theorem 3** (see also [14, Theorem 6.5])**.** *Suppose that Assumption 3 and Assumption 2 hold. Then, for any nice space function S, we have that*

$$\mathbf{BPSPACE}[S] \in \mathbf{DSPACE}\left[\left(2+\frac{c}{C}\right)S\right],$$

where $c > 1$ is some fixed universal constant.

The derandomization algorithm in Theorem 3 does not rely on a pseudorandom generator, but instead works in a "non black-box" way that depends on the *input*. This follows an approach in a recent line of works initiated in [10] (with origins dating back to [18, 17]). As in previous works, the underlying hardness-to-randomness tradeoff is *instance-wise*, in the sense that for every space-S machine M and any *fixed input* x, if a certain machine $R_M(x)$ fails to print a compressed version of $f(x)$, then the deterministic simulation of M succeeds at the particular input x.

[5] The constant hidden in the $O()$ does not depend on C.

On the hardness assumption

The hardness assumption in Theorem 3 is different than the one in Theorem 2, but the conclusion is identical. This lends additional support for the possibility of derandomization with small footprint. Moreover, assuming hardness only for uniform algorithms (as in Theorem 3) is preferable, and the notion of hardness on all but finitely many inputs is necessary for derandomization and was used in several recent works (see, e.g., [10, 32, 33]).

Nevertheless, the particular notion of hardness of *compressing* $f(x)$ is non standard, and we elaborate on it. Recall that, by Kolmogorov-complexity-type arguments, almost all strings do not have *any* concise representation, let alone one that represents a space-bounded machine. (In particular, since such a representation does not exist, then certainly it is impossible to efficiently find it as in Assumption 3.) The crux of the assumption is that such representations are infeasible to find even for the outputs of the efficiently-computable function $f(x)$. We also note that an assumption reminiscent of "hardness of compressing $f(x)$ on all but finitely many inputs x" was recently used to *characterize time-bounded derandomization* (i.e., it is equivalent to the statement $\mathbf{prBPP} = \mathbf{prP}$); see [32] for precise details.

Lastly, since the underlying hardness-vs.-randomness tradeoff is instance-wise, the statement of Theorem 3 is robust, in the following sense: If the hardness holds not on all n-bit inputs, but rather only on $1 - \mu(n)$ fraction of the n-bit inputs over some distribution $\mathbf{x}_n$, then the derandomization succeeds with precisely the same probability and over the same distribution. Further details appear in [14, Section 6].

1.4 Catalytic computation towards an even smaller footprint

In the model of *catalytic computation*, introduced by Buhrman et al. [6], we enrich the space-bounded model with an *auxiliary memory*, that initially already stores some data. While we are allowed to use the auxiliary memory for *our* computation, in addition to the standard work tape, the auxiliary memory needs to be restored to its original content after use. Can such a seemingly restrictive usage be useful for computation? The work of [6] and followup works showed that it is indeed the case. Here, we give a possible *application* of catalytic space to derandomization, a connection that as far as we know, was not suggested before.

Suppose that our hard language from Assumption 1 can be computed catalytically, that is, most of the space used to compute it can be eventually restored. More concretely, consider the following assumption:

▶ **Assumption 4.** *The language from Assumption 1 is computable in space n using additional $C \cdot n$ auxiliary catalytic space.*

Then, we can show:

▶ **Theorem 4.** *Suppose that Assumption 4 and Assumption 2 hold. Then, for any nice space function S, we have that*

$$\mathbf{BPSPACE}[S] \subseteq \mathbf{DSPACE}\left[\left(1 + \frac{c}{C}\right) S\right]$$

for some fixed universal constant c.

The same conclusion holds when adapting Assumption 3 in similar fashion.

Theorem 4 brings us tantalizingly close to derandomization without added memory footprint. Interestingly, the regime of parameters in Assumption 4, where the work space is only a small constant fraction of the catalytic space, has not been studied in the catalytic

computation literature. (So far, the focus has been on the particular case where the catalytic space is exponential in the working space.) We thus view Assumption 3 also as a motivation to study catalytic computation in other regimes of parameters, which are useful for the study of derandomization.

2 Technical Overview

In Section 2.1 we describe the proof of Theorem 1, the main component of which (a new error-correcting code – see Theorem 2) will be used in the subsequent proofs. Then, in Section 2.2 we describe the proofs of Theorem 2 and Theorem 4. In particular, we show how to eliminate derandomization overheads, using a new algorithmic idea for derandomization, a particular type of cryptographic PRG, and an assumption about catalytic space. The proof of Theorem 3 is described in Section 2.3, and requires the strengthening of all the components described in Sections 2.1 and 2.2.

2.1 Warm-up: Hardness amplification for TC^0 circuits in linear space

The proof of Theorem 1 relies on the standard hardness-vs.-randomness approach, following [37, 26, 43]: Given an input $x \in \{0,1\}^n$ the derandomization algorithm first computes the truth-table $f \in \{0,1\}^{\mathrm{poly}(n)}$ of the hard function (on input length $O(\log n)$); then it transforms f into a truth-table $\bar{f} \in \{0,1\}^{\mathrm{poly}(|f|)}$ of a function that is hard on average, using a locally list-decodable error-correcting code; and finally it uses the Nisan–Wigderson generator to transform $\bar{f}$ into pseudorandom strings on which the probabilistic machine is evaluated with input x (see, e.g., [15, Chapters 7, 8], [4, Chapters 19, 20]).

The bottleneck in this approach is the worst-case to average-case reduction underlying the transformation of f to $\bar{f}$. To prove that the derandomization works for logspace machines, it suffices for $\bar{f}$ to be hard on $1/2 - 2^{-\varepsilon \cdot m}$ fraction of its inputs for ROBPs of width $2^{\varepsilon \cdot m}$, for some $\varepsilon > 0$ and where $m = \log(|\bar{f}|)$.[6] In order to deduce this conclusion using the standard argument of [43], we need to assume that f itself is hard (in the worst-case) for $\mathcal{C}$-procedures with oracle access to ROBPs of linear width, where $\mathcal{C}$ is the complexity of the local list-decoding algorithm of the code.

Loosely speaking, to decode from distance $1/2 - \delta$ (and deduce hardness on $1/2 - \delta$ fraction of the inputs), the procedure $\mathcal{C}$ needs to be able to compute the majority function (on $\Theta(1/\delta)$ bits, which in our setting would be $\Theta(2^{\varepsilon \cdot m})$ bits; see [19]).[7] Unfortunately, even when allowing $\mathcal{C} = \mathbf{TC}^0$, the best known decoder, from [19], only handles $\delta = 2^{-\sqrt{m}}$, which is too large for us. The codes that *are* typically used for hardness amplification with $\delta = 2^{-\varepsilon \cdot m}$ (i.e., the ones from [26, 43]) are not known to be locally list-decodable in complexity as low as $\mathbf{TC}^0$.[8]

The key observation allowing us to bridge this gap is that for our application of hardness amplification, we do not have to insist on the $\mathbf{TC}^0$ circuit being of size $\mathrm{poly}(\ell)$, where $\ell = \log(|f|)$, as in the standard setting of local coding. In fact, in our setting we can allow a circuit of size $2^{\varepsilon \cdot \ell}$. Given this relaxation, we construct the following suitable code.

[6] This is actually an over-simplification, and what we actually need is for $\bar{f}$ to be hard on $1/2 - 2^{-\varepsilon \cdot m}$ fraction of the its inputs for $\mathbf{AC}^0$ circuits that have oracle access to an ROBPs of width $2^{\varepsilon \cdot m}$ (this follows from the standard reconstruction argument of [37]). In this high-level overview we ignore the $\mathbf{AC}^0$ overhead, for simplicity of presentation.

[7] In fact, a similar statement holds for any "black-box" worst-case to average-case hardness amplification [46, 41, 20].

[8] The bottleneck in both cases is local list-decoding of the Reed-Muller code; see [10] for a recent construction of a decoder in logspace-uniform $\mathbf{NC}$.

▶ **Theorem 2** (see also [14, Theorem 4.2]). *There exists a universal constant $c > 1$ such that for any constant $\gamma \in (0,1)$ the following holds. For every $k \in \mathbb{N}$ and $\varepsilon > 0$, there exists a logspace-computable code $\mathcal{C}\colon \{0,1\}^k \to \{0,1\}^n$, for $n = (\frac{k}{\varepsilon})^{c/\gamma}$, that is locally list decodable from $1/2 + \varepsilon$ fraction of agreement by constant-depth threshold circuits of size $k^\gamma \cdot (1/\varepsilon)^c$.*

At a high level, the proof of Theorem 2 (wherein one should think of $k = 2^m$) combines three known code constructions:

1. A small modification of the code of [19], which uniquely decodes from agreement $1 - \frac{1}{25}$ using $\mathbf{TC}^0$ circuits;
2. the derandomized direct-product code of [26], which $(1 - \frac{1}{25})$-approximately list-decodes from agreement $\eta = 2^{-O(\varepsilon \cdot m)}$ using a $\mathbf{TC}^0$ circuit of size $2^{O(\varepsilon \cdot m)}$; and,
3. the Hadamard code, which we concatenate with the direct product code and is list-decodable from agreement $\frac{1}{2} + 2^{-\varepsilon \cdot m}$ by $\mathbf{TC}^0$ circuits of size $\mathrm{poly}(m)$.

As a corollary, we obtain a $\mathbf{TC}^0$-computable worst-case to average-case reduction for computing functions in $\mathbf{DSPACE}[O(n)]$; see [14, Corollary 4.1]. This reduction handles the "high-end" parameter regime, which previous reductions for functions in $\mathbf{DSPACE}[O(n)]$ did not handle (see [42, 21, 19]), and is incomparable to reductions computable by probabilistic (uniform) algorithms [45, 9].

The decoder's complexity

For our results we crucially rely on the fact that the decoder can be implemented in $\mathbf{TC}^0$ (e.g., when deducing black-box derandomization from hardness for $\mathbf{TC}^0$ circuits with oracle access to branching programs, or when deducing non-black-box derandomization). We suspect that it is possible to construct a code with weaker guarantees – namely, a logspace decoder, rather than a $\mathbf{TC}^0$ decoder – using simpler techniques (i.e., replacing the "outer" code of [19] by more classical tools).

2.2 Derandomization with minimal footprint using PRGs

Let $S = C \cdot \log n$ denote the space complexity of the machine M we wish to derandomize (the result for arbitrary S will follow from padding). At a high-level, our construction follows an approach first introduced in [11], which composes two "low-cost" PRGs:

- An inner PRG that stretches $O(\log n)$ bits to n^η bits for some tiny constant $\eta > 0$, and,
- an outer PRG that stretches n^η bits to n^C bits.

Specifically, as in [10], we take the inner PRG, denoted by NW, to be an appropriately parameterized Nisan-Wigderson PRG [37] with a hard truth-table $f \in \{0,1\}^{n^2}$, and the outer PRG, denoted by G^{cry}, to be one that relies on a cryptographic assumption.

Unfortunately, materializing this approach in the current setting turns out to be significantly more subtle than in [11]. To see this, observe that the final computation iterates over seeds $s \in \{0,1\}^{O(\log n)}$, and for each s computes

$$M(x, G^{\mathsf{cry}}(\mathsf{NW}^f(s))),$$

where f is the truth-table of the hypothesized hard function. To compute this using space-efficient composition, we use the following chain of simulations:

1. Simulate $M(x, \cdot)$, and whenever it queries its second input –
2. Simulate G^{cry}, and whenever it queries its input –
3. Simulate $\mathsf{NW}^f(s)$, and whenever it queries f –
4. Compute the corresponding bit of f.

Recall that, using space-efficient (emulative) composition, the complexity of the final construction is additive in the space complexity of each of its components, plus additional overheads that are logarithmic in the output length of each component. (The latter logarithmic overhead is caused by the fact that we are simulating a virtual input head for each component. See [14, Proposition 3.2].) A naive implementation of this approach yields space complexity of $3S + \mathsf{Space}(G^{\mathsf{cry}}) + \mathsf{Space}(\mathsf{NW})$, where $\mathsf{Space}(\cdot)$ denotes the space complexity of the corresponding algorithm, and we ignore factors of the form $c \cdot \log n$ where $c > 1$ is a universal constant that doesn't depend on S.

A more efficient derandomization

Our first observation is that the standard way of derandomizing probabilistic space-S machine is wasteful. There, we think of the probabilistic machine as reading a tape of random bits, sequentially; and when simulating it deterministically, we keep track of a counter $i \in [2^S]$, and whenever the machine wishes to read a random bit, we answer using the i-th bit in the pseudorandom output of the generator, and update $i \leftarrow i + 1$.

It might (mistakenly) seem that using a dedicated counter is necessary, because we must ensure that the machine reads each bit in the random (or pseudorandom) sequence exactly once. However, this intuition turns out to be false: Instead of keeping a dedicated counter $i \in [2^S]$, *we can use the machine's own configuration as a counter*. Specifically, recall that at each step the machine has some configuration $\sigma \in \{0,1\}^S$ describing the contents of its work tapes, its current state, and the locations of its heads.[9] Moreover, since for every input x and fixed sequence r of coins, the execution of $M(x,r)$ halts, any configuration $\sigma \in \{0,1\}^S$ is encountered *at most once* during the execution of M (see [14, Claim 3.3]). Thus, we consider the following machine $\bar{M}$, which simulates M using oracle access to a sequence of random coins but without the overhead of keeping a counter:

> Simulate M, and whenever M tries to flip a random coin, access the sequence of random coins at location σ, where σ is M's current configuration.

Since the functionality of $\bar{M}$ and of M at any input x, with uniform coins, is identical, it suffices to faithfully simulate $\bar{M}$ with pseudorandom coins.

At this point the space complexity of the derandomization is essentially

$$2S + \mathsf{Space}(G^{\mathsf{cry}}) + \mathsf{Space}(\mathsf{NW}).$$

Since NW maps a truth-table f of length n^2 to pseudorandom strings of length n^η, it can be computed in space $c' \cdot \log n$ for a universal $c' > 1$ (see [14, Section 5.1]). Thus, our last step is to bound the space complexity of G^{cry}.

We do so by relying on a specific PRG whose space complexity is logarithmic in its input length n^η and sub-logarithmic in its output length n^C. A natural candidate for such a PRG arises from the "cryptography in $\mathbf{NC}^0$" literature (see, e.g., [2, 27, 1, 38]), and in particular we can use Goldreich's PRG [16]. The latter PRG relies on a bipartite lossless expander $\Gamma\colon [n^C] \times [d] \to [n^\eta]$ with a small left-degree d, and on a predicate $P\colon \{0,1\}^d \to \{0,1\}$. For $s \in \{0,1\}^{n^\eta}$ and $i \in [n^C]$, the i-th output of $G^{\mathsf{cry}}(s)$ is

$$P\left(s_{\Gamma(i,1)}, ..., s_{\Gamma(i,d)}\right),$$

where $\Gamma(i,1), \ldots, \Gamma(i,d)$ are the d neighbors of i in the expander.

[9] Indeed, we count the location of the heads and the state in the configuration of the machine, and in fact assume that they are written on dedicated worktapes; see [14, Section 3] for the precise details.

In the cryptography literature, the graph is often taken to be a random one, but in our setting we need a lossless expander whose neighbor function is computable in space $c'' \cdot \log n$, and also a predicate known to withstand existing attacks that is computable in space $c'' \cdot \log n$, where in both cases $c'' > 1$ is a universal constant. We use the recent expander construction of Kalev and Ta-Shma [29], whose degree is $d = \text{polylog}(n)$ and whose neighbor function is computable in space $O(\text{loglog}n)$, and a predicate introduced by Applebaum and Raykov [3] that is computable in space $O(\log d) = O(\text{loglog}n)$. See [14, Section 5.3.1] for further details.

This brings the complexity of the deterministic simulation of M on each particular seed to be $2S + c \cdot \log n$ for some universal constant c, and after enumerating over all seeds and taking the majority output, the space complexity increases only by an additive factor of, say, at most $c \cdot \log n$.

The reconstruction argument

We prove that the derandomization works using a reconstruction argument. Specifically, in the derandomization, we instantiate the NW generator with the code from Theorem 2, and rely on the reconstruction argument of NW and on the local list-decoding algorithm of the code to transform any ROBP distinguisher for the PRG (which arises from the computation of the space-S machine M on a fixed input x) into an efficient procedure that computes f.

The details of the reconstruction procedure appear in [14, Theorem 5.1], so let us only highlight the important points in the argument. First, our distinguisher is actually derived from $\bar{M}$, the machine that reads bits according to its configuration. Secondly, by a standard analysis of PRG composition, the distinguisher for NW^f is not just the ROBP derived from $\bar{M}$ and x, but actually the composed procedure

$$D(r) = \bar{M}^{G^{\mathsf{crry}}(r)}(x).$$

This increases the complexity of the distinguisher from a simple ROBP to a bounded-space machine. Lastly, while the machine implementing D uses space at least $S = C \cdot \log n$, the amount of non-uniform advice that it uses is much smaller than $2^S = n^C$. Specifically, it uses only $|x| = n = 2^{\log(|f|)/2}$ bits of advice.

To sum up, if the derandomization fails on some input x, then there exists a $\mathbf{TC}^0$ circuit C of size n^ε, and a function D that is computable in space $\approx C \cdot \log n$ with n bits of advice,[10] such that $C^D(i) = f_i$ for all $i \in [n^2]$. Finally, using the fact that C is a $\mathbf{TC}^0$ circuit, we show that C^D itself can be computed by a TM with advice, whose space complexity is only slightly larger than the space complexity of D. This contradicts the hardness of f.

Obtaining a sub-double space overhead

The derandomization above takes $2S + c \cdot \log n$ space, where the increase from S to $2S$ is caused by the space complexity of computing f. Indeed, it seems unavoidable that we will need space larger than S to compute f, because we are assuming that it is hard for algorithms running in space S.

The key observation to reducing this overhead is that if f is computable in *catalytic* space S, we can roughly use the existing used worktape cells – which, at any point in time, contain the current configuration of the derandomized machine – in order to compute each query of

[10] The precise complexity of D is $(C + 1 + \varepsilon) \cdot \log n$, but in the high-level overview we ignore these minor overheads, for simplicity of presentation.

NW to f. Specifically, whenever the derandomization machine queries f at location $i \in [n^2]$, we compute f_i using (mostly) the existing space in a catalytic way. After the computation of the bit f_i ends, the original content of the worktapes is restored. The key point is that since the configuration of the machine does not change during the computation of $i \mapsto f_i$, nor is this computation dependent on the configuration in any way, the correctness of the procedure is maintained.

Thus, under this strengthened hypothesis that f is computable in small catalytic space, the final space complexity of the derandomization algorithm is just $S + c \cdot \log n$.

An alternative to Assumption 1

Recall that $D(r)$ above can be computed by a bounded-space machine that uses $|x|$ bits of advice. While indeed D is not a read-*once* branching program, the computation of $D(r)$ is *oblivious*. Namely, computing C^{cry} can be done by a bounded-width branching program that at each layer queries several locations of r, however these locations are determined only by the underlying expander Γ. Thus, we can model C^D as a $\mathbf{TC}^0$ circuit with (non-adaptive) oracle access to branching programs of the aforementioned type. Moreover, we will later see that the $\mathbf{TC}^0$ circuit can be space-efficiently generated using a short advice. The formal assumption is given in [14, Assumption 5.10], and can replace Assumption 1 for both the double and sub-double overhead results.

2.3 Non black-box derandomization with minimal memory footprint

Set $S = C \cdot \log n$ and recall that we wish to obtain the same conclusions for $\mathbf{BPSPACE}[S]$ as in Section 2.2, but from different assumptions. Specifically, we assume the existence of a function $f\colon \{0,1\}^n \to \{0,1\}^{n^2}$ computable in space $S' = S + O(\log n)$ such that for every probabilistic algorithm R running in space $S_R = S' + O(\log n)$, and every $x \in \{0,1\}^n$, the algorithm $R(x)$ fails to print a *compressed* version of $f(x)$ (except with small probability). In this context, a compressed version means a Turing machine of description size $O(n) = O(\sqrt{|f(x)|})$ that runs in space roughly $S + \log n < S'$.

Following ideas from [10, 32], we will construct a *targeted* PRG, which is an algorithm that maps any input x to a set of pseudorandom strings that will fool the machine M with this particular input x. As in those previous works, our targeted PRG is based on the Nisan–Wigderson generator, and we analyze it using an *instance-wise* hardness vs. randomness tradeoff. Specifically, we show that if the derandomization fails on an input x, then a probabilistic space-S_R machine R succeeds in mapping the same fixed x to a compressed version of $f(x)$. This yields Theorem 3, and also the more general version mentioned after the theorem's statement: For every distribution $\mathbf{x}$ over the inputs, if the probability over $x \sim \mathbf{x}$ that R fails to print compressed version of $f(x)$ is $1 - \mu$, then the derandomization succeeds on $1 - \mu$ of the inputs $x \sim \mathbf{x}$.

The derandomization itself is similar to the one from Section 2.2, with a minor difference that is nevertheless crucial. Instead of instantiating NW with f that is the truth-table of a hypothesized hard function, we instantiate NW with $f = f(x)$ obtained from the input x. That is, we compute the majority, over seeds $s \in \{0,1\}^{O(\log n)}$, of

$$\bar{M}^{G^{\mathsf{cry}}(\mathsf{NW}^{f(x)}(s))}(x) \ .$$

Note that the complexity of the derandomization algorithm is essentially identical to that of the algorithm from Section 2.2. Thus, the only question is – why does it work?

Analysis

The main argument underlying Theorem 3 is proving that there exists a space-S_R algorithm $R = R_M$ satisfying the following: For any fixed $x \in \{0,1\}^n$, when NW is instantiated with the code from Theorem 2, if $\mathsf{NW}^{f(x)}$ does not fool M, then $R(x)$ prints a compressed version of $f(x)$.

The intuition for why this might be possible dates back to [24], who showed that the reconstruction algorithm of NW, which maps a distinguisher to a small circuit for the hard truth-table, can be made *uniform* – as long as it is allowed to make queries to the hard truth-table. Recall that in our setting, the algorithm R explicitly gets the input x, and we are allowing R to run in space that is slightly larger than the space complexity of computing $f(x)$. Therefore, R can simulate the reconstruction algorithm, and whenever the latter queries an index i of $f(x)$, the algorithm R simply computes $f(x)$ and returns the relevant bit.

The main technical challenge that we are faced with is making the algorithm R not only uniform, but also a *small-space algorithm*, and doing so when the underlying *code for hardness amplification* is the one from Theorem 2. The resulting statement appears in [14, Theorem 6.1], and its proof is the most technically subtle argument in this paper. In the rest of the section we describe the proof, at a high-level.

Low-space uniform reconstruction and decoding

We first strengthen the analysis of the code $\mathcal{C}$ from Theorem 2, to show that not only is it locally list-decodable, but that it also has a *probabilistic space-$O(\log n)$ uniform decoder, which does not need non-uniform advice* (but rather uses queries to the corrupt codeword). The algorithm R will answer this decoder's queries to the corrupt codeword $\mathcal{C}(f(x))$ by computing $f(x)$ and then $\mathcal{C}$, which it can do in its allotted space. We compose this uniform decoder for $\mathcal{C}$ with a space-$O(\log n)$ reconstruction algorithm for NW.

We stress that the two algorithms underlying R – the decoder, and the NW reconstruction – run in space $O(\log n)$, but print a procedure of description size $n^{\Omega(1)}$. Thus, not only do the two algorithms need to print a description of a procedure without remembering most of the functionality of the machine that they printed so far – but also the algorithms cannot even *evaluate* the procedures that they print.

The key observation is that in both cases, the decoding/reconstruction prints a procedure *almost all of which is a large, static, truth-table*. To see this, let us focus for simplicity on the NW reconstruction algorithm.[11] Recall that this algorithm implements very simple functionality, which can be described by a logspace-uniform constant-depth circuit of size polylog(n), and that is hard-wired with "static" information of size n^{ε} that is mostly obtained from queries to $\mathcal{C}(f(x))$.[12] With some low-level care, we can design an algorithm that queries $\mathcal{C}(f(x))$ and prints a machine that implements that functionality, and has states encoding the foregoing static information. Thus, the algorithm which prints the machine does not need to remember static information that is already printed.

A related complication arises because both the decoding algorithm of $\mathcal{C}$ and the reconstruction algorithm of NW actually succeed only with small probability. The standard approach (e.g., in [24, 10, 32]) is for R to use queries to $\mathcal{C}(f(x))$ to estimate the agreement of the

[11] The computational bottleneck in the decoder for $\mathcal{C}$ is the decoder for the derandomized direct product code of [26], which acts in a similar way to the reconstruction of NW. Thus, we use similar ideas to handle both the reconstruction of NW and the decoder of $\mathcal{C}$.

[12] The information consists of an index i (used for a hybrid argument), of a combinatorial design, of values for the seed outside the i-th set in the design, and of n^{η} partial truth-tables.

procedure that it outputs with $\mathcal{C}(f(x))$, repeating the experiment until it gets a procedure with sufficiently large agreement. Since in our case R cannot evaluate the procedure that it prints, it cannot take this approach. Instead, R prints a procedure that performs this "success amplification" functionality by itself. We leave the details to the technical section.

Composing the two algorithms

The description above refers vaguely to "the procedure" that R print, and being more accurate, this procedure is a $\mathbf{TC}^0$ circuit C of size n^ε making queries to a space-S machine D that uses n bits of advice. This is not enough, since our goal is for R to print a single Turing machine of description size $O(\sqrt{|f(x)|}) = O(n)$ running in space $S + \log n < S'$.

Bridging this gap requires more care in composing the two algorithms. Specifically, our algorithm R prints a machine whose states encode the circuit C, and that implements the standard DFS-style emulation of $\mathbf{NC}^1 \supseteq \mathbf{TC}^0$ circuits in logspace, while reading the description of the hard-coded C out of its own states. The space overhead of the emulation itself is $O(\log |C|) = O(\log(n^\varepsilon))$, and the machine also needs to compute the values of the gates along each DFS path. In particular, this means that we need to ensure that each path contains at most one oracle call to D (otherwise the machine's space complexity will be larger than $2S$). For this purpose, in our strengthened analysis of $\mathcal{C}$ we ensure that its decoding procedure only makes *non-adaptive queries*. This allows us to bound the space complexity of the machine that R prints by $S + O(\varepsilon \cdot \log n) \leq S + \log n$, as desired.

References

1. Benny Applebaum. *Cryptography in constant parallel time*. Information Security and Cryptography. Springer, 2014.
2. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM Journal on Computing*, 36(4):845–888, 2006.
3. Benny Applebaum and Pavel Raykov. Fast pseudorandom functions based on expander graphs. In *Theory of Cryptography. Part I*, volume 9985, pages 27–56. Springer, 2016.
4. Sanjeev Arora and Boaz Barak. *Computational complexity: A modern approach*. Cambridge University Press, Cambridge, 2009.
5. Allan Borodin, Stephen Cook, and Nicholas Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3):113–136, 1983.
6. Harry Buhrman, Richard Cleve, Michal Koucký, Bruno Loff, and Florian Speelman. Computing with a full memory: catalytic space. In *Proc. 46th Annual ACM Symposium on Theory of Computing (STOC)*, pages 857–866, 2014.
7. Lijie Chen. Non-deterministic quasi-polynomial time is average-case hard for *ACC* circuits. In *Proc. 60th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2019.
8. Lijie Chen, Xin Lyu, and R. Ryan Williams. Almost-everywhere circuit lower bounds from non-trivial derandomization. In *Proc. 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2020.
9. Lijie Chen, Ron D. Rothblum, and Roei Tell. Unstructured hardness to average-case randomness. In *Proc. 63rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 429–437. IEEE, 2022.
10. Lijie Chen and Roei Tell. Hardness vs. randomness, revised: uniform, non-black-box, and instance-wise. In *Proc. 62nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2021.

11 Lijie Chen and Roei Tell. Simple and fast derandomization from very hard functions: Eliminating randomness at almost no cost. In *Proc. 53st Annual ACM Symposium on Theory of Computing (STOC)*, 2021.

12 Lijie Chen and Roei Tell. When arthur has neither random coins nor time to spare: Superfast derandomization of proof systems. *Electronic Colloquium on Computational Complexity: ECCC*, 2022.

13 Dean Doron, Dana Moshkovitz, Justin Oh, and David Zuckerman. Nearly optimal pseudorandomness from hardness. In *Proc. 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2020.

14 Dean Doron and Roei Tell. Derandomization with minimal memory footprint. *Electronic Colloquium on Computational Complexity: ECCC*, 30:036, 2023.

15 Oded Goldreich. *Computational complexity: A conceptual perspective*. Cambridge University Press, New York, NY, USA, 2008.

16 Oded Goldreich. Candidate one-way functions based on expander graphs. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 76–87. Springer, 2011.

17 Oded Goldreich. In a world of $P = BPP$. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay Randomness and Computation*, pages 191–232. Springer, 2011.

18 Oded Goldreich and Avi Wigderson. Derandomization that is rarely wrong from short advice that is typically good. In *Proc. 6th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 209–223, 2002.

19 Shafi Goldwasser, Dan Gutfreund, Alexander Healy, Tali Kaufman, and Guy N. Rothblum. Verifying and decoding in constant depth. In *Proc. 39th Annual ACM Symposium on Theory of Computing (STOC)*, pages 440–449, 2007.

20 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *Proc. 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 956–966, 2018.

21 Venkatesan Guruswami and Valentine Kabanets. Hardness amplification via space-efficient direct products. *Computational Complexity*, 17(4):475–500, 2008.

22 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh-Vardy codes. *Journal of the ACM*, 56(4):Art. 20, 34, 2009.

23 William M. Hoza. Better pseudodistributions and derandomization for space-bounded computation. In *Proc. 25th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM)*, pages 28:1–28:23, 2021.

24 Russel Impagliazzo and Avi Wigderson. Randomness vs. time: Derandomization under a uniform assumption. *Journal of Computer and System Sciences*, 63(4):672–688, 2001.

25 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

26 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: derandomizing the XOR lemma. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '97, page 220–229. Association for Computing Machinery, 1997.

27 Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 433–442, 2008.

28 Valentine Kabanets and Russell Impagliazzo. Derandomizing polynomial identity tests means proving circuit lower bounds. *Computational Complexity*, 13(1-2):1–46, 2004.

29 Itay Kalev and Amnon Ta-Shma. Unbalanced expanders from multiplicity codes. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*, volume 245 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 12:1–12:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

30 Jeff Kinne, Dieter van Melkebeek, and Ronen Shaltiel. Pseudorandom generators, typically-correct derandomization, and circuit lower bounds. *Computational Complexity*, 21(1):3–61, 2012.

31 Adam Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002.

32 Yanyi Liu and Rafael Pass. Characterizing derandomization through hardness of Levin-Kolmogorov complexity. In *37th Computational Complexity Conference (CCC 2022)*, volume 234 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

33 Yanyi Liu and Rafael Pass. Leakage-resilient hardness vs. randomness. *Electronic Colloquium on Computational Complexity: ECCC*, 30:113, 2022.

34 Cody Murray and R. Ryan Williams. Circuit lower bounds for nondeterministic quasi-polytime: An easy witness lemma for NP and NQP. In *Proc. 50th Annual ACM Symposium on Theory of Computing (STOC)*, 2018.

35 Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

36 Noam Nisan. $RL \subseteq SC$. *Computational Complexity*, 4(1):1–11, 1994.

37 Noam Nisan and Avi Wigderson. Hardness vs. randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

38 Hanlin Ren and Rahul Santhanam. Hardness of KT characterizes parallel cryptography. In *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:58. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

39 Michael E. Saks and Shiyu Zhou. $BP_HSPACE(S) \subseteq DSPACE(S^{2/3})$. *Journal of Computer and System Sciences*, 58(2):376–403, 1999.

40 Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970.

41 Ronen Shaltiel and Emanuele Viola. Hardness amplification proofs require majority. In *Proc. 40th Annual ACM Symposium on Theory of Computing (STOC)*, pages 589–598, 2008.

42 Daniel A. Spielman. Linear-time encodable and decodable error-correcting codes. *toit*, 42(6):1723–1731, 1996. Codes and complexity.

43 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

44 Roei Tell. Proving that $prBPP = P$ is as hard as proving that "almost NP" is not contained in $P/poly$. *Information Processing Letters*, 152:105841, 2019.

45 Luca Trevisan and Salil Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity*, 16(4):331–364, 2007.

46 Emanuele Viola. Hardness vs. randomness within alternating time. In *Proc. 18th Annual IEEE Conference on Computational Complexity (CCC)*, pages 53–69, 2003.

47 R. Ryan Williams. Non-uniform acc circuit lower bounds. In *2011 IEEE 26th Annual Conference on Computational Complexity*, pages 115–125, 2011.

Improved Learning from Kolmogorov Complexity

Halley Goldberg
Simon Fraser University, Burnaby, Canada

Valentine Kabanets
Simon Fraser University, Burnaby, Canada

Abstract

Carmosino, Impagliazzo, Kabanets, and Kolokolova (CCC, 2016) showed that the existence of natural properties in the sense of Razborov and Rudich (JCSS, 1997) implies PAC learning algorithms in the sense of Valiant (Comm. ACM, 1984), for boolean functions in $\mathsf{P/poly}$, under the uniform distribution and with membership queries. It is still an open problem to get from natural properties learning algorithms that do not rely on membership queries but rather use randomly drawn labeled examples.

Natural properties may be understood as an average-case version of MCSP, the problem of deciding the minimum size of a circuit computing a given truth-table. Problems related to MCSP include those concerning time-bounded Kolmogorov complexity. MKTP, for example, asks for the KT-complexity of a given string. KT-complexity is a relaxation of circuit size, as it does away with the requirement that a short description of a string be interpreted as a boolean circuit. In this work, under assumptions of MKTP and the related problem $\mathsf{MK^tP}$ being easy on average, we get learning algorithms for boolean functions in $\mathsf{P/poly}$ that

- work over any distribution D samplable by a family of polynomial-size circuits (given explicitly in the case of MKTP),
- only use randomly drawn labeled examples from D, and
- are agnostic (do not require the target function to belong to the hypothesis class).

Our results build upon the recent work of Hirahara and Nanashima (FOCS, 2021) who showed similar learning consequences but under a stronger assumption that NP is easy on average.

2012 ACM Subject Classification Theory of computation → Computational complexity and cryptography

Keywords and phrases learning, Kolmogorov complexity, meta-complexity, average-case complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.12

Acknowledgements We thank Shuichi Hirahara, Russell Impagliazzo, Zhenjian Lu, and Igor Oliveira for helpful discussions.

1 Introduction

There is a deep connection between computational learning and pseudorandomness. Loosely speaking, the goal of learning is to extract "structure" (a simple hypothesis) from a "random" environment, whereas the goal of pseudorandom constructions is to hide "structure" within a "random-looking" environment. Before mentioning any examples illustrating this antagonism between learning and pseudorandomness, let us recall the definitions of some basic learning models.

In Valiant's Probably Approximately Correct (PAC) learning model [29], a learner tries to learn an unknown concept c (say, a Boolean function) from a known class $\mathcal{C}$ of concepts, with respect to some (arbitrary) distribution D over inputs to c. The learner gets to see independently sampled labeled examples of the form $(x, c(x))$, where x is sampled by D, and needs to output (with high probability) a hypothesis h that has just tiny disagreement with c with respect to the distribution D. The agnostic PAC learning model [21] is a natural generalization of the PAC model where an unknown concept f to be learned is *not necessarily*

from the concept class $\mathcal{C}$. The learner gets to see independently sampled labeled examples of the form $(x, f(x))$, with x sampled from some distribution D, and needs to output (with high probability) a hypothesis h so that the disagreement between h and f with respect to D is very close to the disagreement between f and the concept $c_f \in \mathcal{C}$ that is closest to f with respect to D. Classical (agnostic) PAC learning model is distribution-independent in the sense that a successful PAC learning algorithm for a concept class $\mathcal{C}$ must work with respect to *any* distribution D of examples. One also considers a distribution-specific setting where a learning algorithm must work with respect to a single fixed distribution D, e.g., the uniform distribution or a polytime-samplable distribution.

Impagliazzo and Levin [18] and Blum et al. [5] (see also [24]) show that breaking *cryptographic* Pseudorandom Generators (PRGs) implies *average-case* PAC learning with respect to polytime-samplable distributions; here, rather than learning *every* concept from some concept class $\mathcal{C}$, one gets to learn a *significant fraction* of concepts from $\mathcal{C}$ under a polytime-samplable distribution over $\mathcal{C}$. In contrast, Nisan and Wigderson [25] show that breaking *complexity-theoretic* PRGs (namely, the NW generators) implies worst-case learning (of every concept in a given concept class $\mathcal{C}$) under the uniform distribution, but here the learning algorithm needs to make *membership queries* (MQs) to the concept $c \in \mathcal{C}$ it is trying to learn, i.e., the learner gets to ask the value $c(x)$ for any input x of its choosing.

Where does one get an algorithm to break a given PRG in order to get a learning algorithm? For the case of the NW PRG, Carmosino et al. [10] showed that a natural property (in the sense of Razborov and Rudich [27]) for a (sufficiently expressive) circuit class $\mathcal{C}$ yields a learning algorithm for $\mathcal{C}$ under the uniform distribution, with membership queries; this was generalized to learning with respect to polytime-samplable distributions by Binnendyk et al. [4]. Using a known natural property for the class $\mathsf{AC}^0[p]$ of constant-depth circuits with AND, OR, NOT, and mod-p counting gates (for any prime modulus p) from [27], [10] obtained a quasipolynomial-time learning algorithm for $\mathsf{AC}^0[p]$ over the uniform distribution, using membership queries. Later, [11] generalized this framework to show that one also gets *agnostic* learning algorithms from certain generalizations of natural properties. It remains an important open problem to get from a natural property a learning algorithm that uses only random labeled examples. In particular, it would be very interesting to get an efficient learning algorithm for $\mathsf{AC}^0[p]$ without membership queries, which would rule out *weak* Pseudorandom Function Generator constructions in $\mathsf{AC}^0[p]$; see [6] for a recent survey on pseudorandom functions.

A natural property for general circuits is an efficient average-case heuristic for Minimum Circuit Size Problem (MCSP) over the uniform distribution, with one-sided error. Namely, it should always accept the truth tables of Boolean functions of low circuit complexity (for a given threshold size parameter s), and should reject at least a constant fraction of all possible truth tables. MCSP is an example of a meta-complexity problem asking to estimate the circuit size of a given truth table. There are closely related meta-complexity problems for variants of time-bounded Kolmogorov complexity.

For example, MKTP (defined in [1]) asks if a given binary string x is efficiently *locally* computable (outputting bit x_i on any input i in at most t steps) by a universal Turing machine with oracle access to some short binary string d, where one seeks to minimize the sum $|d| + t$. As MCSP, MKTP asks for a description of a string x that allows one to compute x locally, any bit x_i at a time. However, such a description of x needn't be a Boolean circuit for the truth table x, the time t of an algorithm computing each x_i is explicitly taken as part of the complexity measure of x, and this reconstruction algorithm is given random access to the description string d.

We show that this extra flexibility of MKTP compared to MCSP leads to improved learning algorithms from an assumed "natural property" (or one-sided error average-case heuristic) for MKTP, where we get agnostic PAC learning algorithms over explicitly given efficiently samplable distributions. Recall that $\mathsf{SIZE}[s(n)]$ denotes the set of all n-variate Boolean functions computable by circuits of size at most $s(n)$. We have the following.

▶ **Theorem 1** (Learning from MKTP: Informal version)**.** *Suppose* MKTP *has an efficient one-sided error average-case heuristic over the uniform distribution over inputs. Then for any circuit size bound $s(n) \leq \mathsf{poly}(n)$, the concept class $\mathcal{C} = \mathsf{SIZE}[s(n)]$ is agnostic PAC-learnable in polytime with respect to any explicitly given ensemble of polysize-samplable distributions $D = \{D_n\}$.*

Here an ensemble D of distributions D_n is polysize-samplable if there is a family of polysize circuits $Samp_n$ that are samplers for D_n, i.e., the distribution of outputs of $Samp_n$ on uniformly random inputs is D_n. Explicitness of D means that a learning algorithm, when asked to learn some n-variate Boolean function, is explicitly given a description of the sampling circuit $Samp_n$ for the distribution D_n. Note that this explicitness condition for $D = \{D_n\}$ is trivially satisfied by the uniform distribution or any polytime-samplable distribution ensemble (in the latter case, one just needs a constant-size description of a polytime Turing machine that samples according to D_n, for any given n).

For the learning setting over distributions $D = \{D_n\}$ where D is polysize-samplable, but the sampling circuits $Samp_n$ are *not* explicitly given to the learning algorithm (and only their sizes are given), we can get efficient agnostic PAC learning from a "natural property" for a different Kolmogorov-complexity measure, K^t. Recall that, for any time parameter $t \in \mathbb{N}$, $\mathsf{K}^t(x)$ is defined as the length of a shortest string $d \in \{0,1\}^*$ such that a universal Turing machine with input d outputs the string x within t steps. Note that, in contrast to KT, here the time t to reconstruct a given string x is a parameter rather than part of the complexity measure of x, and there is no requirement to compute x locally. The minimization version of K^t, denoted $\mathsf{MK^tP}$, needs to decide, for a given binary string x and a size parameter s, if $\mathsf{K}^t(x) \leq s$. We have the following.

▶ **Theorem 2** (Learning from $\mathsf{MK^tP}$: Informal version)**.** *Suppose* $\mathsf{MK^tP}$ *has an efficient one-sided error average-case heuristic over the uniform distribution over inputs. Then for any circuit size bound $s(n) \leq \mathsf{poly}(n)$, the concept class $\mathcal{C} = \mathsf{SIZE}[s(n)]$ is agnostic PAC-learnable in polytime with respect to any ensemble of polysize-samplable distributions $D = \{D_n\}$.*

The conclusion of Theorem 2 is stronger than that of Theorem 1, as it does away with the requirement of explicitness of D. Though we cannot yet reach the same conclusion under average-case easiness of MKTP, we make some partial progress; we show that under *worst-case* easiness of MKTP, learning is possible without the sampling circuit explicitly given.

▶ **Theorem 3** (Learning from worst-case MKTP: Informal version)**.** *Suppose* MKTP *is decidable by an efficient randomized algorithm. Then for any circuit size bound $s(n) \leq \mathsf{poly}(n)$, the concept class $\mathcal{C} = \mathsf{SIZE}[s(n)]$ is agnostic PAC-learnable in polytime with respect to any ensemble of polysize-samplable distributions $D = \{D_n\}$.*

Below we explain our results and proof techniques in more detail.

1.1 Results

In this work, we present agnostic PAC-learners for polynomial-size circuits over efficienly samplable distributions, under assumptions of problems of time-bounded Kolmogorov complexity being easy on average. More specifically, we consider the problem of learning an

unknown target function $f\colon \{0,1\}^n \to \{0,1\}$ with respect to a concept class $\mathcal{C}$ and a class $\mathcal{D}$ of ensembles of distributions. Learnability here is *agnostic* in the sense that f does not necessarily belong to $\mathcal{C}$, and our learner is asked to learn f with error that is just a small additive ε over the disagreement between f and the closest function in $\mathcal{C}$ to f, with high probability. In this case, we say the algorithm achieves ε-agnostic learning; see Section 2.3 for more precise definitions.

We will typically take $\mathcal{C}$ to be $\mathsf{SIZE}[s(n)]$ for some function $s\colon \mathbb{N} \to \mathbb{N}$: that is, the class of functions computable by boolean circuits of size $s(n)$. Our agnostic PAC-learners have access to an *example oracle* $\mathsf{EX}(f, D)$, with each query returning an independent and identically distributed pair (x, b), where x is sampled according to the distribution D and $b = f(x)$. The *sample complexity* of the learning algorithm is the number of queries made to $\mathsf{EX}(f, D)$. Note that our learners may *not* ask membership queries of the target function f.

We will typically take $\mathcal{D}$ to be $\mathsf{Samp}[T(n)]/a(n)$ for some functions $T, a\colon \mathbb{N} \to \mathbb{N}$, i.e., the class of distributions samplable non-uniformly in time $T(n)$ and with $a(n)$ bits of advice. We consider two different kinds of access to the target distribution D. The first is *white-box* access, where the learner is explicitly given the $a(n)$ bits of advice required to sample D (as well as the parameters $T(n)$ and $a(n)$ that define the distribution class $\mathcal{D}$). In this case, we will say that $\mathcal{C}$ is *agnostic PAC-learnable over w.b.-*$\mathsf{Samp}[T(n)]/a(n)$. In the second kind of access to D, the learner is not given the advice to sample D but is given the parameters $T(n)$ and $a(n)$. In this case, we will simply say that $\mathcal{C}$ is *agnostic PAC-learnable over* $\mathsf{Samp}[T(n)]/a(n)$.[1]

We also consider two different notions of time-bounded Kolmogorov complexity. The *minimum* KT*-complexity problem*, MKTP, is the problem of deciding, given a string $x \in \{0,1\}^*$ and a parameter $s \in \mathbb{N}$, whether the KT-complexity of x is at most s. Roughly speaking, KT-complexity is the minimum $|d| + t$ such that a universal TM with oracle access to $d \in \{0,1\}^*$ can compute any individual bit of x in time $t \in \mathbb{N}$. $\mathsf{MK^tP}$ is defined analogously, where K^t-complexity is the minimum description length $|d|$ such that a universal TM U on input d outputs (the whole string) x in time t. See Section 2.2 for formal definitions of these measures of time-bounded Kolmogorov complexity and the associated decision problems.[2]

As mentioned earlier, our notion of an average-case heuristic over the uniform distribution $\mathcal{U}$ over inputs for MKTP or $\mathsf{MK^tP}$ mimics the one-sided error definition of a natural property of [27], where all yes-instances must be accepted, and a constant fraction of all instances must be rejected. Given the extreme sparsity of yes-instances of these problems over the uniform distribution, we easily get required *one-sided error* average-case heuristics for these problems from *errorless* average-case heuristics; the class of errorless randomized heuristics is denoted by AvgBPP (see Section 2.1 for the precise definition). For example, our assumption that there is an efficient errorless randomized heuristic for MKTP under the uniform distribution over inputs will be denoted by $(\mathsf{MKTP}, \mathcal{U}) \in \mathsf{AvgBPP}$.

Learning over explicitly given efficiently samplable distributions

Here we give a more formal statement of our Theorem 1.

---

[1] For context, the first model is that employed in the recent work of [4], and the second model is that employed in the recent work of [16]. In the original PAC-learning framework of [29], the target distribution is allowed to be completely unknown and arbitrary.

[2] $\mathsf{MK^tP}$ has elsewhere been denoted MINKT.

▶ **Theorem 4.** *Suppose* $(\mathsf{MKTP}, \mathcal{U}) \in \mathsf{AvgBPP}$. *Then for any time-constructible function* $s\colon \mathbb{N} \to \mathbb{N}$, *polynomials* $T, a\colon \mathbb{N} \to \mathbb{N}$, *and* $\varepsilon \in (n^{-d}, 1)$ *for a constant* $d > 0$, *the concept class* $\mathsf{SIZE}[s(n)]$ *is* ε*-agnostic PAC-learnable on w.b.-*$\mathsf{Samp}[T(n)]/a(n)$

- *in time* $\mathsf{poly}(n, s(n), T(n), a(n), \varepsilon^{-1})$ *and*
- *with sample complexity* $((s(n) + n)^3 \cdot \varepsilon^{-26})^{1+o(1)}$.

Proof. The theorem follows by combining Theorem 19 and Theorem 36 below. ◀

Learning over unknown efficiently samplable distributions

Below we give a formal statement of our Theorem 2. For $\mathsf{MK^tP}$, as above, we allow errorless *randomized* heuristics.

▶ **Theorem 5.** *Suppose* $(\mathsf{MK^tP}, \mathcal{U}) \in \mathsf{AvgBPP}$. *Then for any time-constructible functions* $s, T, a\colon \mathbb{N} \to \mathbb{N}$ *and* $\varepsilon \in (0, 1)$, *the concept class* $\mathsf{SIZE}[s(n)]$ *is* ε*-agnostic learnable on* $\mathsf{Samp}[T(n)]/a(n)$

- *in time* $\mathsf{poly}(n, s(n), T(n), a(n), \varepsilon^{-1})$ *and*
- *with sample complexity* $((s(n) + a(n) + \log T(n))^3 \cdot \varepsilon^{-8})^{1+o(1)}$.

Proof. The theorem follows by combining Theorem 19 and Theorem 38 below. ◀

Finally, we give a formal statement of our Theorem 3.

▶ **Theorem 6.** *Suppose* $\mathsf{MKTP} \in \mathsf{BPP}$. *Then for any time-constructible functions* $s, T, a\colon \mathbb{N} \to \mathbb{N}$ *and* $\varepsilon \in (0, 1)$, *the concept class* $\mathsf{SIZE}[s(n)]$ *is* ε*-agnostic learnable on* $\mathsf{Samp}[T(n)]/a(n)$ *in time and sample complexity* $\mathsf{poly}(n, s(n), T(n), a(n), \varepsilon^{-1})$.

Proof. The theorem follows by combining Theorem 19 and Theorem 41 below. ◀

1.2 Techniques

All of our proofs work by way of the known reduction, due to Kothari and Livni [22], from agnostic PAC-learning to the task of *correlative RRHS-refutation*. Consider polynomials $s(n)$, $T(n)$, and $a(n)$. Given a concept class $\mathsf{SIZE}[s(n)]$, a distribution class $\mathsf{Samp}[T(n)]/a(n)$, and a tuple of labeled strings $(\langle x^{(1)}, b^{(1)}\rangle, ..., \langle x^{(m)}, b^{(m)}\rangle)$, where each $x^{(i)} \sim D_n$ for some distribution $D \in \mathsf{Samp}[T(n)]/a(n)$, a correlative RRHS-refuter R is asked to distinguish the following two cases:

- A "correlative case", in which the labels b are correlated with the output of some $s(n)$-size circuit f; that is, for each $1 \le i \le m$, independently,

$$\Pr_{x^{(i)} \sim D}\left[b^{(i)} = f(x^{(i)})\right] \ge \frac{1}{2} + \frac{\varepsilon}{2},$$

- and a "random case", in which the labels $b^{(i)}$ are sampled independently and uniformly at random.

Kothari and Livni show that if there is a probabilistic polynomial-time algorithm R satisfying the above conditions, then there is an agnostic learner for f over D. The proof of this statement essentially uses *distribution-specific* boosting algorithms for the agnostic setting, as given by Feldman [12] and Kalai and Kanade [20]. The fact that the distribution D remains the same during polynomially many boosting stages is crucial as it keeps the circuit complexity of the sampler for D polynomially bounded.

For our results, the key intuition is that the concatenated samples in the correlative case will have lower time-bounded Kolmogorov complexity than those in the random case, since the complexity of uniformly random labels $(b^{(1)}, ..., b^{(m)})$ is close to its maximum value

$m + O(1)$ with very high probability. Choosing $m = \mathsf{poly}(n)$ sufficiently larger than the circuit-complexity $s(n)$ of the target function f yields the desired gap between the two cases. In this way, a heuristic algorithm for computing time-bounded K-complexity may be used as a correlative RRHS-refuter.

A first observation is that, regardless of the version of time-bounded K-complexity available as a heuristic algorithm, it is easy to construct a correlative RRHS-refuter working over the uniform distribution. For example, suppose $(\mathsf{MKTP}, \mathcal{U}) \in \mathsf{AvgBPP}$. Let $X := (x^{(1)}, ..., x^{(m)}) \in \{0,1\}^{nm}$ and $b := (b^{(1)}, ..., b^{(m)}) \in \{0,1\}^m$. On one hand, in the correlative case, we will always have

$$\mathsf{KT}(X, b) \leq nm + \ell_s(n) + \delta \cdot m,$$

where $\ell_s(n) \leq O(s(n) \log s(n))$ is the length of an encoding of a circuit for f, and a further $\delta \cdot m$ bits for a constant $\delta < 1$ are used to encode the discrepancy between the labels b and the true outputs of f (see Lemma 34). In particular, given X, it is easy to construct b from the outputs of f (and knowing which of the labels $b^{(i)}$ are incorrect). On the other hand, in the random case, $(X, b) \sim \mathcal{U}_{nm+m}$. Since the KT-complexity of uniformly random strings is usually close to maximum, we have that with high probability,

$$\mathsf{KT}(X, b) \geq nm + m - 10.$$

It is not hard to see that our randomized heuristic for deciding KT-complexity will serve as a randomized distinguisher between these two cases.

For distributions other than uniform, the situation is less straightforward. In particular, our heuristic algorithms are only defined to work well over $\mathcal{U}$; moreover, $\mathsf{KT}(X, b)$ is not necessarily likely to be large. In recent work, Hirahara and Nanashima [16] circumvent these obstacles under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. In particular, they use this assumption to construct a worst-case algorithm approximating K^t within logarithmic additive error. They also use it to prove a worst-case *weak Symmetry of Information* theorem for K^t, which conditionally states that for some polynomial p, for every $X \in \{0,1\}^*$,

$$\mathsf{K}^t(X, b) \geq \mathsf{K}^{p(t)}(X) + |b| - O(\log t)$$

with high probability over a uniformly random string b. The above inequality is used in the random case of RRHS-refutation. In the correlative case, as above, it holds that

$$\mathsf{K}^t(X, b) \leq \mathsf{K}^{t'}(X) + \ell_s(n) + \delta \cdot m$$

for some time-bound $t' < t$. The authors then use the worst-case algorithm for K^t to approximate the value of $\mathsf{K}^t(X, b) - \mathsf{K}^\tau(X)$ for an appropriate choice of τ, thereby distinguishing the two cases. To overcome the technical issue of the different time bounds $p(t)$ and t' in the expressions above, they show that such differences are immaterial in the expectation over an efficiently samplable distribution:[3] for any $D \in \mathsf{Samp}[mT(n)]/a(n)$ and sufficiently large time bound t,

$$\mathop{\mathbb{E}}_{X \sim D} \left[\mathsf{K}^t(X) - \mathsf{K}(X) \right] \leq O(\log(mT(n))) + a(n). \tag{1}$$

In other words, both $\mathsf{K}^t(X)$ and $\mathsf{K}^{t'}(X)$ are likely close enough to $\mathsf{K}(X)$, and therefore close enough to each other.

[3] Note that for $x^{(i)} \sim D$, for $D \in \mathsf{Samp}[T(n)]/a(n)$, we have $X = (x^{(1)}, ..., x^{(m)}) \sim D'$, for $D' \in \mathsf{Samp}[mT(n)]/a(n)$.

1.2.1 Learning from $\mathsf{MK^tP}$

To prove our result for $\mathsf{MK^tP}$, we show that similar arguments may be carried out under a significantly weaker assumption. One issue is that [16] uses the assumption $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ to achieve derandomization, as shown possible by [9]. Roughly speaking, one "encodes" a string x into a distribution $\mathsf{DP}(x)$ such that any efficient algorithm distinguishing $\mathsf{DP}(x)$ from uniform can be used to show that $\mathsf{K}^t(x)$ is small, a process that crucially relies on the derandomization of the DP reconstruction. Such derandomization is not known to hold under the assumption $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, let alone our weaker assumption of $(\mathsf{MK^tP}, \mathcal{U}) \in \mathsf{AvgBPP}$, where $\mathsf{MK^tP}$ is not even known to be NP-hard. In our setting, compression via the DP generator gives a randomized algorithm A that, for any string X and sufficiently large $t \in \mathbb{N}$, outputs a value $\tilde{s} \in \mathbb{N}$ such that

$$\mathsf{pK}^{\mathsf{poly}(t)}(X) - O(\log t) \leq \tilde{s} \leq \mathsf{K}^t(X),$$

where $\mathsf{pK}^{\mathsf{poly}(t)}$ denotes a *probabilistic* measure of time-bounded Kolmogorov complexity. See Section 2.2 for a definition. In general, $\mathsf{pK}^t(X)$ could be much smaller that $\mathsf{K}^t(X)$, so the algorithm A does not appear very useful a priori. However, as we outline below, it turns out to be sufficient for the purposes of learning.

Another challenge in our setting is to argue for Eq. (1) above, which says that $\mathsf{K}^t(X)$ is close to $\mathsf{K}(X)$ in the expectation. In [16], the proof of that statement relies on a conditional *source-coding* theorem for K^t: if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then for any distribution $D \in \mathsf{Samp}[mT(n)]/a(n)$ and $X \in \mathrm{supp}(D)$,

$$\mathsf{K}^{\mathsf{poly}(mT(n))}(X) \lesssim \log(1/D(X)), \tag{2}$$

where $D(X)$ denotes the probability of X under D, and "$\lesssim$" hides the term $O(\log(mT(n))) + a(n)$. Specifically, to prove Eq. (1) from this statement, one observes that

$$\begin{aligned} \mathop{\mathbb{E}}_{X\sim D}\left[\mathsf{K}^t(X)\right] &\lesssim \mathop{\mathbb{E}}_{X\sim D}\left[\log(1/D(X))\right] \\ &= H(D) \\ &\leq \mathop{\mathbb{E}}_{X\sim D}[\mathsf{K}(X)], \end{aligned}$$

where $H(D)$ denotes the Shannon entropy of the distribution D.

In our setting, without derandomization, Eq. (2) is not known to hold. Unconditionally, it is only known that with high probability over $r \sim \mathcal{U}_{\mathsf{poly}(mT(n))}$,

$$\mathsf{K}^{\mathsf{poly}(mT(n))}(X, r) \lesssim \log(1/D(X)) + |r|. \tag{3}$$

That is, source coding for K^t only holds in the presence of additional uniform randomness. A statement of this kind was originally proved in [3]. In analogy with Eq (1), we may use Eq. (3) to prove that

$$\mathbb{E}\left[\mathsf{K}^t(X, r) - \mathsf{K}(X, r)\right] \leq O(\log(mT(n))) + a(n), \tag{4}$$

for $X \sim D$ and $r \sim \mathcal{U}_{\mathsf{poly}(mT(n))}$.

We cope with the necessity of this additional randomness r by *incorporating it* into our correlative RRHS-refuter R. That is, we use the randomness of R to uniformly sample a string r, and rather than approximating $\mathsf{K}^\tau(X)$ and $\mathsf{K}^t(X, b)$, we approximate $\mathsf{K}^\tau(X, r)$ and $\mathsf{K}^t(X, b, r)$. We show the analysis of the RRHS-refutation to be unharmed by this modification.

Importantly, Eq. (4) allows us to make use of our inferior approximation algorithm A described at the beginning of this section. For any strings X and r, $\mathsf{pK}^t(X, r)$ is known to be lower-bounded by the time-unrestricted $\mathsf{K}(X, r)$. Eq. (4) then implies that the expected value of $\mathsf{K}^t(X, r) - \mathsf{pK}^t(X, r)$ will be low, for X sampled from an efficiently samplable distribution and r sampled uniformly at random. Intuitively, there is a "smoothing out" of the differences between different measures of Kolmogorov complexity in the expectation, so the correlative RRHS-refuter we construct may sometimes safely ignore such differences.

Finally, there is the issue of the Symmetry of Information theorem for K^t, which is not known to hold in the absence of derandomization. To get around this, we observe that such a statement is actually *not necessary* for our purposes. Rather, since $\mathsf{K}^t(X, b, r)$ will be close to $\mathsf{K}(X, b, r)$ with high probability over $X \sim D_n^m$, $b \sim \mathcal{U}_m$, and $r \sim \mathcal{U}_{\mathsf{poly}(mT(n))}$, we may simply apply the well-known, unconditional Symmetry of Information theorem for time-unbounded Kolmogorov complexity. This observation has the advantage of simplifying our proofs as well as painting a clearer picture of the true prerequisites of learning.

1.2.2 Learning from MKTP

Many of the tools available in the K^t setting, such as compression via generator reconstruction yielding a worst-to-average reduction, become unavailable in the setting of KT. For this reason, we can no longer apply the framework of [16], and we obtain a model of learning that requires a stronger form of access to the target distribution in question. In this setting, we take advantage of the fact, as described above, that it is easy to learn via KT over the uniform distribution. Our goal is then to reduce the task of learning over arbitrary distributions in $\mathsf{PSamp/poly}$ to that of learning over the uniform distribution. Inspired by a recent work of Binnendyk et al. [4], we employ the *distributional inverters* of [19]. A distributional inverter for a function $g\colon \{0,1\}^* \to \{0,1\}^*$ is an algorithm that, given some $y = g(x)$, outputs a nearly uniformly random member of the set $\{z \mid g(z) = y\}$. It is already known that $(\mathsf{MKTP}, \mathcal{U}) \in \mathsf{AvgBPP}$ implies the existence of such objects for every efficiently computable g (see Section 2.4).

To construct a correlative RRHS-refuter for an arbitrary distribution $D \in \mathsf{PSamp/poly}$, we apply distributional inversion in the following way. Let I be a distributional inverter for the sampler for D, which is a polynomial-size circuit C. Recall that in the problem of correlative RRHS-refutation, we are provided labeled examples $\{(x^{(i)}, b^{(i)})\}$, where either the $b^{(i)}$s are uniformly random, or they are correlated with the outputs $f(x^{(i)})$ of the target function f. Given such pairs $\{(x^{(i)}, b^{(i)})\}$, we apply I to the first part to simulate pairs of the form $\{(r^{(i)}, b^{(i)})\}$, where the $r^{(i)}$s are now from a distribution close to uniform, and the $b^{(i)}$s are either uniformly random, or they are correlated with the outputs $f(C(r^{(i)}))$ of the target function f composed with the sampler C. Using a correlative RRHS-refuter for $f \circ C$ over the uniform distribution, we can distinguish these two cases, thereby distinguishing the two cases of the original problem over D. Because I must have oracle access to the non-uniform circuit C it is inverting, the learner will ultimately require an *explicit* description of C, so that the learner can output a circuit for f with no extra oracle gates.

1.3 Related Work

[16] proved a version of Theorem 2 under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$. In [13], the authors adapted the learning result of [16] to the case of the randomized average-case easiness assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgBPP}$, by showing that the probabilistic Kolmogorov complexity measure pK^t may be used instead of K^t, and proving (under the same average-case easiness

assumption) various results for pK^t (e.g., the existence of a randomized approximation algorithm for pK^t, and the Symmetry of Information). Using [13], it is fairly straightforward to get a learning algorithm from the assumption that $\mathsf{MpK^tP}$ (the minimization problem for pK^t) is in AvgBPP, relying on the properties of pK^t proved in [13]. However, in the present paper, we use a weaker assumption that $\mathsf{MK^tP}$ is in AvgBPP (under the uniform distribution), and avoid using any nontrivial properties of pK^t. Intuitively, the reason we are able to do so is the "smoothing out" phenomenon mentioned above: the time-bounded Kolmogorov complexity measures K^t and pK^t are close to the time-unbounded measure K, in expectation over appropriate efficiently samplable distributions.

Recall that $\mathsf{Partial\text{-}MCSP}$ is the problem of deciding, given a collection of pairs $\{(x_i, b_i)\}$, whether there is a small circuit C such that every $C(x_i) = b_i$. Ilango, Loff, and Oliveira prove that under an average-case easiness assumption about $\mathsf{Partial\text{-}MCSP}$, PAC-learning without membership queries is possible over the uniform distribution [17]. This relies on a reduction of Vadhan [28] from PAC-learning (in a distribution-*independent* sense) to the problem of "RRHS-refutation": namely, the simpler version of correlative RRHS-refutation in which the labels b_i are precisely the outputs of the target concept f applied to the samples x_i. We expect that by using the tools of this work, including correlative RRHS-refutation and distributional inversion, the statement of [17] could be extended to the agnostic setting and arbitrary efficiently samplable distributions, in the sense of our Theorem 1.

2 Preliminaries

2.1 Average-case Complexity

A *distributional problem* is a pair (L, D), where $L \subseteq \{0,1\}^*$ is a language and D is a family of distributions $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$. We denote by $\mathcal{U}$ the family of *parameterized uniform distributions* $\{\mathcal{U}_{\langle n,t_1,\ldots,t_k\rangle}\}_{n,t_1,\ldots,t_k\in\mathbb{N}}$, where k is a constant, each $\mathcal{U}_{\langle n,t_1,\ldots,t_k\rangle} := (\mathcal{U}_n, 1^{t_1}, \ldots, 1^{t_k})$, and $\mathcal{U}_n$ is the uniform distribution over n-bit strings.[4]

▶ **Definition 7** (AvgBPP [7]). *A distributional problem (L, D) belongs to* AvgBPP *if there is an algorithm A and polynomial p such that, on any $n \in \mathbb{N}$, $x \in supp(D_n)$, and $\delta > 0$, $A(x; n, \delta)$ runs in time at most $p(n/\delta)$, and*

1. $\Pr_A [A(x; n, \delta) \notin \{L(x), \perp\}] \leq \frac{1}{10}$;
2. $\Pr_{x\sim D_n} [\Pr_A[A(x; n, \delta) = \perp] \geq 1/10] \leq \delta(n)$.

Such an algorithm A is called a randomized errorless heuristic scheme for (L, D).

2.2 Time-bounded Kolmogorov Complexity

▶ **Definition 8** (KT [1]). *Fix a universal oracle TM $\mathcal{U}$. For strings $x, y \in \{0,1\}^*$, the* KT*-complexity of x given y is defined as*

$$\mathsf{KT}(x \mid y) := \min_{d\in\{0,1\}^*,\, t\in\mathbb{N}} \left\{ |d| + t \mid \forall\, 1 \leq i \leq N+1,\ \mathcal{U}^{d,y}(i) = x_i \text{ in at most } t \text{ steps} \right\},$$

where $x_{N+1} := \perp$, and the notation $\mathcal{U}^{d,y}$ means that $\mathcal{U}$ has random (oracle) access to strings d and y.

[4] Formally, $\langle t_1, \ldots, t_k\rangle$ denotes $\mathsf{Enc}(t_1, \ldots, t_k)$, where $\mathsf{Enc} : \mathbb{N}^* \to \mathbb{N}$ is an efficiently computable and decodable encoding function. Such an encoding function is known to exist by standard techniques; see, for example, [7].

▶ **Definition 9** (K^t). *Fix a universal deterministic TM $\mathcal{U}$. For strings $x, y \in \{0,1\}^*$ and a time bound $t \in \mathbb{N}$, the t-time-bounded Kolmogorov complexity of x given y is defined as*

$$\mathsf{K}^t(x \mid y) := \min_{k \in \mathbb{N}} \left\{ k \;\middle|\; \exists\, w \in \{0,1\}^k,\ \mathcal{U}(w, y) \text{ outputs } x \text{ within } t \text{ steps} \right\}.$$

▶ **Definition 10** (pK^t_δ). *Fix a universal deterministic TM $\mathcal{U}$. For strings $x, y \in \{0,1\}^*$, a time bound $t \in \mathbb{N}$, and $\delta \in [0,1]$, the δ-probabilistic t-time-bounded Kolmogorov complexity of x given y is defined as*

$$\mathsf{pK}^t_\delta(x \mid y) := \min_{k \in \mathbb{N}} \left\{ k \;\middle|\; \Pr_{r \sim \{0,1\}^t} \left[\exists\, w \in \{0,1\}^k,\ \ \mathcal{U}(w, y, r) \text{ outputs } x \text{ within } t \text{ steps} \right] \geq \delta \right\}.$$

▶ **Definition 11** (MKTP and $\mathsf{MK^tP}$). *We define languages*

- $\mathsf{MKTP} := \{(x, 1^s) \mid x \in \{0,1\}^*,\ s \in \mathbb{N},\ \text{and}\ \ \mathsf{KT}(x) \leq s\}$;
- $\mathsf{MK^tP} := \{(x, 1^s, 1^t) \mid x \in \{0,1\}^*,\ s, t \in \mathbb{N},\ \text{and}\ \ \mathsf{K}^t(x) \leq s\}$;

▶ **Proposition 12.** *For any string $x \in \{0,1\}^*$ and time bound $t \in \mathbb{N}$,*

$$\mathsf{pK}^t(x) \leq \mathsf{K}^t(x).$$

▶ **Proposition 13** ([13]). *There is a constant c such that, for any string $x \in \{0,1\}^*$ and time bound $t \in \mathbb{N}$,*

$$\mathsf{K}(x \mid t) \leq \mathsf{pK}^t(x) + c \log |x|.$$

▶ **Proposition 14.** *There is a constant c' such that, for any string $x \in \{0,1\}^*$ and time bound $t \in \mathbb{N}$,*

$$\mathsf{K}^t(x) \leq |x| + c'.$$

▶ **Lemma 15** (Symmetry of Information for Time-unbounded K-complexity [31]). *For every pair of strings $x \in \{0,1\}^*$ and $y \in \{0,1\}^*$,*

$$\mathsf{K}(xy) \geq \mathsf{K}(x) + \mathsf{K}(y \mid x) - O(\log |xy|).$$

2.3 Agnostic PAC-Learning and Correlative RRHS-Refutation

In the PAC-learning framework, one is asked to learn an unknown *concept*: namely, a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ for some $n \in \mathbb{N}$. A *concept class* $\mathcal{C}$ refers to a set of such concepts, and $\mathcal{C}_n$ denotes $\mathcal{C} \cap \{f\colon \{0,1\}^n \to \{0,1\}\}$. One may ask whether $\mathcal{C}$ is PAC-learnable over a class $\mathcal{D}$ of ensembles $D = \{D_n\}_{n \geq 1}$ of distributions D_n. $\mathcal{D}_n$ denotes $\{D_n \mid D \in \mathcal{D}\}$. For a *hypothesis* $h\colon \{0,1\}^n \to \{0,1\}$, define

$$\mathsf{err}_{D_n}(h, f) = \Pr_{x \sim D_n} [h(x) \neq f(x)].$$

We also define the *minimum relative distance* between f and $\mathcal{C}$ with respect to D_n as the disagreement between f and the best-fitting hypothesis $c \in \mathcal{C}$, i.e.,

$$\mathsf{opt}_{\mathcal{C}_n, D_n, f} \;=\; \min_{c \,\in\, \mathcal{C}_n} \mathsf{err}_{D_n}(c, f).$$

Learners are provided an *example oracle* $\mathsf{EX}(f, D_n)$ such that each query returns an independently sampled pair (x, b), where $x \sim D_n$ and $b = f(x)$. We will use the term *sample complexity* to mean the number of queries made to $\mathsf{EX}(f, D_n)$.

▶ **Definition 16** (PAC learning [29]). *Let $\mathcal{C}$ be a concept class, and let D be a class of distributions. We say that $\mathcal{C}$ is PAC-learnable on $\mathcal{D}$ if there is an algorithm A with the following property. For every $n \geq 1$, $\varepsilon > 0$, $\delta > 0$, distribution $D \in \mathcal{D}_n$, and concept $f\colon \{0,1\}^n \to \{0,1\}$ belonging to $\mathcal{C}_n$,*

$$\Pr_{A,\,\mathsf{EX}(f,D)}\left[A^{\mathsf{EX}(f,D)}(n,\varepsilon,\delta) \text{ outputs a hypothesis } h \text{ such that } \mathsf{err}_D(h,f) \leq \varepsilon\right] \geq 1-\delta,$$

where the probability is over the internal randomness of A and random examples provided by $\mathsf{EX}(f,D)$.

The following definition of *agnostic* PAC learning is a generalization of the PAC learning definition above to the case where a function f to be learned is not necessarily from the concept class $\mathcal{C}$. In this case, the hypothesis h output by the learning algorithm should have an error close to the minimum relative distance between f and the concept class $\mathcal{C}$.

▶ **Definition 17** (Agnostic PAC learning [21]). *Let $\mathcal{C}$ be a concept class, and let $\mathcal{D}$ be a class of distributions. We say that $\mathcal{C}$ is ε-agnostic PAC-learnable on $\mathcal{D}$ if there is an algorithm A with the following property. For every $n \geq 1$, $\varepsilon > 0$, $\delta > 0$, distribution $D \in \mathcal{D}_n$, and concept $f\colon \{0,1\}^n \to \{0,1\}$,*

$$\Pr_{A,\,\mathsf{EX}(f,D)}\left[A^{\mathsf{EX}(f,D)}(n,\varepsilon,\delta) \text{ outputs a hypothesis } h \text{ such that } \mathsf{err}_D(h,f) \leq \mathsf{opt}_{\mathcal{C}_n,D,f}+\varepsilon\right] \geq 1-\delta.$$

▶ **Definition 18** (Correlative RRHS-Refutation). *Let $\mathcal{C}$ be a concept class, and let $D = \{D_n\}_{n\geq 1}$ be an ensemble of distributions. A randomized algorithm R is a* ε-correlative random-right-hand-side-refuter (ε-correlative RRHS-refuter) *for $\mathcal{C}$ on D with sample complexity m provided it satisfies the following. Given input parameters $n \in \mathbb{N}$ and $\varepsilon \in (0,1)$, as well as a set*

$$S = \left(\left\langle x^{(1)}, b^{(1)}\right\rangle, \ldots, \left\langle x^{(m)}, b^{(m)}\right\rangle\right)$$

of samples, where $x^{(i)} \in \{0,1\}^n$ and $b^{(i)} \in \{0,1\}$ for $i \in [m]$;

- ***Soundness:*** *Suppose the samples S are i.i.d. from a distribution D' on $\{0,1\}^n \times \{0,1\}$ such that the marginal on $\{0,1\}^n$ equals D_n, and for some $f \in \mathcal{C}_n$,*

  $$\Pr_{\left\langle x^{(i)}, b^{(i)}\right\rangle \sim D'}\left[b^{(i)} = f(x^{(i)})\right] \geq \frac{1}{2} + \frac{\varepsilon}{2}.$$

  Then,

  $$\Pr_{S,R}\left[R(n,\varepsilon,S) = \textit{correlative}\right] \geq 2/3.$$

- ***Completeness:*** *Suppose the samples S are i.i.d. with $x^{(1)}, \ldots, x^{(m)} \sim D_n$ and $b^{(1)}, \ldots, b^{(m)} \sim \mathcal{U}$. Then,*

  $$\Pr_{S,R}\left[R(n,\varepsilon,S) = \textit{random}\right] \geq 2/3.$$

Kothari and Livni [22] prove an equivalence between distribution-specific agnostic PAC learning and RRHS-refutation. We will be using the following direction from RRHS-refutation to agnostic learning.

▶ **Theorem 19** (Agnostic Learning from RRHS-Refutation [22]). *Let $\mathcal{C}$ be a concept class, and let $D = \{D_n\}_{n\geq 1}$ be an ensemble of distributions. If there exists an ε-correlative RRHS-refuter for $\mathcal{C}$ on D with sample complexity $m(n,\varepsilon)$ and running time $T(n,\varepsilon)$, then $\mathcal{C}$ is (2ε)-agnostic PAC-learnable over D with*

- *sample complexity $O\big(m(n,\varepsilon/2)^3 \cdot \varepsilon^{-2}\big)$, and*
- *running time $O\big(T(n,\varepsilon/2) \cdot m(n,\varepsilon/2)^2 \cdot \varepsilon^{-2}\big)$.*

The proof of the above theorem relies on *distribution-specific* boosting algorithms for the agnostic setting, such as those of Feldman [12] and Kalai and Kanade [20]. These algorithms transform a weak agnostic learner over some distribution into a strong agnostic learner over that same distribution; they work by adaptively modifying the labels of example points rather than the distributions on those points as is typically the case in boosting. Interestingly, in the agnostic setting, it is possible to accomplish this without a superpolynomial increase in the running time of the learner.

2.4 Inversion

In this section, we cover definitions of *inversion* of functions, which are the negations of corresponding definitions of the existence of one-way functions. Throughout, we take the word "function" to include *auxiliary input* functions in the sense of Ostrovsky and Wigderson, in which both function and potential inverter have access to the same non-uniform input (denoted y below) [26].

▶ **Definition 20** (Invertible functions). *Consider a function $g(y,x)$ computable uniformly in polynomial time. The function g is said to be* weakly invertible *if there is a probabilistic polynomial-time Turing machine I and a constant b such that for every $n \in \mathbb{N}$ and for every $y \in \{0,1\}^*$,*

$$\Pr_{x\sim\mathcal{U}_n}[g(y,I(y,g(y,x))) = g(y,x)] \geq \frac{1}{n^b}.$$

The function g is said to be strongly invertible *if for every constant d there is a probabilistic polynomial-time Turing machine I such that for every $n \in \mathbb{N}$ and for every $y \in \{0,1\}^*$,*

$$\Pr_{x\sim\mathcal{U}_n}[g(y,I(y,g(y,x))) = g(y,x)] \geq 1 - \frac{1}{n^d}.$$

▶ **Definition 21** (Statistical Indistinguishability). *Two probability distributions D and D' are* statistically indistinguishable within δ *if for all $T \subseteq \{0,1\}^n$,*

$$\left|\Pr_{x\sim D_n}[x \in T] - \Pr_{x\sim D'_n}[x \in T]\right| \leq \delta.$$

We denote this as $D \equiv_\delta D'$.

▶ **Definition 22** (Distributionally invertible functions). *Consider a function $g(y,x)$ computable uniformly in polynomial time. The function g is said to be* distributionally invertible *if for every constant $b > 0$ there is a probabilistic polynomial-time oracle Turing Machine I such that for every $n \in \mathbb{N}$ and $y \in \{0,1\}^*$,*

$$(x, g(y,x)) \equiv_{n^{-b}} (I(y,g(y,x)), g(y,x)),$$

*where $x \sim \mathcal{U}_n$. We refer to the machine I as an n^{-b}-*distributional inverter*.*

▶ **Lemma 23** ([30]). *If every function computable in polynomial time is weakly invertible, then every such function is strongly invertible.*

▶ **Lemma 24** ([19]). *If every function computable in polynomial time is strongly invertible, then every such function is distributionally invertible.*

▶ **Lemma 25** ([1]). *If $(\mathsf{MKTP},\mathcal{U}) \in \mathsf{AvgBPP}$, then every function computable in polynomial time is weakly invertible.*

▶ **Corollary 26.** *If $(\mathsf{MKTP},\mathcal{U}) \in \mathsf{AvgBPP}$, then every function computable in polynomial time is distributionally invertible.*

2.5 Direct Product Generator and pK^t-Compression

▶ **Definition 27** (Direct Product Generator). *For $n, k \in \mathbb{N}$, the k-wise direct product generator $\mathsf{DP}_k : \{0,1\}^n \times \{0,1\}^{nk} \to \{0,1\}^{nk+k}$ is the function defined by*

$$\mathsf{DP}_k(x; z^1, ..., z^k) = (z^1, ..., z^k; \langle x, z^1\rangle, ..., \langle x, z^k\rangle),$$

where $\langle -, -\rangle$ denotes the inner product $\langle x, y\rangle = \left(\sum_{i=1}^{|x|} x_i y_i\right) \mod 2$.

▶ **Lemma 28** (Probabilistic pK^t Reconstruction [13]). *There is a polynomial p' with the following property. For $\varepsilon > 0$, $x \in \{0,1\}^n$, $s \in \mathbb{N}$, and $k \in \mathbb{N}$ satisfying $k \leq 2n$, let D be a randomized algorithm that takes an advice string β, runs in time t_D, and ε-distinguishes $\mathsf{DP}_k(x; \mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Then*

$$\mathsf{pK}^{p'(t_D \cdot n/\varepsilon)}(x \mid \beta) \leq k + \log p'(t_D \cdot n/\varepsilon).$$

2.6 Source Coding Theorem

The following lemma is very similar to one of [3], but with a greater probability of success on the right-hand side, which is necessary for the application in Lemma 37. For completeness, we present a slight modification of a proof due to [2], which uses hashing.

▶ **Lemma 29.** *The following holds unconditionally. There exist polynomials p and q such that for any $T, a : \mathbb{N} \to \mathbb{N}$, $n \in \mathbb{N}$, $D \in \mathsf{Samp}[T(n)]/a(n)$, and $x \in \mathsf{Supp}(D_n)$,*

$$\Pr_{r \sim \mathcal{U}_{3T(n)}}\left[\mathsf{K}^{p(T(n))}(x, r) \leq \log(1/D_n(x)) + |r| + a(n) + \log p(T(n))\right] \geq 1 - \frac{1}{4T(n)},$$

where $D_n(x)$ denotes the probability of x under D_n.

Proof. Let A be a non-uniform algorithm sampling $D \in \mathsf{Samp}[T(n)]/a(n)$. That is, there is some $\alpha \in \{0,1\}^{a(n)}$ such that for any $x \in \mathrm{supp}(D_n)$,

$$\Pr_{w \sim \mathcal{U}_{T(n)}}[A(w; \alpha, 1^n)] = D_n(x).$$

Let s be the smallest integer such that $D_n(x) \geq 2^{-s}$. Define $\ell := T(n)$ and $k := \ell - s - \log(8T(n))$. Consider a universal hash function family $\mathcal{H} = \{h : \{0,1\}^\ell \to \{0,1\}^k\}$. For each $h \in \mathcal{H}$ and $w \in \{0,1\}^{T(n)}$, $h(w) = U \cdot w + v$ for some binary Toeplitz matrix U of dimension $k \times \ell$ and binary vector v of dimension k. Define a set

$$S_x := \{w \in \{0,1\}^{T(n)} \mid A(w; \alpha, 1^n) = x\}.$$

For $h \sim \mathcal{H}$, define a random variable $X := |S_x \cap h^{-1}(0^k)|$. Note that $|S_x| = D_n(x) \cdot 2^{T(n)} \geq 2^{\ell - s}$. Then $|S_x|/2^k \geq 8T(n)$, and by universality,

$$\mathsf{Var}[X] \leq \mathbb{E}[X] = \frac{|S_x|}{2^k}.$$

By Chebyshev's Inequality,

$$\begin{aligned}\Pr[X = 0] &\leq \Pr\left[|X - \mathbb{E}[X]| \geq \mathbb{E}[X]\right] \\ &\leq \mathsf{Var}[X]/\mathbb{E}[X]^2 \\ &\leq 1/8T(n).\end{aligned}$$

Now define a random variable $Y = |h^{-1}(0^k)|$, where $h \sim \mathcal{H}$. Note that $\mathbb{E}[Y] = 2^\ell/2^k = 2^{s+\log(8T(n))}$. Then by Markov's Inequality,

$$\begin{aligned}\Pr[Y \geq 2^{s+2\log(8T(n))}] &= \Pr[Y \geq 8T(n) \cdot \mathbb{E}[Y]] \\ &\leq 1/8T(n).\end{aligned}$$

By a union bound,

$$\Pr[X = 0 \text{ or } Y \geq 2^{s+2\log(8T(n))}] \leq 1/4T(n).$$

Assume $X > 0$ and $Y < 2^{s+2\log(8T(n))}$. It is possible to represent x with descriptions of the hash function h, the index of a string $w \in S_x$ in the set $h^{-1}(0^k)$, and the advice string α used in the sampler A. In particular, x may be recovered by performing Gaussian elimination to compute the set $h^{-1}(0^k)$ from the description (U, v) of h, locating w in this set, and then returning the output of $A(w; \alpha, 1^n)$. This requires $|(U, v)| < 3T(n)$ bits to describe h, at most $\log Y \leq s + 2\log(8T(n)) \leq \log(1/D_n(x)) + 1 + 2\log(8T(n))$ bits to describe the position of w in $h^{-1}(0^k)$, and $|\alpha| = a(n)$ bits to run the sampler A. Define the "random" string $r \in \{0,1\}^{3T(n)}$ as the description (U, v) of h. Overall, we have that with probability at least $1 - 1/4T(n)$ over r sampled uniformly,

$$\mathsf{K}^{p(T(n))}(x, r) \leq \log(1/D_n(x)) + |r| + a(n) + \log p(T(n))$$

for some polynomial p. ◀

3 Approximating K^t

▶ **Lemma 30** (implicit in [15]). *If* $(\mathsf{MK^tP}, \mathcal{U}) \in \mathsf{AvgBPP}$*, then there is a polynomial* p *such that the following promise problem is in* promiseBPP*:*

$$\Pi_{\mathsf{YES}} := \left\{ (x, 1^s, 1^t) \mid x \in \{0,1\}^*,\ s, t \in \mathbb{N},\ t \geq |x|,\ \textit{and}\ \mathsf{K}^t(x) \leq s \right\},$$
$$\Pi_{\mathsf{NO}} := \left\{ (x, 1^s, 1^t) \mid x \in \{0,1\}^*,\ s, t \in \mathbb{N},\ t \geq |x|,\ \textit{and}\ \mathsf{pK}^{p(t)}(x) > s + \log p(t) \right\}.$$

Proof. Let the input $(x, 1^s, 1^t)$ be given, where $x \in \{0,1\}^n$ and $s \leq n + O(1)$. Define

$$k := s + 2\log q(t), \text{ and}$$
$$s' := s + nk + \log q(t),$$

where q is a polynomial chosen later.

Let B_0 be a randomized errorless heuristic scheme for $(\mathsf{MK^tP}, \mathcal{U})$, with failure probability $1/n$. Let B be the modification of B_0 that outputs "1" whenever B_0 would output "$\perp$". Note that on yes-instances of $\mathsf{MK^tP}$, B errs with probability at most $1/10$ over its own internal randomness.

Define another algorithm B' as follows:

On input $(x, 1^s, 1^t)$, sample $z \sim \mathcal{U}_{nk}$ and then output $B(\mathsf{DP}_k(x; z), 1^{s'}, 1^{q(t)})$.

In the remainder of the proof, we argue that B' solves $(\Pi_{\mathsf{YES}}, \Pi_{\mathsf{NO}})$ correctly with high probability in the worst case.

First, suppose $(x, 1^s, 1^t) \in \Pi_{\mathsf{YES}}$. Observe that for our choice of k, given any $x \in \{0,1\}^n$ and $z \in \{0,1\}^{nk+k}$, it is possible to compute $\mathsf{DP}_k(x;z)$ in polynomial time. Thus, we let q be a polynomial such that for any $z \in \{0,1\}^{nk}$ and sufficiently large $t \in \mathbb{N}$,

$$\begin{aligned}\mathsf{K}^{q(t)}(\mathsf{DP}_k(x;z)) &\leq \mathsf{K}^t(x) + |z| + \log q(t)\\ &\leq s + |z| + \log q(t)\\ &= s'.\end{aligned}$$

Then by definition of B, for $(x, 1^s, 1^t) \in \Pi_{\mathsf{YES}}$,

$$\Pr[B'(x, 1^s, 1^t) = 1] \geq 9/10,$$

where the above probability is over the inner randomness of B and $z \sim \mathcal{U}_{nk}$.

Now suppose $(x, 1^s, 1^t) \in \Pi_{\mathsf{NO}}$. For a contradiction, suppose

$$\Pr[B'(x, 1^s, 1^t) = 1] = \Pr_{B,z}[B(\mathsf{DP}_k(x;z), 1^{s'}, 1^{q(t)}) = 1] > 1/4. \tag{5}$$

By a counting argument, for randomly selected $w \sim \mathcal{U}_{nk+k}$,

$$\Pr_w\left[\mathsf{K}^{q(t)}(w) \leq s'\right] \leq \frac{2^{s'}}{2^{nk+k}} = \frac{1}{q(t)}.$$

Then by definition of B,

$$\begin{aligned}\Pr_{B,w}\left[B(w, 1^{s'}, 1^{q(t)}) = 1\right] &= \frac{1}{10} + \frac{1}{n} + \frac{1}{q(t)}\\ &< 1/8.\end{aligned} \tag{6}$$

Comparing Equations (5) and (6), we see that $B(-, 1^{s'}, 1^{q(t)})$ (1/8)-distinguishes $\mathsf{DP}_k(x;\mathcal{U}_{nk})$ from $\mathcal{U}_{nk+k}$. Then by Lemma 28, for some polynomial p',

$$\begin{aligned}\mathsf{pK}^{p'(t)}(x) &\leq k + O(\log t)\\ &= s + O(\log t).\end{aligned}$$

In other words, for an appropriate choice of the polynomial p in the statement of the lemma, $(x, 1^s, 1^t)$ is *not* in Π_{NO}. This gives a contradiction. We conclude that for $(x, 1^s, 1^t) \in \Pi_{\mathsf{NO}}$,

$$\Pr\left[B'(x, 1^s, 1^t) = 1\right] \leq 1/4. \qquad \blacktriangleleft$$

▶ **Lemma 31** ([15]). *If $(\mathsf{MK^tP}, \mathcal{U}) \in \mathsf{AvgBPP}$, then there exists a polynomial p and a randomized algorithm A that on input $(x, 1^t)$, where $x \in \{0,1\}^n$ and $t \in \mathbb{N}$, runs in time $\mathsf{poly}(n,t)$ and with probability at least $1 - 2^{-n}$ outputs an integer $\widetilde{s}$ such that*

$$\mathsf{pK}^{t^c}(x) - \log p(t) \;\leq\; \widetilde{s} \;\leq\; \mathsf{K}^t(x).$$

Proof. Consider the polynomial-time randomized algorithm B' that solves the promise problem from Lemma 30. By standard success amplification, we may assume that the error of B' is at most 2^{-2n} on inputs satisfying the promise. Algorithm A runs B' on $(x, 1^s, 1^t)$ for $s = 1, 2, \ldots, n + \log n$, and outputs the first $\widetilde{s}$ such that $B'(x, 1^{\widetilde{s}}, 1^t) = 1$. If B' never accepts, A simply outputs $n + \log n$.

On one hand, if $s = \mathsf{K}^t(x)$, then $(x, 1^s, 1^t) \in \Pi_{\mathsf{YES}}$, so $\Pr[B'(x, 1^s, 1^t) = 1] \geq 1 - 2^{-2n}$. On the other, if $s < \mathsf{pK}^{p(t)}(x) - \log p(t)$, then $(x, 1^s, 1^t) \in \Pi_{\mathsf{NO}}$, so $\Pr[B'(x, 1^s, 1^t) = 1] \leq 2^{-2n}$.

By a union bound, with probability at least $1 - 2^{-n}$, $\widetilde{s}$ has the desired property. $\blacktriangleleft$

4 Agnostic Learning from Heuristics for K-complexity

In what follows, for a distribution D and $m \in \mathbb{N}$, D^m will denote the distribution $(x^{(1)}, ..., x^{(m)})$ where $x^{(i)} \sim D$ for $i \in [m]$. Moreover, $\ell_s(n) \leq O(s(n) \log s(n))$ will denote the number of bits needed to encode a function $f \in \mathsf{SIZE}[s(n)]$.

▶ **Lemma 32** ([16]). *There exists a polynomial t' such that for any $m \geq n \in \mathbb{N}$, string $b \in \{0,1\}^m$, function $f : \{0,1\}^n \to \{0,1\}$, $X = (x^{(1)}, ..., x^{(m)}) \in (\{0,1\}^n)^m$, and $\delta \in (0,1)$ satisfying*

$$\left|\{i \in [m] \mid b_i = f(x^{(i)})\}\right| \geq (1/2 + \delta) \cdot m,$$

we have that for any $r \in \{0,1\}^*$,

$$\mathsf{K}^{t'(m)}(b \mid X, r) \leq \ell_s(n) + \left(1 - 2\delta^2\right) \cdot m.$$

Proof. Given X, we can compute $f(x^{(1)}), \ldots, f(x^{(m)})$ in time $\mathsf{poly}(m \cdot \ell_s(n))$ using the encoding of f, which requires $\ell_s(n)$ bits. Note that b and $f(x^{(1)}), \ldots, f(x^{(m)})$ disagree on at most $(1/2 - \delta) \cdot m$ coordinates. So to recover b, it suffices to encode the string $e \in \{0,1\}^m$ such that $e_i = 1$ iff $f(x^{(i)}) \neq b_i$. We will show that $\mathsf{K}^{\mathsf{poly}(m)}(e) \leq (1 - 2\delta^2) \cdot m$, which will conclude the proof of the lemma.

Note that e has hamming weight at most $m' = (1/2 - \delta) \cdot m$. Every m'-size subset of an m-size set can be represented using $\log_2 \binom{m}{m'}$ bits, via the combinatorial number system, with both encoding and decoding algorithms running in time polynomial in m (see, e.g., [14] for details). Using standard inequalities for binomial coefficients and the binary entropy function H_2, we get

$$\begin{aligned}\log_2 \binom{m}{m'} &\leq \log_2 2^{H_2(m'/m) \cdot m} \\ &= H_2\left(1/2 - \delta\right) \cdot m \\ &\leq \left(1 - 2\delta^2\right) \cdot m,\end{aligned}$$

as required. ◀

We will also need a lemma similar to the above for the case of KT: that is, bounding the KT-complexity of the labels b in the case that they correlate with a function f. Lemma 32 is insufficient as-is, since the time bound $t'(m)$ would render $\mathsf{KT}(b)$ trivial. To overcome this issue, we use an encoding scheme from Golovnev et al. for strings of bounded hamming weight.

▶ **Lemma 33** ([14]). *For some $m, m' \in \mathbb{N}$ and $e \in \{0,1\}^m$, suppose e has hamming weight at most m'. Then there is a string e' of length at most $\log \binom{m}{m'} + m^{3/4}$ such that for all $1 \leq i \leq m$, e_i can be computed with random access to e' in time $m^{2/3}$.*

▶ **Lemma 34.** *For any $m, n \in \mathbb{N}$, string $b \in \{0,1\}^m$, function $f : \{0,1\}^n \to \{0,1\}$, $X = (x^{(1)}, ..., x^{(m)}) \in (\{0,1\}^n)^m$, $r \in \{0,1\}^*$, and $\delta \in (0,1)$ satisfying*

$$\left|\{i \in [m] \mid b_i = f(x^{(i)})\}\right| \geq (1/2 + \delta) \cdot m,$$

we have that

$$\mathsf{KT}(X, b, r) \leq \mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(m)}), r) + \left(1 - 2\delta^2\right) \cdot m + 2m^{3/4}.$$

Proof. It is clear that any bit of X or r can be computed in time and description size upper-bounded by $\mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(m)}), r)$. To compute a bit b_i of b, for $i \in [m]$, we observe the following. As in Lemma 32, let $e \in \{0,1\}^m$ be such that $e_i = 1$ iff $f(x^{(i)}) \neq b_i$. Then b_i is $f(x^{(i)}) \oplus e_i$. Note that the the hamming weight of e is at most $m' := (1/2 - \delta) \cdot m$. Applying Lemma 33, e_i may be computed in time at most $m^{2/3}$ from a description e' of length at most

$$\log \binom{m}{m'} + m^{3/4}.$$

Arguing as in Lemma 32, we upper-bound the above by $(1 - 2\delta^2) \cdot m + m^{3/4}$.

To compute a bit b_i, we first use time and description size $\mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(m)}), r)$ to obtain the corresponding $f(x^{(i)})$. Then, given $f(x^{(i)})$, b_i may be computed in time at most $m^{2/3} + O(1)$ from a description of e' of size at most $\left(1 - 2\delta^2\right) \cdot m + m^{3/4}$. This concludes the proof. ◀

4.1 Learning over the Uniform Distribution from MKTP

Here, we construct a correlative RRHS-refuter, working over distributions that are statistically close to uniform, under the assumption that MKTP is easy on average. In the next section, we will reduce the case of arbitrary efficiently samplable distributions to this case.

▶ **Theorem 35.** *If* $(\mathsf{MKTP}, \mathcal{U}) \in \mathsf{AvgBPP}$*, then for any time-constructible function* $s\colon \mathbb{N} \to \mathbb{N}$*, constants* $c, \zeta > 0$*, and any family of distributions* D *such that* $D_n \equiv_{n^{-c}} \mathcal{U}_n$*, there is an* ε*-correlative RRHS-refuter for* $\mathsf{SIZE}[s(n)]$ *under* D_n *taking parameters* $n \in \mathbb{N}$ *and* $\varepsilon \in (0,1)$ *with sample complexity*

$$m(n, \varepsilon) := \left(\frac{s(n) + n}{\varepsilon^8}\right)^{1+\zeta}$$

and running time $\mathsf{poly}\left(n, \varepsilon^{-1}, s(n)\right)$.

Proof. Let A_0 be a randomized errorless heuristic scheme for $(\mathsf{MKTP}, \mathcal{U})$ with failure probability $1/n$. Let A be the algorithm that simulates A_0 and outputs "correlative" whenever it would output "1" or "$\perp$", and "random" whenever it would output "0". Note that on yes-instances of MKTP, A errs with probability at most $1/10$ over its own internal randomness.

The (correlative) RRHS-refuter R. On input $n \in \mathbb{N}$, $\varepsilon > 0$, and a set

$$S = \left(\left\langle x^{(1)}, b^{(1)} \right\rangle, \ldots, \left\langle x^{(m)}, b^{(m)} \right\rangle\right)$$

of samples, let $X := (x^{(1)}, \ldots, x^{(m)})$ and $b := (b^{(1)}, \ldots, b^{(m)})$. R is defined as follows.

1. Compute $\theta := mn + (1 - \varepsilon^2/16) \cdot m$.
2. Evaluate $A((X, b), 1^\theta)$. Output "correlative" if A accepts, and output "random" otherwise.

Correlative Case (Soundness). Suppose the labeled examples in S are sampled i.i.d from some distribution D' on $\{0,1\}^n \times \{0,1\}$, whose marginal on $\{0,1\}^n$ is given by D_n, and there exists $f \in \mathsf{SIZE}[s(n)]$ such that

$$\Pr_{\left\langle x^{(i)}, b^{(i)} \right\rangle \sim D'} \left[b^{(i)} = f(x^{(i)})\right] \geq \frac{1}{2} + \frac{\varepsilon}{2}.$$

In this case, by a Chernoff bound, the probability over $S \sim (D')^m$ that

$$\left|\{i \in [m] \mid b_i = f(x^{(i)})\}\right| < (1/2 + \varepsilon/4) \cdot m$$

is at most $\exp(-2m(\varepsilon/4)^2) \leq o(1)$. So with probability $1 - o(1)$, the conditions of Lemma 34 are met. Now observe that

$$\mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(m)})) \leq mn + 2\ell_s(n) + 2n,$$

using an mn-bit description of X to obtain any bit of X in constant time, along with an $\ell_s(n)$-bit description of a circuit computing f to obtain any bit $f(x^{(i)})$ from X in time at most $\ell_s(n) + 2n$. This, along with Lemma 34 (with r the empty string), implies that

$$\mathsf{KT}(X, b) \leq mn + \left(1 - \varepsilon^2/8\right) \cdot m + 2\ell_s(n) + 2n + 2m^{3/4}. \tag{7}$$

Finally, by our choice of $m = \omega\left((s(n) + n) \cdot \varepsilon^{-8}\right)$,

$$m > \frac{32\left(\ell_s(n) + n + m^{3/4}\right)}{\varepsilon^2};$$

re-written and combined with Eq. (7),

$$\begin{aligned}\mathsf{KT}(X, b) &\leq mn + \left(1 - \varepsilon^2/8\right) \cdot m + 2\ell_s(n) + 2n + 2m^{3/4} \\ &< mn + \left(1 - \varepsilon^2/16\right) \cdot m \\ &= \theta.\end{aligned}$$

By definition of A, R will output "correlative" with probability at least $9/10 - o(1) > 2/3$.

Random Case (Completeness). Suppose (X, b) is sampled from the distribution $(D_n^m, \mathcal{U}_m)$. Note that for $X \sim \mathcal{U}_n^m$ and $b \sim \mathcal{U}_m$, it holds that

$$\Pr_{X,b}[\mathsf{KT}(X, b) > mn + m - 10] \geq 9/10.$$

Then by the definition of statistical distance, with probability at least $9/10 - o(1)$ over $X \sim D_n^m$ and $b \sim \mathcal{U}_m$,

$$\begin{aligned}\mathsf{KT}(X, b) &> mn + m - 10 \\ &> \theta.\end{aligned}$$

In other words, $((X, b), 1^\theta) \notin \mathsf{MKTP}$.

Now, since the failure probability of our heuristic A is at most $1/10 + 1/n$ over the uniform distribution, the definition of statistical distance implies that its failure probability is at most $1/10 + o(1)$ over the distribution $(D_n^m, \mathcal{U}_m)$.

Overall, by a union bound, R outputs "random" with probability at least $4/5 - o(1) > 2/3$. ◀

4.2 Learning over PSAMP/poly from MKTP

In this section, we generalize the previous theorem to give correlative RRHS-refuters working over arbitrary efficiently samplable distributions. In particular, we reduce to the case of a nearly-uniform distribution by inverting the circuit that samples our given target distribution.

This requires *distributional inversion* as defined by Impagliazzo and Luby [19], which is possible under the assumption of MKTP being easy on average.[5]

▶ **Theorem 36.** *Suppose* $(\mathsf{MKTP}, \mathcal{U}) \in \mathsf{AvgBPP}$. *Consider any time-constructible function* $s : \mathbb{N} \to \mathbb{N}$, *polynomials* $T, a : \mathbb{N} \to \mathbb{N}$, *constant* $\zeta > 0$, *and* $\varepsilon \in (n^{-d}, 1)$ *for a constant* $d > 0$. *Let* $D = \{D_n\}_{n\in\mathbb{N}}$ *be a family of distributions such that each* D_n *is samplable in time* $T(n)$ *with* $a(n)$ *bits of non-uniform advice* α_n. *There is an algorithm which, given* α_n *and parameters* $n \in \mathbb{N}$ *and* ε, *is an* ε*-correlative RRHS-refuter for* $\mathsf{SIZE}[s(n)]$ *under* D_n. *This RRHS-refuter has sample complexity*

$$m(n, \varepsilon) := \left(\frac{s(n) + n}{\varepsilon^8}\right)^{1+\zeta}$$

and running time $\mathsf{poly}\big(n, T(n), a(n), s(n), \varepsilon^{-1}\big)$.

Proof. By Corollary 26, every function $g(y, x)$ computable in polynomial time is distributionally invertible. In particular, let I be a $\varepsilon/4$-distributional inverter for the function g that evaluates the Boolean circuit y on the input string x. Let $\{C_n\}_{n\in\mathbb{N}}$ be the family of circuits that sample D. In particular, each C_n applies the $T(n)$-time sampler for D_n along with the advice α_n. By the definition of distributional inversion (Definition 22), we have that for all sufficiently large $n \in \mathbb{N}$,

$$(I(C_n, C_n(w)), C_n(w)) \equiv_{\varepsilon/4} (w, C_n(w)), \tag{8}$$

where $w \sim \mathcal{U}_\ell$, $\ell \le a(n)$, and I runs in time $\mathsf{poly}(T(n), a(n))$.

Given labeled samples of the form (x, b), where $x \sim D_n = C_n(\mathcal{U}_\ell)$, one may apply I to the first part to simulate labeled samples of the form (r', b), where $r' \in \{0,1\}^\ell$. Specifically, $r' \sim D'_\ell$, where D'_ℓ is the distribution $I(C_n, C_n(\mathcal{U}_\ell))$ sampled by the circuit $C'_\ell(-) := I(C_n, C_n(-))$. By Eq. (8), $D'_\ell \equiv_{\varepsilon/4} \mathcal{U}_\ell$.

We will reduce to the case of a nearly-uniform distribution: namely, the case of Theorem 35. Consider a target function f computable in $\mathsf{SIZE}[s(n)]$. By Theorem 35, since D' is statistically close to uniform, there is a correlative RRHS-refuter R' for $f \circ C_n$ over D' with parameter $\varepsilon' := \varepsilon/2$ that has sample complexity $m = \big((s(n) + n)/\varepsilon^8\big)^{1+\zeta}$ and running time $\mathsf{poly}(n, s(n), \varepsilon^{-1})$. To get a correlative RRHS-refuter R for f over D, we simply return the output of this R' on the simulated examples (r', b). Note that R takes time $\mathsf{poly}(n, T(n), a(n), s(n), \varepsilon^{-1})$ overall.

We now argue that in the "random" case of the original problem, R will output "random" with high probability, and likewise for the "correlative" case. In the random case, the labels b are simply sampled from the uniform distribution $\mathcal{U}$, so R will output "random" with probability at least $2/3$, by the correctness of R'. In the correlative case, b is such that

$$\Pr_{x\sim D_n}[b = f(x)] \ge \frac{1}{2} + \frac{\varepsilon}{2}. \tag{9}$$

We would now like to show that the above probability is not too much smaller when x is sampled from $C_n(D'_\ell)$ rather than $D_n = C_n(\mathcal{U}_\ell)$. Define a set

$$T := \{(r, x) \mid x = C_n(r)\}$$

[5] Similar ideas are employed in the work of Binnendyk et al. [4], which shows that PAC-learning with membership queries over arbitrary efficiently samplable distributions is possible under the existence of natural properties.

and note that samples from the distribution $(r, C_n(r))$, for $r \sim \mathcal{U}_\ell$, belong to T with probability 1. By the property of distributional inversion, ie. Eq. (8), samples from the distribution $(I(C_n, C_n(r)), C_n(r)) = (C'_\ell(r), C_n(r))$, for $r \sim \mathcal{U}_\ell$, belong to the set T with probability at least $1 - \varepsilon/4$. Whenever this holds, by definition of T, we have that $C_n(C'_\ell(r)) = C_n(r)$. Particularly, $f(C_n(r')) = f(x)$, for $r' = C'_\ell(r)$ and $x = C_n(r)$. Then by a union bound with Eq. (9), in the correlative case of the original problem,

$$\Pr_{r' \sim D'_\ell}[b = f(C_n(r'))] \geq \frac{1}{2} + \frac{\varepsilon}{2} - \frac{\varepsilon}{4} = \frac{1}{2} + \frac{\varepsilon'}{2}.$$

Thus, R will output "correlative" with probability at least $2/3$, by the correctness of R'. This completes the proof of the theorem. ◀

4.3 Learning from MK$^{\mathsf{t}}$P

The following lemma is similar to one from [16], but accounts for a uniformly random string $r \sim \mathcal{U}_{3mT(n)}$, which is essential given Lemma 29. This lemma states that in the expectation, over an efficiently samplable distribution (along with the uniformly random string r), the K^t-complexity of a string is close to its time-unbounded K-complexity. Note that the lemma from [16] holds under the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ whereas this one holds unconditionally.

▶ **Lemma 37.** *There exists a polynomial $p_1 \colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ such that for any $T, a \colon \mathbb{N} \to \mathbb{N}$ and $n, m \in \mathbb{N}$, the following holds unconditionally. Let $D_n \in \mathsf{Samp}[T(n)]/a(n)$. For every $t \geq p_1(T(n), m)$, $X \sim D_n^m$, and $r \sim \mathcal{U}_{3mT(n)}$,*

$$\mathop{\mathbb{E}}_{X,r}[\mathsf{K}^t(X, r) - \mathsf{K}(X, r)] \leq a(n) + O(\log m + \log T(n)).$$

Proof. Let p_1 be the polynomial p in Lemma 29. Note that for $D_n \in \mathsf{Samp}[T(n)]/a(n)$, we have $D_n^m \in \mathsf{Samp}[m \cdot T(n)]/a(n)$. For every $t \geq p_1(T(n), m)$, for $X \sim D_n^m$ and $r \sim \mathcal{U}_{3mT(n)}$,

$$\begin{aligned}
\mathop{\mathbb{E}}_{X,r}\left[\mathsf{K}^t(X, r)\right] &\leq \mathop{\mathbb{E}}_{X,r}\left[\mathsf{K}^{p_1(T(n),m)}(X, r)\right] \\
&\leq \frac{1}{4mT(n)} \cdot (mn + 3mT(n) + O(\log mn)) && \text{(Proposition 14)} \\
&\qquad + \mathop{\mathbb{E}}_{X}[\log(1/D_n^m(X))] + |r| + a(n) + O(\log(m) + \log T(n)) \\
& && \text{(Lemma 29)} \\
&\leq H(D_n^m) + |r| + a(n) + O(\log(m) + \log T(n)) \\
&\leq \mathop{\mathbb{E}}_{X,r}[\mathsf{K}(X) + \mathsf{K}(r \mid X)] + a(n) + O(\log(m) + \log T(n)) \\
&\leq \mathop{\mathbb{E}}_{X,r}[\mathsf{K}(X, r)] + a(n) + O(\log(m) + \log T(n)), && \text{(Time-unbounded S.o.I.)}
\end{aligned}$$

where the second last inequality uses the fact that for any distribution D, the Shannon entropy $H(D)$ is at most $\mathbb{E}[\mathsf{K}(x)]$ for $x \sim D$ (see [23, Theorem 8.1.1]), as well as a counting argument showing that $\mathbb{E}_{X,r}[\mathsf{K}(r \mid X)] \geq |r| - 3$.

Rearranging the above, we get

$$\mathop{\mathbb{E}}_{X,r}[\mathsf{K}^t(X, r) - \mathsf{K}(X, r)] \leq a(n) + O(\log m + \log T(n))$$

as desired. ◀

▶ **Theorem 38.** *If* $(\mathsf{MK^tP}, \mathcal{U}) \in \mathsf{AvgBPP}$*, then for any time-constructible functions* $s, T, a\colon \mathbb{N} \to \mathbb{N}$*, any* $\varepsilon \in (0,1)$*, and any constant* $\zeta > 0$*, there is an* ε*-correlative RRHS-refuter for* $\mathsf{SIZE}[s(n)]$ *under* $\mathsf{Samp}[T(n)]/a(n)$ *taking parameters* $n \in \mathbb{N}$ *and* $\varepsilon \in (0,1)$ *with sample complexity*

$$m := \left(\frac{s(n) + a(n) + \log T(n)}{\varepsilon^2}\right)^{1+\zeta}$$

and running time $\mathsf{poly}\big(n, \varepsilon^{-1}, T(n), a(n), s(n)\big)$.

Proof. The proof closely follows that of [16, Theorem 8].

The (correlative) RRHS-refuter R. On input $n \in \mathbb{N}$, $\varepsilon > 0$, and a set

$$S = \left(\left\langle x^{(1)}, b^{(1)}\right\rangle, \ldots, \left\langle x^{(m)}, b^{(m)}\right\rangle\right)$$

of samples, let $X := (x^{(1)}, \ldots, x^{(m)})$ and $b := (b^{(1)}, \ldots, b^{(m)})$. R is defined as follows.

1. Compute $t := p_1(T(n), m)$, where p_1 is the polynomial from Lemma 37. Also compute $t' := t'(p(t))$, where t' is the polynomial from Lemma 32 and p is the polynomial from Lemma 31.
2. Sample $r \sim \mathcal{U}_{3mT(n)}$.
3. Compute

$$\beta := A\left((X, r), 1^t\right) \text{ and}$$
$$\beta' := A\left((X, b, r), 1^{t'}\right),$$

   where A is the randomized algorithm from Lemma 31.
4. Output "correlative" if $\beta' - \beta \le \theta$, where $\theta = \left(1 - \frac{\varepsilon^2}{16}\right) m$, and output "random" otherwise.

We now argue for the correctness of R. Consider any distribution $D_n \in \mathsf{Samp}[T(n)]/a(n)$.

Correlative Case (Soundness). Suppose the samples S are i.i.d. from a distribution D' on $\{0,1\}^n \times \{0,1\}$ such that the marginal on $\{0,1\}^n$ equals D_n, and there exists $f \in \mathsf{SIZE}[s(n)]$ such that

$$\Pr_{\langle x^{(i)}, b^{(i)}\rangle \sim D'}\left[b^{(i)} = f(x^{(i)})\right] \ge \frac{1}{2} + \frac{\varepsilon}{2}.$$

Chernoff bounds imply that

$$\left|\{i \in [m] \mid b_i = f(x^{(i)})\}\right| \ge (1/2 + \varepsilon/4) \cdot m$$

holds with probability at least $1 - \exp(-2m(\varepsilon/4)^2)$ over the choice of samples $S \sim (D')^m$, in which case the conditions of Lemma 32 are met.

Now, suppose that in Step 3 of R, β and β' output by the algorithm A are good approximations in terms of Lemma 31, which happens with probability at least $1 - o(1)$. Moreover, by Lemma 37,

$$\begin{aligned}\mathop{\mathbb{E}}_{X,r}\left[\mathsf{K}^t(X,r) - \mathsf{pK}^{p(t)}(X,r)\right] &\le \mathop{\mathbb{E}}_{X,r}\left[\mathsf{K}^t(X,r) - \mathsf{K}(X,r)\right] && \text{(Prop. 13)}\\ &\le a(n) + O(\log(mT(n))).\end{aligned}$$

Applying Markov's inequality, with probability at least $3/4$, there is a constant c such that

$$\mathsf{K}^t(X,r) - \mathsf{pK}^{p(t)}(X,r) \le c \cdot (a(n) + \log(mT(n))). \tag{10}$$

Thus, by a union bound, with probability at least $3/4 - o(1) > 2/3$ over the samples $S \sim (D')^m$ and the internal randomness of R,

$$\begin{aligned}
\beta' - \beta &\le \mathsf{K}^{t'}(X,b,r) - \mathsf{pK}^{p(t)}(X,r) + \log p(t) && (\beta' \text{ and } \beta \text{ are good approximations}) \\
&\le \left(\mathsf{K}^t(X,r) - \mathsf{pK}^{p(t)}(X,r)\right) + \log p(t) + \ell_s(n) + \left(1 - \varepsilon^2/8\right) \cdot m && \text{(Lemma 32)} \\
&\le m \cdot \left(1 - \varepsilon^2/8\right) + c \cdot (a(n) + \log(mT(n))) + \ell_s(n) && \text{(Eq. (10))} \\
&< \theta.
\end{aligned}$$

For the last inequality, observe that by our choice of $m = \omega((s(n) + \log T(n) + a(n)) \cdot \varepsilon^{-2})$,

$$m > 16 \cdot \left(\frac{c \cdot (a(n) + \log m + \log T(n)) + \ell_s(n)}{\varepsilon^2}\right);$$

re-written,

$$\begin{aligned}
m \cdot \left(1 - \varepsilon^2/8\right) + c \cdot (a(n) + \log(mT(n))) + \ell_s(n) &< m \cdot \left(1 - \varepsilon^2/16\right) \\
&= \theta.
\end{aligned}$$

Thus, R will output "correlative".

Random Case (Completeness). Suppose the labels b_i are sampled from $\mathcal{U}$. For $X \sim D_n^m$, $r \sim \mathcal{U}_{3mT(n)}$, and $b \sim \mathcal{U}^m$, we get by Lemma 37 and Markov's inequality, that, with probability at least $3/4$ over X, r,

$$\mathsf{K}^t(X,r) - K(X,r) \le 4(a(n) + O(\log mT(n))). \tag{11}$$

Since β' and β are good estimates with high probability, we get that, with probability at least $3/4 - o(1)$ over X, r, b and the internal randomness of A,

$$\begin{aligned}
\beta' - \beta &\ge \mathsf{pK}^{p(t')}(X,b,r) - \mathsf{K}^t(X,r) - O(\log(mT(n))) && (\beta', \beta \text{ good w.h.p.}) \\
&\ge \mathsf{K}(X,b,r) - \mathsf{K}^t(X,r) - O(\log(mT(n))) && \text{(Prop. 13)} \\
&\ge \mathsf{K}(X,r) + \mathsf{K}(b \mid X,r) - \mathsf{K}^t(X,r) - O(\log(mT(n))) && \text{(Lemma 15)} \\
&= m - \left(\mathsf{K}^t(X,r) - \mathsf{K}(X,r)\right) - O(\log(mT(n))) && (b \sim \mathcal{U}^m) \\
&\ge m - 4\,(a(n) + O(\log(mT(n)))) && \text{(Eq. (11))} \\
&> \theta,
\end{aligned}$$

and hence R outputs "random". ◀

4.4 Learning from Worst-case Easiness of MKTP

In this section, we show that if MKTP is easy for efficient randomized algorithms in the *worst case*, then it is possible to PAC learn *without* white-box access to the target distribution.

The following lemma is analogous to the source-coding lemma for K^t, Lemma 29, but with some modifications to allow for KT-compression in the case that we have many independent samples from the distribution D_n.

▶ **Lemma 39.** *For some constant $d \in \mathbb{N}$, the following holds unconditionally. For any $T, a : \mathbb{N} \to \mathbb{N}$, $m, n \in \mathbb{N}$, distribution $D \in \mathsf{Samp}[T(n)]/a(n)$, and string $X = (x^{(1)}, ..., x^{(m)}) \in \mathsf{Supp}(D_n^m)$,*

$$\mathsf{KT}(X, r) \leq \log(1/D_n^m(X)) + |r| + a(n) + d \cdot m^{3/4} \cdot T(n)^3$$

holds with probability at least $1 - \frac{1}{6mT(n)}$ over $r \sim \mathcal{U}_{4mT(n)}$.

Moreover, for any $s : \mathbb{N} \to \mathbb{N}$ and function $f \in \mathsf{SIZE}[s(n)]$,

$$\mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(m)}), r) \leq \log(1/D_n^m(X)) + 2\ell_s(n) + |r| + a(n) + d \cdot m^{3/4} \cdot T(n)^3$$

holds with probability at least $1 - \frac{1}{6mT(n)}$ over $r \sim \mathcal{U}_{4mT(n)}$.

Proof. The proof is quite similar to that of Lemma 29, with some modifications (namely, the partitioning of $[m]$) to get the bound for KT-complexity. Let A be a non-uniform algorithm sampling $D \in \mathsf{Samp}[T(n)]/a(n)$. That is, there is some $\alpha \in \{0,1\}^{a(n)}$ such that for any $x \in \mathrm{supp}(D_n)$,

$$\Pr_{w \sim \mathcal{U}_{T(n)}}[A(w; \alpha, 1^n)] = D_n(x).$$

Consider any $X = (x^{(1)}, ..., x^{(m)}) \in \mathsf{Supp}(D_n^m)$. For N and L chosen later, we will partition $[m]$ into N blocks $b_1, ..., b_N$, each of size at most L. For every block b_j, let s_j be the largest integer such that $D_n^L(x^{(j_1)}, ..., x^{(j_L)}) \leq 2^{-s_j}$, where $b_j = \{j_1, ..., j_L\}$, and let $s = \sum_{j \in [N]} s_j$. For each b_j, consider a universal hash function family $\mathcal{H}_j = \{h : \{0,1\}^{L \cdot T(n)} \to \{0,1\}^{k_j}\}$, where $k_j = L \cdot T(n) - s_j - \log(12m^2T(n)) - 1$. As in Lemma 29, we represent hash functions with Toeplitz matrices.

For each block b_j, define a set

$$S_j := \{(w_1, ..., w_L) \in (\{0,1\}^{T(n)})^L \mid \forall l \in [L],\ A(w_l; \alpha, 1^n) = x^{(j_l)}\},$$

where j_l denotes the l^{th} element of b_j. For each $j \in [N]$, define a random variable $X_j := |S_j \cap h^{-1}(0^{k_j})|$, where $h \sim \mathcal{H}_j$. Arguing as in Lemma 29,

$$\Pr[X_j = 0] \leq \frac{1}{12m^2T(n)}.$$

Now, for each $j \in [N]$, define a random variable $Y_j = |h^{-1}(0^{k_j})|$, where $h \sim \mathcal{H}_j$. Arguing as in Lemma 29,

$$\Pr[Y_j \geq 2^{s_j + 2\log(12m^2T(n)) + 1}] \leq \frac{1}{12m^2T(n)}.$$

By a union bound, with probability at least $1 - 1/6mT(n)$, we have that for every $j \in [N]$, $X_j \neq 0$ and $Y_j < 2^{s_j + 2\log(12m^2T(n)) + 1}$.

Assume the above holds. It is possible to obtain any bit of a substring $x^{(i)}$ of X, for i in some block b_j, from the description of the hash function h sampled from $\mathcal{H}_j$, the index of a string $(w_1, ..., w_L) \in S_j$ in the set $h^{-1}(0^{k_j})$, and the advice string α used in the sampler A. In particular, $x^{(i)}$ may be recovered by performing Gaussian elimination to compute the set $h^{-1}(0^{k_j})$ from the description of h, locating w_l in this set such that i is the l^{th} element of b_j, and then returning the desired bit of $A(w_l; \alpha, 1^n) = x^{(i)}$. Given h, this requires at most $\log Y_j \leq s_j + 2\log(12m^2T(n)) + 1$ bits to describe the position of $(w_1, ..., w_L)$ in $h^{-1}(0^{k_j})$ and $|\alpha| = a(n)$ bits to run the sampler A. Define the "random" string $r_j \in \{0,1\}^{3L \cdot T(n)}$ as the description of $h \sim \mathcal{H}_j$. So, a description working for *any* block (and therefore any bit of X) is of length

$$\sum_{j\in[N]} \left[s_j + 2\log(12m^2T(n)) + 1\right] + a(n) \le s + N \cdot (2\log(12m^2T(n)) + 1) + a(n)$$

given randomness $r = (r_1, ..., r_N)$ of length $3L \cdot N \cdot T(n)$. The amount of time required is dominated by the Gaussian elimination step, at most $O((L \cdot T(n))^3)$.

To obtain some bit $f(x^{(i)})$, one may apply the above procedure to obtain $x^{(i)}$ and then apply an $\ell_s(n)$-bit description of a circuit computing f, taking additional time at most $\ell_s(n)$.

Overall, with probability at least $1 - 1/6mT(n)$ over r, we have that

$$\mathsf{KT}(X, r) \le s + |r| + a(n) + N \cdot O(\log(mT(n))) + O((L \cdot T(n))^3)$$

and

$$\mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(m)}), r) \le s + |r| + 2\ell_s(n) + a(n) + N \cdot O(\log(mT(n))) + O((L \cdot T(n))^3).$$

The lemma follows by setting $L = m^{1/4}$ and $N = \lceil m^{3/4} \rceil$. ◀

The following lemma is analogous to Lemma 37, showing that KT and K complexities are somewhat close in the expectation over efficiently sampled strings.

▶ **Lemma 40.** *For any $T, a \colon \mathbb{N} \to \mathbb{N}$, $n, m \in \mathbb{N}$, $D_n \in \mathsf{Samp}[T(n)]/a(n)$, $X \sim D_n^m$, $b \sim \mathcal{U}_m$, and $r \sim \mathcal{U}_{4mT(n)}$,*

$$\mathop{\mathbb{E}}_{X,b,r}[\mathsf{KT}(X, b, r) - \mathsf{K}(X, b, r)] \le a(n) + 2d \cdot m^{3/4} \cdot T(n)^3,$$

where d is the constant from Lemma 39.

Moreover, for any function $f \in \mathsf{SIZE}[s(n)]$,

$$\mathop{\mathbb{E}}_{X,r}[\mathsf{KT}(X, f(x^{(1)}), ..., f(x^{(1)}), r) - \mathsf{K}(X, f(x^{(1)}), ..., f(x^{(1)}), r)] \le a(n) + 2\ell_s(n) + 2d \cdot m^{3/4} \cdot T(n)^3.$$

Proof. The proof closely follows that of Lemma 37.

$$\begin{aligned}
\mathop{\mathbb{E}}_{X,b,r}[\mathsf{KT}(X, b, r)] &\le \mathop{\mathbb{E}}_{X,r}[\mathsf{KT}(X, r)] + |b| + \log m \\
&\le \frac{1}{6mT(n)} \cdot (mn + 4mT(n) + m + O(\log mn)) \\
&\quad + \mathop{\mathbb{E}}_{X}[\log(1/D_n^m(X))] + |r| + a(n) + d \cdot m^{3/4} \cdot T(n)^3 + |b| + \log m \qquad \text{(Lemma 39)} \\
&\le H(D_n^m) + |r| + a(n) + d \cdot m^{3/4} \cdot T(n)^3 + |b| + O(\log m) \\
&\le \mathop{\mathbb{E}}_{X,b,r}[\mathsf{K}(X) + \mathsf{K}(b \mid X) + \mathsf{K}(r \mid b, X)] + a(n) + d \cdot m^{3/4} \cdot T(n)^3 + O(\log m) \\
&\le \mathop{\mathbb{E}}_{X,b,r}[\mathsf{K}(X, b, r)] + a(n) + 2d \cdot m^{3/4} \cdot T(n)^3. \qquad \text{(Time-unbounded S.o.I.)}
\end{aligned}$$

Rearranging the above, we get

$$\mathop{\mathbb{E}}_{X,b,r}[\mathsf{KT}(X, b, r) - \mathsf{K}(X, b, r)] \le a(n) + 2d \cdot m^{3/4} \cdot T(n)^3$$

as desired.

The proof of the "moreover" part of the lemma is very similar. It follows by applying the "moreover" part of Lemma 39 in the second line, and in the last line using the simple fact that $\mathsf{K}(X, r) \le \mathsf{K}(X, f(x^{(1)}), ..., f(x^{(1)}), r)$. ◀

▶ **Theorem 41.** *If* $\mathsf{MKTP} \in \mathsf{BPP}$*, then for any time-constructible functions* $s, T, a\colon \mathbb{N} \to \mathbb{N}$*, and any* $\varepsilon \in (0,1)$*, there is an* ε*-correlative RRHS-refuter for* $\mathsf{SIZE}[s(n)]$ *under* $\mathsf{Samp}[T(n)]/a(n)$ *taking parameters* $n \in \mathbb{N}$ *and* $\varepsilon \in (0,1)$ *with sample complexity*

$$m := \left(\frac{s(n) + a(n) + T(n)^{12}}{\varepsilon^8}\right)^{12}$$

and running time $\mathsf{poly}\big(n, \varepsilon^{-1}, T(n), a(n), s(n)\big)$.

Proof. Let A_0 be the assumed randomized algorithm for MKTP, and let A be the randomized poly-time "search" algorithm that on input y runs $A_0(y, 1^s)$ for $s = 1, ..., |y| + \log|y|$ and outputs the smallest s on which A accepts. It is not hard to see, using standard techniques, that A can be made to correctly compute $\mathsf{KT}(y)$ with probability $1 - 2^{-|y|}$.

The (correlative) RRHS-refuter R. On input $n \in \mathbb{N}$, $\varepsilon > 0$, and a set

$$S = \left(\left\langle x^{(1)}, b^{(1)}\right\rangle, \ldots, \left\langle x^{(m)}, b^{(m)}\right\rangle\right)$$

of samples, let

$$k := \left(\frac{s(n) + a(n) + T(n)^{12}}{\varepsilon^8}\right)^2.$$

Partition the $m = k^6$ samples S into k^5 sets, each containing k samples. Denote these sets S_i, for $i \in [k]$. Then partition each S_i into two equally sized sets,

$$S_i^0 = \left(\left\langle x_i^{(1)}, b_i^{(1)}\right\rangle, \ldots, \left\langle x_i^{(k/2)}, b_i^{(k/2)}\right\rangle\right) \text{ and } S_i^1 = \left(\left\langle x_i^{(k/2+1)}, b_i^{(k/2+1)}\right\rangle, \ldots, \left\langle x_i^{(k)}, b_i^{(k)}\right\rangle\right).$$

Let $Z_i := (x_i^{(1)}, ..., x_i^{(k/2)})$, $X_i := (x_i^{(k/2+1)}, \ldots, x_i^{(k)})$ and $b_i := (b_i^{(k/2+1)}, \ldots, b_i^{(k)})$.

R is defined as follows. We repeat the following k^5 times: once on each set of samples S_i. For simplicity, we omit the subscripts i: denote $(\langle x^{(1)}, b^{(1)}\rangle, \ldots, \langle x^{(k)}, b^{(k)}\rangle) := S_i$, $Z := Z_i$, $X := X_i$, and $b := b_i$.

1. Sample $r \sim \mathcal{U}_{2kT(n)}$.
2. Sample $u \sim \mathcal{U}_{k/2}$, and using the first half of the samples Z, compute

   $$\gamma_i := A(Z, u, r).$$

3. Using the second half of the samples X along with their given labels b, compute

   $$\beta_i := A\left(X, b, r\right).$$

4. Let $w_i = \gamma_i - \beta_i$.
5. At the end, after k^5 repetitions of the above, take the sum

   $$w = \sum_{i \in [k^5]} w_i.$$

   Let d be the constant from Lemma 39. Output "correlative" if $w \geq k^5 \cdot \theta$, where $\theta = 2 \cdot \big(a(n) + 4d \cdot k^{3/4} \cdot T(n)^3\big)$, and output "random" otherwise.

We begin by showing that the expected value of γ_i is roughly $H(D_n^{k/2}) + k/2 + |r|$. On one hand, we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{Z,u,r,A}[\gamma_i] &\leq \mathop{\mathbb{E}}_{Z,u,r}[\mathsf{KT}(Z,u,r)] + O(1) && \text{(definition of } A\text{)} \\
&\leq \mathbb{E}[\mathsf{K}(Z,u,r)] + a(n) + 2d \cdot k^{3/4} \cdot T(n)^3 + O(1) && \text{(Lemma 40)} \\
&\leq \Big(H(D_n^{k/2}) + k/2 + |r|\Big) + a(n) + 3d \cdot k^{3/4} \cdot T(n)^3, && (12)
\end{aligned}$$

where the last line follows by a counting argument and the fact that $\mathbb{E}[\mathsf{K}(Z)] \leq H(D_n^{k/2})$.

On the other hand,

$$\begin{aligned}
\mathop{\mathbb{E}}_{Z,u,r,A}[\gamma_i] &\geq \mathop{\mathbb{E}}_{Z,u,r}[\mathsf{KT}(Z,u,r)] - O(1) && \text{(definition of } A\text{)} \\
&\geq \mathbb{E}[\mathsf{K}(Z,u,r)] - O(1) \\
&\geq \mathbb{E}[\mathsf{K}(Z) + \mathsf{K}(u \mid Z) + \mathsf{K}(r \mid Z,u)] - O(\log(kn)) && \text{(symmetry of information)} \\
&\geq \Big(H(D_n^{k/2}) + k/2 + |r|\Big) - O(\log(kn)), && (13)
\end{aligned}$$

where in the last line we use that $\mathbb{E}[\mathsf{K}(Z)] \geq H(D_n^{k/2}) - O(\log(kn))$ [23, Theorem 8.1.1].

Correlative Case (Soundness). Suppose the samples S are i.i.d. from a distribution D' on $\{0,1\}^n \times \{0,1\}$ such that the marginal on $\{0,1\}^n$ equals D_n, and there exists $f \in \mathsf{SIZE}[s(n)]$ such that

$$\Pr_{\langle x^{(j)}, b^{(j)} \rangle \sim D'}\left[b^{(j)} = f(x^{(j)})\right] \geq \frac{1}{2} + \frac{\varepsilon}{2}.$$

Chernoff bounds imply that

$$\left|\{j \in \{k/2+1, ..., k\} \mid b_i = f(x^{(j)})\}\right| \geq (1/2 + \varepsilon/4) \cdot k/2$$

holds with probability at least $1 - \exp(-k(\varepsilon/4)^2/8)$ over the choice of samples S_i^1, in which case the conditions of Lemma 34 are met. Then,

$$\begin{aligned}
\mathop{\mathbb{E}}_{X,b,r,A}[\beta_i] &\leq \mathop{\mathbb{E}}_{X,b,r}[\mathsf{KT}(X,b,r)] + O(1) \\
&\leq \mathbb{E}[\mathsf{KT}(X, f(x^{(k/2+1)}), ..., f(x^{(k)}), r)] + (1 - \varepsilon^2/8) \cdot k/2 + 2k^{3/4} && \text{(Lemma 34)} \\
&\leq \mathbb{E}[\mathsf{K}(X, f(x^{(k/2+1)}), ..., f(x^{(k)}), r)] + (1 - \varepsilon^2/8) \cdot k/2 + a(n) + 2\ell_s(n) + 3d \cdot k^{3/4} \cdot T(n)^3 \\
&\leq H(D_n^{k/2}) + (1 - \varepsilon^2/8) \cdot k/2 + |r| + a(n) + 3\ell_s(n) + 3d \cdot k^{3/4} \cdot T(n)^3,
\end{aligned}$$

where the second-last line uses Lemma 40, and the last line uses the observation that

$$\mathsf{K}(f(x^{(k/2+1)}), ..., f(x^{(k)}) \mid X) \leq \ell_s(n).$$

Combining the above with Eq. (13), we have

$$\begin{aligned}
\mathbb{E}[\gamma_i - \beta_i] &\geq \frac{\varepsilon^2}{16} \cdot k - \Big(a(n) + 3\ell_s(n) + O(k^{3/4} T(n)^3)\Big) \\
&\geq 2\theta,
\end{aligned}$$

by our choices of k and θ.

After k^5 trials of the above, we have $\mathbb{E}[w] \geq 2k^5\theta$. By Hoeffding's inequality, with probability at least $1 - 2^{-k}$, it holds that $|2k^5\theta - w| \leq k^5\theta$, and so

$$w \geq k^5\theta,$$

in which case R will output "correlative".

Random Case (Completeness). Suppose the labels b_j are sampled from $\mathcal{U}$. Arguing as in Eq. (13),

$$\mathop{\mathbb{E}}_{X,b,r,A}[\beta_i] \geq \left(H(D_n^{k/2}) + k/2 + |r|\right) - O(\log(kn)).$$

Combining the above with Eq. (12),

$$\begin{aligned}\mathbb{E}[\gamma_i - \beta_i] &\leq a(n) + 4d \cdot k^{3/4} \cdot T(n)^3 \\ &= \theta/2.\end{aligned}$$

After k^5 trials of the above, we have $\mathbb{E}[w] \leq k^5\theta/2$. By Hoeffding's inequality, with probability at least $1 - 2^{-k}$, it holds that $|k^5\theta/2 - w| < k^5\theta/2$, and so

$$w < k^5\theta,$$

in which case R will output "random". ◀

5 Open questions

We showed that "natural propeties" for more expressive Kolmogorov-complexity relatives of MCSP such as MKTP and $\mathsf{MK^tP}$ allow one to cross the divide between learning algorithms with membership queries and those without. An obvious disadvantage of relying on more expressive Kolmogorov measures rather than MCSP is that it is difficult to get meaningful circuit class restrictions when talking about MKTP or $\mathsf{MK^tP}$, and utilize the known circuit lower bound proofs for these restricted circuit classes in order to derive a learning algorithm. Can one use our understanding of $\mathsf{AC}^0[2]$ circuit lower bounds (e.g., the known natural property for $\mathsf{AC}^0[2]$) to get an RRHS-refuter for $\mathsf{AC}^0[2]$ on uniform distribution? This question is also very interesting from the point of view of cryptography in the context of efficient constructions of weak PRFs; see, e.g., [8] for more discussion on this direction.

Another question is whether it is possible to bridge the gap between the assumptions used in our two main theorems. More precisely, is it possible to get an agnostic PAC learning algorithm over any *not necessarily explicitly given* polysize samplable distribution ensemble D from a one-sided average-case heuristic for MKTP rather than $\mathsf{MK^tP}$?

References

1 Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006. doi:10.1137/050628994.

2 Eric Allender, Joshua A. Grochow, Dieter van Melkebeek, Cristopher Moore, and Andrew Morgan. Minimum circuit size, graph isomorphism, and related problems. *SIAM J. Comput.*, 47(4):1339–1372, 2018. doi:10.1137/17M1157970.

3 Luis Filipe Coelho Antunes and Lance Fortnow. Worst-case running times for average-case algorithms. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009, Paris, France, 15-18 July 2009*, pages 298–303. IEEE Computer Society, 2009. doi:10.1109/CCC.2009.12.

4 Eric Binnendyk, Marco Carmosino, Antonina Kolokolova, R. Ramyaa, and Manuel Sabin. Learning with distributional inverters. In Sanjoy Dasgupta and Nika Haghtalab, editors, *International Conference on Algorithmic Learning Theory, 29-1 April 2022, Paris, France*, volume 167 of *Proceedings of Machine Learning Research*, pages 90–106. PMLR, 2022. URL: https://proceedings.mlr.press/v167/binnendyk22a.html.

5 Avrim Blum, Merrick L. Furst, Michael J. Kearns, and Richard J. Lipton. Cryptographic primitives based on hard learning problems. In Douglas R. Stinson, editor, *Advances in Cryptology – CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings*, volume 773 of *Lecture Notes in Computer Science*, pages 278–291. Springer, 1993. doi:10.1007/3-540-48329-2_24.

6 Andrej Bogdanov and Alon Rosen. Pseudorandom functions: Three decades later. In Yehuda Lindell, editor, *Tutorials on the Foundations of Cryptography*, pages 79–158. Springer International Publishing, 2017. doi:10.1007/978-3-319-57048-8_3.

7 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1), 2006. doi:10.1561/0400000004.

8 Dan Boneh, Yuval Ishai, Alain Passelègue, Amit Sahai, and David J. Wu. Exploring crypto dark matter: New simple PRF candidates and their applications. In Amos Beimel and Stefan Dziembowski, editors, *Theory of Cryptography – 16th International Conference, TCC 2018, Panaji, India, November 11-14, 2018, Proceedings, Part II*, volume 11240 of *Lecture Notes in Computer Science*, pages 699–729. Springer, 2018. doi:10.1007/978-3-030-03810-6_25.

9 Harry Buhrman, Lance Fortnow, and Aduri Pavan. Some results on derandomization. *Theory Comput. Syst.*, 38(2):211–227, 2005. doi:10.1007/s00224-004-1194-y.

10 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning algorithms from natural proofs. In Ran Raz, editor, *31st Conference on Computational Complexity, CCC 2016, May 29 to June 1, 2016, Tokyo, Japan*, volume 50 of *LIPIcs*, pages 10:1–10:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.CCC.2016.10.

11 Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Agnostic learning from tolerant natural proofs. In Klaus Jansen, José D. P. Rolim, David Williamson, and Santosh S. Vempala, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2017, August 16-18, 2017, Berkeley, CA, USA*, volume 81 of *LIPIcs*, pages 35:1–35:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.APPROX-RANDOM.2017.35.

12 Vitaly Feldman. Distribution-specific agnostic boosting. In Andrew Chi-Chih Yao, editor, *Innovations in Computer Science – ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 241–250. Tsinghua University Press, 2010. URL: http://conference.iiis.tsinghua.edu.cn/ICS2010/content/papers/20.html.

13 Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor Carboni Oliveira. Probabilistic kolmogorov complexity with applications to average-case complexity. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 16:1–16:60. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CCC.2022.16.

14 Alexander Golovnev, Rahul Ilango, Russell Impagliazzo, Valentine Kabanets, Antonina Kolokolova, and Avishay Tal. $AC^0[p]$ lower bounds against MCSP via the coin problem. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 66:1–66:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.66.

15 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 247–258. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00032.

16 Shuichi Hirahara and Mikito Nanashima. On worst-case learning in relativized heuristica. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 751–758. IEEE, 2021. doi:10.1109/FOCS52979.2021.00078.

17 Rahul Ilango, Bruno Loff, and Igor Carboni Oliveira. Np-hardness of circuit minimization for multi-output functions. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 22:1–22:36. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.22.

18 Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 812–821. IEEE Computer Society, 1990. doi:10.1109/FSCS.1990.89604.

19 Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *30th Annual Symposium on Foundations of Computer Science, Research Triangle Park, North Carolina, USA, 30 October – 1 November 1989*, pages 230–235. IEEE Computer Society, 1989. doi:10.1109/SFCS.1989.63483.

20 Adam Kalai and Varun Kanade. Potential-based agnostic boosting. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems 2009. Proceedings of a meeting held 7-10 December 2009, Vancouver, British Columbia, Canada*, pages 880–888. Curran Associates, Inc., 2009. URL: https://proceedings.neurips.cc/paper/2009/hash/13f9896df61279c928f19721878fac41-Abstract.html.

21 Michael J. Kearns, Robert E. Schapire, and Linda Sellie. Toward efficient agnostic learning. *Mach. Learn.*, 17(2-3):115–141, 1994. doi:10.1007/BF00993468.

22 Pravesh K. Kothari and Roi Livni. Improper learning by refuting. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, volume 94 of *LIPIcs*, pages 55:1–55:10. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.ITCS.2018.55.

23 Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, 4th Edition*. Texts in Computer Science. Springer, 2019. doi:10.1007/978-3-030-11298-1.

24 Moni Naor and Eylon Yogev. Bloom filters in adversarial environments. *ACM Trans. Algorithms*, 15(3):35:1–35:30, 2019. doi:10.1145/3306193.

25 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

26 Rafail Ostrovsky and Avi Wigderson. One-way fuctions are essential for non-trivial zero-knowledge. In *Second Israel Symposium on Theory of Computing Systems, ISTCS 1993, Natanya, Israel, June 7-9, 1993, Proceedings*, pages 3–17. IEEE Computer Society, 1993. doi:10.1109/ISTCS.1993.253489.

27 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.

28 Salil P. Vadhan. On learning vs. refutation. In Satyen Kale and Ohad Shamir, editors, *Proceedings of the 30th Conference on Learning Theory, COLT 2017, Amsterdam, The Netherlands, 7-10 July 2017*, volume 65 of *Proceedings of Machine Learning Research*, pages 1835–1848. PMLR, 2017. URL: http://proceedings.mlr.press/v65/vadhan17a.html.

29 Leslie G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, 1984. doi:10.1145/1968.1972.

30 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91. IEEE Computer Society, 1982. doi:10.1109/SFCS.1982.45.

31 Alexander K. Zvonkin and Leonid A. Levin. The complexity of finite objects and the development of the concepts of information and randomness by means of the theory of algorithms. *Russian Mathematical Surveys*, 25(6):83, 1970.

New Lower Bounds Against Homogeneous Non-Commutative Circuits

Prerona Chatterjee
Department of Computer Science, Tel Aviv University, Israel

Pavel Hrubeš
Institute of Mathematics, Czech Academy of Sciences, Prague, Czech Republic

Abstract

We give several new lower bounds on size of homogeneous non-commutative circuits. We present an explicit homogeneous bivariate polynomial of degree d which requires homogeneous non-commutative circuit of size $\Omega(d/\log d)$. For an n-variate polynomial with $n > 1$, the result can be improved to $\Omega(nd)$, if $d \leq n$, or $\Omega(nd\frac{\log n}{\log d})$, if $d \geq n$. Under the same assumptions, we also give a quadratic lower bound for the ordered version of the central symmetric polynomial.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases Algebraic circuit complexity, Non-Commutative Circuits, Homogeneous Computation, Lower bounds against algebraic circuits

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.13

Related Version *Full Version*: https://arxiv.org/abs/2301.01676

Funding *Prerona Chatterjee*: Funded by the Azrieli International Fellowship. This work was done while the author was a postdoctoral researcher at the Czech Academy of Sciences, Prague, and was funded by Czech Science Foundation GAČR grant 19-27871X.
Pavel Hrubeš: Czech Science Foundation GAČR grant 19-27871X.

Acknowledgements Prerona would like to acknowledge Cafedu for being such a nice place to work from. Pavel thanks Amir Yehudayoff for useful ideas on this topic which were exchanged in distant and joyous past.

1 Introduction

Arithmetic Circuit Complexity aims to categorize polynomials according to how hard they are to compute in algebraic models of computation. The most natural model is that of an arithmetic circuit: a directed acyclic graph with constant or variables as the leaf labels and addition or multiplication as labels of the internal nodes. Therefore, starting from variables or constants at the leaves, the every node in the circuit naturally computes new polynomials by means of addition and multiplication operations. The question is how many of these operations are needed.

The most challenging problem is to prove super-polynomial lower bounds against arithmetic circuits computing a low-degree polynomial. This is known as the VP vs VNP problem and is the algebraic analogue of the famed P vs. NP question. The classical result of Baur and Strassen [13, 1] gives an $\Omega(n \log d)$ lower bound for an n variate polynomial of degree d. A variety of lower bounds has since been obtained by imposing various restrictions on the computational model - e.g., arithmetic formulas[1] [8] or monotone circuits[2] [15]. But the result of Baur and Strassen remains the strongest lower bound on unrestricted arithmetic circuits.

[1] Similar to circuits except that the underlying graph is only allowed to be a tree instead of a DAG.
[2] Similar to circuits except that only non-negative constants are allowed.

In this paper, we are interested in the non-commutative setting where multiplication does not multiplicatively commute. Starting with the seminal works of Hyafil [7] and Nisan [9], non-commutative circuits are a well-studied object. The lack of commutativity is a severe limitation on the computational power which makes the task of proving circuit lower bounds seemingly easier. Nisan gave an exponential lower bound for non-commutative formulas whereas, commutatively, the best bound is only quadratic [8, 4]. Since then, it seemed that exponential non-commutative circuit lower bounds are just around the corner. Recently, Limaye, Srinivasan and Tavenas [14] proved such a lower bound in the *homogeneous, constant depth* setting for a polynomial that can be computed efficiently by non-commutative ABPs[3]. They showed that any constant depth Δ non-commutative homogeneous circuit for the *iterated matrix multiplication polynomial* (a polynomial over n variables of degree d must have size $n^{\Omega(d^{\frac{1}{\Delta}})}$. However for general circuits, even in the non-commutative setting, the strongest lower bound remains $\Omega(n \log d)$.

We improve this lower bound to $\Omega(nd/\log d)$ under the assumption that the non-commutative circuit is additionally homogeneous (see Section 2 for definition). Non-commutatively, this is already interesting if $n = 2$: we obtain a bivariate polynomial of degree d which requires circuit size nearly linear in d. It is well-known that a (commutative or not) circuit computing a homogeneous polynomial of degree d can be converted to an equivalent homogeneous circuit with at most a d^2 increase in size (see, for example, [6]). Hence, homogeneity is not a serious restriction if either d is small or if one proves a super-polynomial lower bound. However, our results fall in neither category and we do not know how to remove the homogeneity restriction. Furthermore, Carmosino et al. [3] have shown that strong enough superlinear lower bounds can be amplified to truly exponential ones. Unfortunately, the parameters of our result are not sufficient to allow amplification. Nevertheless, we strongly believe that it can be removed and that stronger non-commutative circuit lower bounds are just around the corner.

2 Notation and preliminaries

Let $\mathbb{F}$ be a field. A *non-commutative polynomial* over $\mathbb{F}$ is a formal sum of products of variables and field elements. We assume that the variables do not multiplicatively commute, whereas they commute additively and with elements of $\mathbb{F}$. The ring of non-commutative polynomials in variables $x_1, \ldots, x_n$ is denoted $\mathbb{F}\langle x_1, \ldots, x_n \rangle$. A polynomial is said to be *homogeneous* if all monomials with a non-zero coefficient in f have the same degree.

A *non-commutative arithmetic circuit* $\mathcal{C}$ over the field $\mathbb{F}$ is a directed acyclic graph as follows. Nodes (or gates) of in-degree zero are labelled by either a variable or an element in the field $\mathbb{F}$. All the other nodes have in-degree two and they are labelled by either $+$ or $\times$. The two edges going into a gate labelled by $\times$ are labelled by *left* and *right* to indicate the order of multiplication. Gates of in-degree zero will be called *input* gates; gates of out-degree zero will be called *output* gates.

Every node in $\mathcal{C}$ computes a non-commutative polynomial in the obvious way. We say that $\mathcal{C}$ computes a polynomial f if there is a gate in $\mathcal{C}$ computing f (not necessarily an output gate). $\mathcal{C}$ will be called *homogeneous* if every gate in $\mathcal{C}$ computes a homogeneous polynomial. Given a circuit $\mathcal{C}$, let $\widehat{\mathcal{C}} := (f \ : \ f$ is computed by some gate in $\mathcal{C})$.

[3] An algebraic computational model whose power lies in between that of circuits and formulas.

A product gate will be called *non-scalar*, if both of its inputs compute a non-constant polynomial. We define the *size* of $\mathcal{C}$ to be the number of non-input gates in it, and the *non-scalar size* of $\mathcal{C}$ to be the number of non-scalar product gates in it.

Given integers n_1, n_2, $[n_1, n_2]$ is the interval $\{n_1, n_1+1, \ldots, n_2\}$ and $[n] := [1, n]$.

Note. Unless stated otherwise, circuits and polynomials are assumed to be non-commutative and the underlying field $\mathbb{F}$ is fixed but arbitrary.

3 Main results

For univariate polynomials there is no difference between commutative and non-commutative computations. Already with two variables, non-commutative polynomials display much richer structure. There are 2^d monomials in variables x_0, x_1 of degree d (as opposed to $d+1$ in the commutative world); so a generic bivariate polynomial requires a circuit of size exponential in d.

Our first result is a lower bound that is almost linear in d. The hard polynomial is a bivariate monomial (a specific product of variables x_0, x_1).

▶ **Theorem 1.** *For every $d > 1$, there exists an explicit bivariate monomial of degree d such that any homogeneous non-commutative circuit computing it has non-scalar size $\Omega(d/\log d)$.*

In Remark 10, we point out a complementary $O(d/\log d)$ upper bound for every bivariate monomial. Note that commutatively every such monomial can be computed in size $O(\log d)$.

For n-variate polynomials, we obtain a stronger result (the hard polynomial is no longer a monomial).

▶ **Theorem 2.** *For every $n, d > 1$ there exists an explicit n-variate homogeneous polynomial of degree d which requires a homogenous non-commutative circuit of non-scalar size $\Omega(nd)$, if $d \le n$, or $\Omega(nd\frac{\log n}{\log d})$, if $d \ge n$.*

Theorem 1 and Theorem 2 are proved in Sections 4.1 and 4.2 respectively.

Given $0 \le d, n$, the *ordered symmetric polynomial*, OS_n^d, is the polynomial[4]

$$\mathrm{OS}_n^d(x_1, \ldots, x_n) = \sum_{1 \le i_1 < \cdots < i_d \le n} \left(\prod_{j=1}^{d} x_{i_j} \right).$$

It can be thought of as an ordered version of the commutative elementary symmetric polynomial. In Section 5, we shall prove a lower bound for this polynomial.

▶ **Theorem 3.** *If $2 \le d \le n/2$, any homogeneous non-commutative circuit computing $\mathrm{OS}_n^d(x_1, \ldots, x_n)$ must have non-scalar size $\Omega(dn)$.*

For the central ordered symmetric polynomial $\mathrm{OS}_n^{\lfloor n/2 \rfloor}$, the lower bound becomes $\Omega(n^2)$. We also observe that the known commutative upper bounds on elementary symmetric polynomials work non-commutatively as well.

▶ **Proposition 4.** *$\mathrm{OS}_n^1, \ldots, \mathrm{OS}_n^n$ can be simultaneously computed by a non-commutative circuit of size $O(n \log^2 n \log\log n)$, and by a homogeneous non-commutative circuit of size $O(n^2)$.*

[4] Hence $\mathrm{OS}_n^0 = 1$ and $\mathrm{OS}_n^d = 0$ whenever $d > n$.

The polylog factor in the proposition depends on the underlying field and can be improved for some $\mathbb{F}$s. Moreover, when measuring non-scalar size, one can obtain an $O(n \log n)$ upper bound if $\mathbb{F}$ is infinite – this is tight by [1].

The ordered symmetric polynomial can be contrasted with the truly symmetric polynomial

$$S_n^k = \sum_{1 i_1,\ldots,i_k \in [n] \text{ distinct}} x_{i_1} \cdots x_{i_k} ,$$

Non-commutatively, already S_n^n is as hard as the permanent [6] and is expected to require exponential circuits.

▶ Remark 5. A polynomial of degree d can be uniquely written as $f = \sum_{k=0}^{d} f^{(k)}$ where $f^{(k)}$ is homogeneous of degree k. It is well-known that if f has a circuit of size s, the homogeneous parts $f^{(0)}, \ldots, f^{(d)}$ can be simultaneously computed by a homogeneous circuit of size $O(sd^2)$ (this holds non-commutatively as well [6]). Note that $\mathrm{OS}_n^0, \ldots, \mathrm{OS}_n^n$ are the homogeneous parts of $\prod_{i=1}^{n}(1 + x_i)$ which has a circuit of a linear size. Theorem 3 shows that in this case, homogenization provably costs a factor of the degree.

4 Lower bounds against homogeneous non-commutative circuits

Let us define the measure we use to prove our lower bounds. Suppose $f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle$ is a homogeneous polynomial of degree d. Given an interval $J = [a, b] \subseteq [d]$, the polynomial f^J is obtained be setting variables in position *outside* of J to one. More precisely, if $\alpha = \prod_{i=1}^{d} x_{j_i}$ is a monomial then $\alpha^J := \prod_{i=a}^{b} x_{j_i}$, and the map is extended linearly so that $f^J = \sum_k c_k \alpha_k^J$ whenever $f = \sum_k c_k \alpha_k$. Given a non-negative integer ℓ, let

$$\boldsymbol{\mathcal{F}}_\ell(f) = \left(f^J \ : \ J \subseteq [d] \text{ is an interval of length } \ell \right) .$$

Given homogeneous polynomials $f_1, \ldots, f_m$, our hardness measure is defined as

$$\mu_\ell(f_1, \ldots, f_m) := \dim(\mathsf{span}(\bigcup_{i=1}^{m} \boldsymbol{\mathcal{F}}_\ell(f_i))) .$$

Here, $\mathsf{span}(\boldsymbol{\mathcal{F}})$ denotes the vector space of $\mathbb{F}$-linear combinations of polynomials in $\boldsymbol{\mathcal{F}}$ and dim is its dimension.

The following lemma bounds the measure in terms of circuit size.

▶ **Lemma 6.** *Let $\boldsymbol{\mathcal{C}}$ be a homogeneous circuit with s non-scalar multiplication gates. Then for every $\ell \geq 2$, $\mu_\ell(\widehat{\boldsymbol{\mathcal{C}}}) \leq (\ell - 1)s$.*

Proof. This is by induction on the size of $\boldsymbol{\mathcal{C}}$. If $\boldsymbol{\mathcal{C}}$ consists of input gates only then $\boldsymbol{\mathcal{F}}_\ell(\widehat{\boldsymbol{\mathcal{C}}}) = \emptyset$, as we assumed $\ell \geq 2$ and $\widehat{\boldsymbol{\mathcal{C}}}$ consists of linear polynomials.

Otherwise, assume that u is some output gate of $\boldsymbol{\mathcal{C}}$ and let $\boldsymbol{\mathcal{C}}'$ be the circuit obtained by removing that gate. If u is a sum gate or a scalar product gate then

$$\mu_\ell(\widehat{\boldsymbol{\mathcal{C}}}) \leq \mu_\ell(\widehat{\boldsymbol{\mathcal{C}}'}) .$$

For if u computes f then $f = a_1 f_1 + a_2 f_2$ for some constants a_1, a_2 and $f_1, f_2 \in \widehat{\boldsymbol{\mathcal{C}}'}$. If f has degree d then for every interval $J \subseteq [d]$ of length ℓ, $f^J = (a_1 f_1 + a_2 f_2)^J = a_1 f_1^J + a_2 f_2^J \in \mathsf{span}(\boldsymbol{\mathcal{F}}_\ell(\widehat{\boldsymbol{\mathcal{C}}'}))$.

If u is a non-scalar product gate computing $f = f_1 \cdot f_2$ then

$$\mu_\ell(\widehat{\boldsymbol{\mathcal{C}}}) \leq \mu_\ell(\widehat{\boldsymbol{\mathcal{C}}'}) + (\ell - 1) .$$

To see this assume f_1, f_2 have degrees d_1 and d_2 respectively, and let $J \subseteq [d_1 + d_2]$ be an interval of length ℓ. If J is contained in $[d_1]$, $f^J = (f_1 f_2)^J = f_1^J f_2^{\emptyset}$ is a scalar multiple of f_1^J and hence f^J is contained in $\mathsf{span}(\boldsymbol{\mathcal{F}}_\ell(\widehat{\boldsymbol{\mathcal{C}}'}))$; similarly if J is contained in $[d_1 + 1, d_2]$. Otherwise, both d_1 and $d_1 + 1$ are contained in J. But there are only $\ell - 1$ such intervals. Hence $\boldsymbol{\mathcal{F}}_\ell(\widehat{\boldsymbol{\mathcal{C}}})$ contains at most $\ell - 1$ polynomials outside of $\mathsf{span}(\boldsymbol{\mathcal{F}}_\ell(\widehat{\boldsymbol{\mathcal{C}}'}))$.

This means that μ_ℓ increases only at product gates, and that it increases only by $\ell - 1$ at such gates. Hence $\mu_\ell(\widehat{\boldsymbol{\mathcal{C}}}) \leq (\ell - 1)s$. ◀

▶ Remark 7. If f has n variables and degree d, the measure $\mu_\ell(f)$ can be at most the minimum of $d - (\ell - 1)$ and n^ℓ. Hence, Lemma 6 can by itself give a lower of at most the order of $d \log n / \log d$.

4.1 Lower bounds for a single monomial

Interestingly, Lemma 6 gives non-trivial lower bounds for f being merely a product of variables (that is, monomials), namely lower bounds of the form $\tilde{\Omega}(d)$ for a monomial of degree d. The simplest example is for an n-variate monomial of degree n^2.

▶ **Proposition 8.** *Every homogeneous circuit computing $f = \prod_{i=1}^n \prod_{j=1}^n (x_i x_j)$ contains at least n^2 non-scalar product gates.*

Proof. This is an application of Lemma 6 with $\ell = 2$. The family $\boldsymbol{\mathcal{F}}_2(f)$ consists of all monomials $x_i x_j$. Hence, $\mu_2(f) = n^2$. If $\boldsymbol{\mathcal{C}}$ computes f, we have $\mu_2(\widehat{\boldsymbol{\mathcal{C}}}) \geq \mu_2(f)$ and hence $\boldsymbol{\mathcal{C}}$ contains at least n^2 product gates. ◀

Another case of interest is a monomial in two variables, x_0, x_1, of degree d. Suppose $f = \prod_{i=1}^d x_{\sigma_i}$ where $\sigma = (\sigma_1, \ldots, \sigma_d) \in \{0,1\}^d$. Then $\mu_\ell(f)$ equals the number of distinct substrings of σ of length ℓ. Hence we want to find a σ which contains as many substrings as possible. One construction of such an object is provided by the *de Bruijn sequence* [5].

de Bruijn sequences

For a given k, a de Bruijn sequence of order k over alphabet A is a cyclic sequence σ in which every k-length string from A^k occurs exactly once as a substring. Note that σ must have length $|A|^k$. Furthermore, precisely $k - 1$ of the substrings overlap the beginning and the end of the sequence and σ contains $|A|^k - (k - 1)$ substrings when viewed as an ordinary sequence. de Bruijn sequences are widely studied and, in particular, they exist. Moreover, efficient algorithms are known for constructing de Bruijn sequences (see, for example, [11] and its references). In the case of binary alphabet $A = \{0, 1\}$, this is especially so. We can start with a string of k zeros. At each stage, extend the sequence by 1, unless this results in a k-string already encounters, otherwise extend by 0.

Given $d \geq 2$, let σ be a binary de Bruijn sequence of order $\lceil \log_2 d \rceil$. It has length $2^{\lceil \log_2 d \rceil} \geq d$. Define the polynomial

$$\mathsf{B}_d(x_0, x_1) := \prod_{i=1}^{d} x_{\sigma_i} .$$

The following implies the result of Theorem 1.

▶ **Proposition 9.** *Every homogeneous circuit computing B_d contains $\Omega(d/\log d)$ non-scalar product gates.*

Proof. This is an application of Lemma 6 with $\ell = \lceil \log_2 d \rceil$. $[d]$ contains $d - \ell - 1$ intervals of length ℓ, all of which give rise to different substrings of σ. The family $\boldsymbol{\mathcal{F}}_\ell(\mathsf{B}_d)$ consists of $d - (\ell - 1)$ different monomials and hence $\mu_\ell(\mathsf{B}_d) = d - (\ell - 1)$. By the lemma, assuming $\ell > 1$, a homogenous circuit for B_d must contain $(d - (\ell - 1))/(\ell - 1) = \Omega(d/\log d)$ product gates. ◀

▶ Remark 10. Using de Bruijn sequences over alphabet of size n, one can give an explicit monomial in $n > 1$ variables and degree $d \geq n$ which requires homogeneous circuit of non-scalar size $\Omega(d \log n / \log d)$. This can also be deduced from Proposition 9 by viewing degree k bivariate monomials as a single variable.

Conversely, every such monomial α can be computed in size $O(d \log n / \log d)$ using multiplication gates only (such a computation is automatically homogeneous). Indeed, we can first compute all monomials of degree at most k by a circuit of size $O(n^{k+1})$ and then compute α using $\lceil d/k \rceil$ additional multiplication gates. Choosing k around $0.5 \log_2 d \log_2^{-1} n$ is sufficient. This also means that the bound in Theorem 2 is tight.

4.2 Computing partial derivatives simultaneously

In order to obtain stronger lower bounds, we will translate the classical theorem of Baur and Strassen [1] on computing partial derivatives to the non-commutative setting.

We define partial derivative with respect to first position only, as follows. Given a polynomial f and a variable x, f can be uniquely written as $f = xf_0 + f_1$ where no monomial in f_1 contains x in the first position. We set $\partial_x f := f_0$.

The proof of the following lemma is almost the same as the one due to Baur and Strassen. The only additional subtlety is that we need the derivatives to be computed by a homogeneous circuit. This requires the generalization of homogeneity to allow arbitrary variable weights. We emphasize that taking derivatives with respect to the first position is essential in the non-commutative setting.

▶ **Lemma 11.** *Assume that $f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle$ can be computed by a homogeneous circuit of size s and non-scalar size $s_\times$. Then $\partial_{x_1} f, \ldots, \partial_{x_n} f$ can be simultaneously computed by a homogeneous circuit of size $O(s)$ and non-scalar size $O(s_\times)$.*

Proof. Given $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{N}^n$, let w_i be the *weight* of x_i and let the weight of a monomial $\alpha = \prod_{j=1}^{d} x_{i_j}$ be defined as $\mathsf{wt}(\alpha) = \sum_{j=1}^{d} w_{i_j}$. A polynomial $f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle$ is said to be $\mathbf{w}$-homogeneous if every monomial in it has the same weight. We call this the weight of f, denoted by $\mathsf{wt}(f)$. Furthermore we say that a circuit $\boldsymbol{\mathcal{C}}$ is $\mathbf{w}$-homogeneous if every gate in it computes a $\mathbf{w}$-homogeneous polynomial. The weight of any node, v, in a $\mathbf{w}$-homogeneous circuit is defined to be the weight of the polynomial being computed by it.

Note that if $(w_1, \ldots, w_n) = (1, \ldots, 1)$, then $\mathbf{w}$-homogeneity coincides with the usual notion of homogeneity. Therefore Lemma 11 follows from the following claim.

▷ Claim 12. For any $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{N}^n$, if there is a $\mathbf{w}$-homogenous circuit that computes $f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle$ of size s and non-scalar size $s_\times$, then there is a $\mathbf{w}$-homogeneous circuit that computes $\mathbb{D}(f) = \{\partial_{x_1} f, \ldots, \partial_{x_n} f\}$ of size at most $5s$ and non-scalar size at most $2s_\times$.

We prove this claim by induction on s. Recall that circuit size is measured by the number of non-input gates. For the base case, $s = 0$, the circuit only consists of leaves. The derivatives are then either 0 or 1 and can again be computed in zero size.

Assume $s > 0$. Let $\mathbf{w} = (w_1, \ldots, w_n) \in \mathbb{N}^n$ be arbitrarily fixed. Furthermore, suppose there is a $\mathbf{w}$-homogenous circuit $\mathcal{C}$ that computes $f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle$ of size s. Choose a vertex v in $\mathcal{C}$ such that both its children are leaves, and let $\widehat{v}$ be the polynomial it computes. $\widehat{v}$ is a homogeneous polynomial in at most two variables and degree at most two; w.l.o.g., we can also assume that $\widehat{v}$ is at least linear (otherwise v could be replaced by a leaf).

Let $\mathcal{C}'$ be the circuit obtained from $\mathcal{C}$ by removing the incoming edges to v and labelling the vertex v with a new variable, say x_0. Let us assign it weight $w_0 := \mathsf{wt}(\widehat{v})$.

Let f' be the polynomial computed by $\mathcal{C}'$. Then, $\mathbb{D}(f) = \{\partial_{x_1} f, \ldots, \partial_{x_n} f\}$ can be recovered from $\mathbb{D}(f') = \{\partial_{x_0} f', \partial_{x_1} f', \ldots, \partial_{x_n} f'\}$ using the following version of chain rule:

$$\partial_{x_k} f = (\partial_{x_k} f' + \partial_{x_k} \widehat{v} \cdot \partial_{x_0} f')|_{x_0 := \widehat{v}}.$$

Note that $\partial_{x_k} \widehat{v}$ is a variable or a constant, and that it is zero except for at most two of the x_k's.

Let us set $\mathbf{w}' = (w_0', w_1, \ldots, w_n)$. Note that the weight of every vertex in $\mathcal{C}'$ is the same as the corresponding vertex in $\mathcal{C}$. Therefore, since $\mathcal{C}$ is $\mathbf{w}$-homogeneous, $\mathcal{C}'$ is $\mathbf{w}'$-homogeneous. Furthermore, $\mathcal{C}'$ has $s - 1$ non-input gates and, by the inductive assumption, there is a $\mathbf{w}'$-homogeneous circuit $\mathcal{D}'$ of size $5(s-1)$ which computes $\mathbb{D}(f')$. Using $\mathcal{D}'$ and the chain rule above, we can construct a circuit with 5 additional gates which computes $\mathbb{D}(f)$. The size of this circuit is at most $5(s-1) + 5 = 5s$ and is easily seen to be $\mathbf{w}$-homogeneous.

When counting non-scalar complexity, note that in the construction, only non-scalar product gates introduce non-scalar gates, and we always introduce at most two such gates. ◀

We can now prove Theorem 2.

Proof of Theorem 2. Let n, d be given with[5] $n > 1$, $d > 2$. Let k be the smallest integer such that $n^k \geq n(d-1)$. Take a de Bruijn sequence σ of order k in alphabet $[n]$. Take sequences $\sigma^1, \ldots, \sigma^n \in [n]^{d-1}$ so that their concatenation $\sigma^1 \ldots \sigma^n$ is the initial segment of σ. Define the polynomial

$$f = x_1\alpha_1 + \cdots + x_n\alpha_n \text{ , where } \alpha_i = \prod_{j=1}^{d-1} x_{\sigma^i_j}.$$

Assume f has a homogeneous circuit of non-scalar size s. Then, by Lemma 11, $\alpha_1, \ldots, \alpha_n$ can be simultaneously computed by a homogeneous circuit of size $s' = O(s)$. We now apply Lemma 6 with $\ell = k$. By construction, $\mu_k(\alpha_1, \ldots, \alpha_n) = n(d-1-(k-1)) = n(d-k)$. This is because α_i^J are distinct monomials for different i's and intervals of length k. The lemma then gives $s' \geq n(d-k)/(k-1)$. If $d \leq n$, we have $k = 2$ and so $s' \geq n(d-2)$. If $d > n$, we have $k \leq c_1 \log_2 d / \log_2 n$ and $d - k \geq c_2 d$, for some constants $c_1, c_2 > 0$. Hence indeed $s' \geq \Omega(nd \frac{\log n}{\log d})$. ◀

4.3 Lower bound for ordered symmetric polynomials

We now prove Theorem 3. Firstly, we note the following.

▶ Remark 13. OS_n^2 requires $\Omega(n)$ non-scalar product gates (even in the commutative setting). This can be proved by a standard partial derivatives argument as in [10].

Hence we can focus on degree $d > 2$, in which case we give the following strengthening of Theorem 3:

[5] If $d = 2$, OS_n^2 satisfies the theorem; see Remark 13.

▶ **Theorem 14.** *If $1 < k < n$, any homogeneous circuit computing $\mathrm{OS}_n^{k+1}(x_1, \ldots, x_n)$ requires non-scalar size $\Omega(k(n-k))$.*

Proof. Assume that a homogeneous circuit computes $f = \mathrm{OS}_n^{k+1}(x_1, \ldots, x_n)$ using s non-scalar product gates. Then by Lemma 11 there is a homogeneous circuit of non-scalar size $O(s)$ which simultaneously computes $\{\partial_{x_1} f, \ldots, \partial_{x_n} f\}$. Let this circuit be $\boldsymbol{\mathcal{C}}$. Then, by Lemma 6, $\mu_2(\widehat{\boldsymbol{\mathcal{C}}}) \leq O(s)$. Note that

$$\partial_{x_i} f = \mathrm{OS}_{n-i}^{k}(x_{i+1}, \ldots, x_n)\,.$$

Let $f_{i,j} := (\partial_{x_i} f)^{[j,j+1]}$. We claim that the polynomials in $F := (f_{i,j} : i \in [n-k], j \in [k-1])$ are linearly independent. This implies that $\mu_2(\widehat{\boldsymbol{\mathcal{C}}}) \geq (n-k)(k-1)$ and gives a lower bound of $\Omega(k(n-k))$ as required.

We now prove that F is indeed linearly independent. Consider the lexicographic ordering on $S := [n-k] \times [k-1]$ defined by:

$$(i_0, j_0) < (i, j) \text{ iff } (j_0 > j) \text{ or } (j_0 = j \text{ and } i_0 < i)\,.$$

Let $(i_0, j_0) \in S$ be given. Denote $\delta_{i_0,j_0}(g)$ the coefficient of the monomial $x_{i_0+j_0} x_{n+j_0-k+1}$ in g. Then for every $(i, j) \in S$,

$$\delta_{i_0,j_0}(f_{i,j}) = \begin{cases} 1 & \text{if } (i_0, j_0) = (i, j) \\ 0 & \text{if } (i_0, j_0) < (i, j)\,. \end{cases} \tag{1}$$

To see (1), assume that $\partial_{x_i} f$ contains x_{n+j_0-k+1} in position $j+1$ in some monomial α with a non-zero coefficient. The degree of α is k, and the positions $j+1, \ldots, k$ need to be filled with variables from $x_{n+j_0-k+1}, \ldots, x_n$ in an ascending order. There are $k-j$ such positions and $k-j_0$ such variables. Therefore $j \geq j_0$. Furthermore, if $j = j_0$, the last $k-j_0$ positions in α are uniquely determined as the variables $x_{n+j_0-k+1}, \ldots, x_n$ in that order. Similarly, if $\partial_{x_i} f$ contains $x_{i_0+j_0}$ in position j_0 in some α, the first j_0 positions must be filled with variables from $x_{i+1}, \ldots, x_{i_0+j_0}$. Hence $i \leq i_0$, and in case of equality, the first j_0 positions are uniquely determined. This means that $\delta_{i_0,j_0}(f_{i,j}) = 0$ whenever $(i_0, j_0) < (i, j)$. Furthermore, $\alpha := \prod_{p=i_0+1}^{i_0+j_0} x_p \prod_{p=n+j_0-k+1}^{n} x_p$ is the unique monomial in f_{i_0,j_0} with $\delta_{i_0,j_0}(\alpha) = 1$, concluding (1).

Finally, assume for the sake of contradiction that there exists a non-trivial linear combination

$$\sum_{(i,j)\in S} \gamma_{i,j} f_{i,j} = 0\,.$$

Let (i_0, j_0) be the first pair in the lexicographic ordering with $\gamma_{i_0,j_0} \neq 0$. Then we have

$$0 = \sum_{(i,j)\in S} \gamma_{i,j} \delta_{i_0,j_0}(f_{i,j}) = \gamma_{i_0,j_0} \delta_{i_0,j_0}(f_{i_0,j_0}) + \sum_{(i,j)>(i_0,j_0)} \gamma_{i,j} \delta_{i_0,j_0}(f_{i,j})\,.$$

Using (1), the last sum is zero and $\gamma_{i_0,j_0} \delta_{i_0,j_0}(f_{i_0,j_0}) = \gamma_{i_0,j_0} = 0$, contrary to the assumption $\gamma_{i_0,j_0} \neq 0$. ◀

5 Upper bounds for ordered symmetric polynomials

In Proposition 4, we promised upper bounds on the complexity of elementary symmetric polynomials. The promise we now fulfil.

A quadratic upper bound in the homogeneous setting

We want to show that for $d \in \{0,\dots,n\}$, OS_n^d can be simultaneously computed by a homogeneous circuit of size $O(n^2)$.

Note that

$$\mathrm{OS}_n^d(x_1,\dots,x_n) = \mathrm{OS}_{n-1}^{d-1}(x_1,\dots,x_{n-1})\cdot x_n + \mathrm{OS}_{n-1}^{d}(x_1,\dots,x_{n-1}).$$

Hence, once we have computed OS_{n-1}^d, $d \in \{0,\dots,n-1\}$, we can compute OS_n^d, $d \in \{0,\dots,n\}$ using $O(n)$ extra gates. The overall complexity is quadratic.

An almost linear upper bound in the non-homogeneous setting

We want to show that OS_n^d, $d \in \{0,\dots,n\}$, can be simultaneously computed by a non-commutative circuit of size $n \cdot \mathrm{poly}(\log n)$.

The proof is the same as its commutative analog for elementary symmetric polynomials, see [1] or the monograph by Burgisser et al. [2, Chapters 2.1-2.3].

The main observation is that polynomial multiplication can be done efficiently. Let

$$f = \sum_{i=0}^{n} y_i t^i, \qquad g = \sum_{i=0}^{n} z_i t^i,$$

where $f, g \in \mathbb{F}\langle y_0,\dots,y_n,z_0,\dots,z_n\rangle[t]$. In other words, we assume that t commutes with otherwise non-commuting variables $y_0,\dots,y_n,z_0,\dots,z_n$. We view f,g as univariate polynomials in the variable t with non-commutative coefficients. Then $fg = \sum_{i=0}^{2n} c_i t^i$ with $c_i = \sum_{j=0}^{i} y_j z_{i-j}$. Commutatively, the polynomials $c_0,\dots,c_{2n}$ can be simultaneously computed by a small circuit. Indeed, if $\mathbb{F}$ contains sufficiently many roots of unity, one can obtain an $O(n\log n)$ circuit using Fast Fourier Transform; in other fields there are modification giving a circuit of size $O(n\log n\log\log n)$ see [12, 2]. When counting only non-scalar product gates, this can be improved to $O(n)$ if $\mathbb{F}$ is sufficiently large. We observe that the same holds if the coefficients of f,g do not commute. This is because the polynomials c_k are bilinear in $y_0,\dots,y_n,z_0,\dots,z_n$. Commutativity does not make a difference in this case (an exercise).

Now consider the polynomial $h_n(t) = \prod_{i=1}^{n}(x_i + t) \in \mathbb{F}\langle x_1,\dots,x_n\rangle[t]$. Then one can see that $\mathrm{OS}_n^d(x_1,\dots,x_n)$ is the coefficient of t^{n-d} in $h(t)$. The coefficients can be be recursively computed by first computing $\prod_{i=1}^{\lceil n/2\rceil}(x_i+t)$, $\prod_{i=\lceil n/2\rceil+1}^{n}(x_i+t)$, and then combining the two by means of the fast polynomial multiplication above. This gives the claimed complexity.

6 Open problems

We end with two open problems.

▶ **Open Problem 1.** *Find an explicit bivariate polynomial of degree d which requires non-commutative homogeneous circuit of size superlinear in d*

▶ **Open Problem 2.** *Given a non-commutative monomial α, can addition gates help to compute α?*

Observe that the bounds obtained in this paper are barely linear in d. Problem 1 simply asks for a quantitative improvement. A circuit with no addition gates is automatically homogeneous – hence a negative answer to Problem 2 would allow to remove the homogeneity assumption in Theorem 1.

References

1. Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theoretical computer science*, 22(3):317–330, 1983.
2. Peter Bürgisser, Michael Clausen, and M Amin Shokrollahi. Algebraic complexity theory, with the collaboration of thomas lickteig. *Grundlehren der Mathematischen Wissenschaften*, 315, 1997.
3. Marco L. Carmosino, Russell Impagliazzo, Shachar Lovett, and Ivan Mihajlin. Hardness amplification for non-commutative arithmetic circuits. In Rocco A. Servedio, editor, *33rd Computational Complexity Conference, CCC 2018, June 22-24, 2018, San Diego, CA, USA*, volume 102 of *LIPIcs*, pages 12:1–12:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018.
4. Prerona Chatterjee, Mrinal Kumar, Adrian She, and Ben Lee Volk. Quadratic lower bounds for algebraic branching programs and formulas. *Comput. Complex.*, 31(2):8, 2022.
5. N.G. de Bruijn. A combinatorial problem. *Proceedings of the Section of Sciences of the Koninklijke Nederlandse Akademie van Wetenschappen te Amsterdam*, 49(7):758–764, 1946.
6. Pavel Hrubes, Avi Wigderson, and Amir Yehudayoff. Non-commutative circuits and the sum-of-squares problem. *J. Amer. Math. Soc.*, 24(3):871–898, 2011.
7. Laurent Hyafil. The power of commutativity. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October – 1 November 1977*, pages 171–174. IEEE Computer Society, 1977.
8. K. Kalorkoti. A lower bound for the formula size of rational functions. *SIAM J. Comput.*, 14(3):678–687, 1985.
9. Noam Nisan. Lower bounds for non-commutative computation (extended abstract). In Cris Koutsougeras and Jeffrey Scott Vitter, editors, *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing, May 5-8, 1991, New Orleans, Louisiana, USA*, pages 410–418. ACM, 1991.
10. Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Comput. Complex.*, 6(3):217–234, 1997.
11. Joe Sawada, Aaron Williams, and Dennis Wong. Generalizing the classic greedy and necklace constructions of de bruijn sequences and universal cycles. *Electron. J. Comb.*, 23(1):1, 2016.
12. Arnold Schönhage and Volker Strassen. Schnelle Multiplikation großer Zahlen. *Computing*, 7(3-4):281–292, 1971.
13. Volker Strassen. Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten. *Numerische Mathematik*, 20(3):238–251, 1973.
14. Sébastien Tavenas, Nutan Limaye, and Srikanth Srinivasan. Set-multilinear and non-commutative formula lower bounds for iterated matrix multiplication. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20–24, 2022*, pages 416–425. ACM, 2022.
15. Leslie G. Valiant. Negation can be exponentially powerful. *Theor. Comput. Sci.*, 12:303–314, 1980.

On Relaxed Locally Decodable Codes for Hamming and Insertion-Deletion Errors

Alexander R. Block
University of Maryland, College Park, MD, USA
Georgetown University, Washington, D.C., USA

Jeremiah Blocki
Department of Computer Science,
Purdue University, West Lafayette, IN, USA

Kuan Cheng
Center on Frontiers of Computing Studies,
Peking University, China

Elena Grigorescu
Department of Computer Science,
Purdue University, West Lafayette, IN, USA

Xin Li
Department of Computer Science,
Johns Hopkins University, Baltimore, MD, USA

Yu Zheng
Meta Platforms, Inc., Bellevue, WA, USA

Minshen Zhu
Department of Computer Science,
Purdue University, West Lafayette, IN, USA

Abstract

Locally Decodable Codes (LDCs) are error-correcting codes $C : \Sigma^n \to \Sigma^m$, encoding *messages* in Σ^n to *codewords* in Σ^m, with super-fast decoding algorithms. They are important mathematical objects in many areas of theoretical computer science, yet the best constructions so far have codeword length m that is super-polynomial in n, for codes with constant query complexity and constant alphabet size.

In a very surprising result, Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan (SICOMP 2006) show how to construct a relaxed version of LDCs (RLDCs) with constant query complexity and almost linear codeword length over the binary alphabet, and used them to obtain significantly-improved constructions of Probabilistically Checkable Proofs.

In this work, we study RLDCs in the standard Hamming-error setting, and introduce their variants in the insertion and deletion (Insdel) error setting. Standard LDCs for Insdel errors were first studied by Ostrovsky and Paskin-Cherniavsky (*Information Theoretic Security, 2015*), and are further motivated by recent advances in DNA random access bio-technologies.

Our first result is an exponential lower bound on the length of Hamming RLDCs making 2 queries (even adaptively), over the binary alphabet. This answers a question explicitly raised by Gur and Lachish (SICOMP 2021) and is the first exponential lower bound for RLDCs. Combined with the results of Ben-Sasson et al., our result exhibits a "phase-transition"-type behavior on the codeword length for some constant-query complexity. We achieve these lower bounds via a transformation of RLDCs to standard Hamming LDCs, using a careful analysis of restrictions of message bits that fix codeword bits.

We further define two variants of RLDCs in the Insdel-error setting, a weak and a strong version. On the one hand, we construct weak Insdel RLDCs with almost linear codeword length and constant query complexity, matching the parameters of the Hamming variants. On the other hand, we prove exponential lower bounds for strong Insdel RLDCs. These results demonstrate that, while these variants are equivalent in the Hamming setting, they are significantly different in the insdel setting. Our results also prove a strict separation between Hamming RLDCs and Insdel RLDCs.

2012 ACM Subject Classification Theory of computation → Error-correcting codes; Mathematics of computing → Coding theory; Theory of computation → Lower bounds and information complexity

Keywords and phrases Relaxed Locally Decodable Codes, Hamming Errors, Insdel Errors, Lower Bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.14

Related Version *Full Version*: https://arxiv.org/abs/2209.08688

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 14; pp. 14:1–14:25

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

Funding *Alexander R. Block*: NSF Award CCF-1910659 and DARPA agreement No. HR00112020022 and No. HR00112020025. The views, opinions, findings, conclusions and/or recommendations expressed in this material are those of the author and should not be interpreted as reflecting the position or policy of the Department of Defense or the U.S. Government, and no official endorsement should be inferred.
Jeremiah Blocki: NSF CAREER Award CNS-2047272 and NSF Award CCF-1910659.
Elena Grigorescu: NSF CCF-1910659, NSF CCF-1910411, and NSF CCF-2228814.
Xin Li: NSF CAREER Award CCF-1845349 and NSF Award CCF-2127575.
Yu Zheng: NSF CAREER Award CCF-1845349.
Minshen Zhu: NSF CCF-1910659, NSF CCF-1910411, and NSF CCF-2228814.

Acknowledgements We are indebted to some anonymous reviewers who helped us improve the presentation of the paper.

1 Introduction

Locally Decodable Codes (LDCs) [55, 72] are error-correcting codes $C : \Sigma^n \to \Sigma^m$ that have super-fast decoding algorithms that can recover individual symbols of a *message* $x \in \Sigma^n$, even when worst-case errors are introduced in the *codeword* $C(x)$. Similarly, Locally Correctable Codes (LCCs) are error-correcting codes $C : \Sigma^n \to \Sigma^m$ for which there exist very fast decoding algorithms that recover individual symbols of the *codeword* $C(x) \in \Sigma^m$, even when worst-case errors are introduced. LDCs/LCCs were first discovered by Katz and Trevisan [55] and since then have proven to be crucial tools in many areas of computer science, including private information retrieval, probabilistically checkable proofs, self-correction, fault-tolerant circuits, hardness amplification, and data structures (e.g., [2, 4, 17, 18, 20, 28, 62] and surveys [36, 73]).

The *parameters* of interest of these codes are their *rate*, defined as the ratio between the message length n and the codeword length m, their *relative minimum distance*, defined as the minimum normalized Hamming distance between any pair of codewords, and their *locality* or *query complexity*, defined as the number of queries a decoder makes to a received word $y \in \Sigma^m$. Trade-offs between the achievable parameters of Hamming LDCs/LCCs have been studied extensively over the last two decades [8–11, 32–35, 37, 56, 57, 74, 75, 78, 79] (see also surveys by Yekhanin [79] and by Kopparty and Saraf [58]).

Specifically, for 2-query Hamming LDCs/LCCs it is known that $m = 2^{\Theta(n)}$ [6, 11, 37, 56]. However, for $q > 2$ queries, the current gap between upper and lower bounds is superpolynomial in n. In particular, the best constructions have super-polynomial codeword length [32, 34, 78], while the most general lower bounds for $q \geq 3$ are of the form $m = \Omega((\frac{n}{\log n})^{1+1/(\lceil \frac{q}{2} \rceil - 1)})$ [55, 56]. In particular, for $q = 3$, [55] showed an $m = \Omega(n^{3/2})$ bound, which was improved in [56] to $m = \Omega(n^2/\log^2 n)$. This was further improved by [75, 76] to $m = \Omega(n^2/\log n)$ for general codes and $m = \Omega(n^2)$ for linear codes. [11] used new combinatorial techniques to obtain the same $m = \Omega(n^2/\log n)$ bound. A very recent paper [1] breaks the quadratic barrier and proves that $m = \Omega(n^3/\operatorname{poly}\log n)$. We note that the exponential lower bound on the length of 3-query LDCs from [35] holds only for some restricted parameter regimes, and do not apply to the natural ranges of the known upper bounds.

Motivated by this large gap in the constant-query regime, as well as by applications in constructions of Probabilistically Checkable Proofs (PCPs), Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan [7] introduced a relaxed version of LDCs for Hamming errors. Specifically, the decoder is allowed to output a "decoding failure" answer (marked as "$\perp$"), as long as it errs with some small probability. More precisely, a $(q, \delta, \alpha, \rho)$-*relaxed LDC* is an error-correcting code satisfying the following properties.

▶ **Definition 1.** *A $(q, \delta, \alpha, \rho)$-Relaxed Locally Decodable Code $C : \Sigma^n \to \Sigma^m$ is a code for which there exists a decoder that makes at most q queries to the received word y, and satisfies the following further properties:*

1. *(Perfect completeness) For every $i \in [n]$, if $y = C(x)$ for some message x then the decoder, on input i, outputs x_i with probability 1.*[1]
2. *(Relaxed decoding) For every $i \in [n]$, if y is such that $dist(y, C(x)) \leq \delta$ for some unique $C(x)$, then the decoder, on input i, outputs x_i or $\perp$ with probability $\geq \alpha$.*
3. *(Success rate) For every y such that $dist(y, C(x)) \leq \delta$ for some unique $C(x)$, there is a set I of size $\geq \rho n$ such that for every $i \in I$ the decoder, on input i, correctly outputs x_i with probability $\geq \alpha$.*

We will call an RLDC that satisfies all 3 *conditions by the notion of* strong *RLDC, and one that satisfies just the first* 2 *conditions by the notion of* weak *RLDC, in which case it is called a (q, δ, α)-RLDC. Furthermore, if the q queries are made in advance, before seeing entries of the codeword, then the decoder is said to be* non-adaptive*; otherwise, it is called* adaptive*.*

The above definition is quite general, in the sense that $dist(a, b)$ can refer to several different distance metrics. In the most natural setting, we use $dist(a, b)$ to mean the "relative" Hamming distance between $a, b \in \Sigma^m$, namely $dist(a, b) = |\{i \colon a_i \neq b_i\}|/m$. This corresponds to the standard RLDCs for Hamming errors. As it will be clear from the context, we also use $dist(a, b)$ to mean the "relative" Edit distance between $a, b \in \Sigma^*$, namely $dist(a, b) = \mathsf{ED}(a, b)/(|a| + |b|)$, where $\mathsf{ED}(a, b)$ is the minimum number of insertions and deletions to transform string a into b. This corresponds to the new notion introduced and studied here, which we call *Insdel RLDCs*. Throughout this paper, we only consider the case where $\Sigma = \{0, 1\}$.

Definition 1 has also been extended recently to the notion of *Relaxed Locally Correctable Codes (RLCCs)* by Gur, Ramnarayan, and Rothblum [40]. RLDCs and RLCCs have been studied in a sequence of exciting works, where new upper and lower bounds have emerged, and new applications to probabilistic proof systems have been discovered [3, 27, 29, 38–40].

Surprisingly, [7] constructs strong RLDCs with $q = O(1)$ queries and $m = n^{1+O(1/\sqrt{q})}$, and more recently Asadi and Shinkar [3] improve the bounds to $m = n^{1+O(1/q)}$, in stark contrast with the state-of-the-art constructions of standard LDCs. Gur and Lachish [39] show that these bounds are in fact tight, as for every $q \geq 2$, every weak q-query RLDC must have length $m = n^{1+1/O(q^2)}$ for non-adaptive decoders. We remark that the lower bounds of [39] hold even when the decoder does not have perfect completeness and in particular valid message bits are decoded with success probability 2/3. Dall'Agnon, Gur, and Lachish [30] further extend these bounds to the setting where the decoder is adaptive, with $m = n^{1+1/O(q^2 \log^2 q)}$.

1.1 Our results

As discussed before, since the introduction of RLDCs, unlike standard LDCs, they displayed a behaviour amenable to nearly linear-size constructions, with almost matching upper and lower bounds. However, recently [39] conjecture that for $q = 2$ queries, there is in fact an exponential lower bound, matching the bounds for standard LDCs.

[1] We remark that the initial definition in [7] only requires that x_i is output with probability 2/3 when there are no errors. However, to the best of our knowledge, all constructions of RLDCs (and LDCs) from the literature do satisfy perfect completeness. Moreover, some lower bounds (e.g., [11]) only hold with respect to perfect completeness.

In this paper, our first contribution is a proof of their conjecture, namely to show that Hamming 2-query RLDCs require exponential length. In fact, our exponential lower bound for $q = 2$ applies even to weak RLDCs, which only satisfy the first two properties (perfect completeness and relaxed decoding), and even for adaptive decoders.

▶ **Theorem 2.** *Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be a weak adaptive $(2, \delta, 1/2 + \varepsilon)$-RLDC. Then $m = 2^{\Omega_{\delta,\varepsilon}(n)}$.*

Our results are the first exponential bounds for RLDCs. Furthermore, combined with the constructions with nearly linear codeword length for some constant number of queries [3, 7], our results imply that RLDCs experience a "phase transition"-type phenomena, where the codeword length drops from being exponential at $q = 2$ queries to being almost linear at $q = c$ queries for some constant $c > 2$. In particular, this also implies that there is a query number q where the codeword length drops from being super-polynomial at q to being polynomial at $q + 1$. Finding this exact threshold query complexity is an intriguing open question.

As our second contribution, we introduce and study the notion of RLDCs correcting *insertions and deletions*, namely Insdel RLDCs. The non-relaxed variants of Insdel LDCs were first introduced in [68], and were further studied in [12, 13, 26]. Local decoding in the Insdel setting is motivated in DNA storage [77], and in particular [5] show recent advances in bio-technological aspects of random access to data in these precise settings.

In [13, 68], the authors give Hamming to Insdel reductions which transform any Hamming LDC into an Insdel LDC with rate reduced by a constant multiplicative factor, and locality increased by a polylog(m) multiplicative factor. Unfortunately, these compilers do not imply constant-query Insdel LDCs, whose existence is still an open question.

The results of [14] show strong lower bounds on the length of constant-query Insdel LDCs. In particular, they show that linear Insdel LDCs with 2 queries do no exist, general Insdel LDCs for $q = 3$ queries must have $m = \exp(\Omega(\sqrt{n}))$, and for $q \geq 4$ they must have $m = \exp(n^{\Omega(1/q)})$.

In this work we continue the study of locally decodable codes in insertion and deletion channels by proving the first upper and lower bounds regarding the relaxed variants of Insdel LDCs. We first consider strong Insdel RLDCs, which satisfy all three properties of Definition 1 and where the notion of distance is now that of relative edit distance. We adapt and extend the results of [14] to establish strong lower bounds on the codeword length of strong Insdel RLDCs. In particular, we prove that $m = \exp(n^{\Omega(1/q)})$ for any strong Insdel RLDC with locality q.

▶ **Theorem 3.** *Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be a non-adaptive strong $(q, \delta, 1/2 + \beta, \rho)$-Insdel RLDC where $\beta > 0$. Then for every $q \geq 2$ there is a constant $c_1 = c_1(q, \delta, \beta, \rho)$ such that*

$$m = \exp\left(c_1 \cdot n^{\Omega_\rho(\beta^2/q)}\right).$$

Furthermore, the same bound holds even if C does not have perfect completeness. If C has an adaptive decoder, the same bound holds with β replaced by $\beta/2^{q-1}$. Formally, there exists a constant $c_2 = c_1(q, \delta, \beta/2^{q-1}, \rho)$ such that

$$m = \exp\left(c_2 \cdot n^{\Omega_\rho(\beta^2/(q2^{2q}))}\right).$$

Our reduction shown in the proof of Theorem 2, together with the impossibility results of standard *linear* or *affine* 2-query Insdel LDCs from [14] show a further impossibility result for linear and for affine 2-query Insdel RLDCs. A linear code of length m is defined over a finite field $\mathbb{F}$ and it is a linear subspace of the vector space $\mathbb{F}^m$, while an affine code is an affine subspace of $\mathbb{F}^m$.

We then consider *weak* Insdel RLDCs that only satisfy the first two properties (perfect completeness and relaxed decoding). In contrast with Theorem 3, we construct weak Insdel RLDCs with constant locality $q = O(1)$ and length $m = n^{1+\gamma}$ for some constant $\gamma \in (0, 1)$. To the best of our knowledge, this is the first positive result in the constant-query regime and the Insdel setting. However, the existence of a constant-query standard Insdel LDC (or even a constant-query strong Insdel RLDC) with any rate remains an open question. Finally, it is easy to see that our exponential lower bound for weak Hamming RLDCs with locality $q = 2$ still applies in the Insdel setting, since Insdel errors are more general than Hamming error. Thus, in the Insdel setting we discover the same "phase transition"-type phenomena as for Hamming RLDCs.

▶ **Theorem 4.** *For any $\gamma > 0$ and $\varepsilon \in (0, 1/2)$, there exist constants $\delta \in (0, 1/2)$ and $q = q(\delta, \varepsilon, \gamma)$, and non-adaptive weak $(q, \delta, 1/2+\varepsilon)$-Insdel RLDCs $C\colon \{0,1\}^n \to \{0,1\}^m$ with $m = O(n^{1+\gamma})$.*

We remark that in the Hamming setting, [7] shows that the first two properties of Definition 1 imply the third property for codes with constant query complexity and which can withstand a constant fraction of errors. Our results demonstrate that, in general, unlike in the Hamming case, the first two properties do not imply the third property for Insdel RLDCs from Definition 1. Indeed, while for strong Insdel RLDCs we have $m = \exp(n^{\Omega(1/q)})$ for codes of locality q, there exists $q = O(1)$ for which we have constructions of weak Insdel RLDCs with $m = n^{1+\gamma}$. This observation suggests that there are significant differences between Hamming RLDCs and Insdel RLDCs.

We note that our construction of weak Insdel RLDCs can be modified to obtain strong Insdel Relaxed Locally Correctable Codes (Insdel RLCCs). Informally, an Insdel RLCC is a code for which codeword entries can be decoded to the correct value or $\perp$ with high probability, even in the presence of insdel errors (see the full version for a formal definition of RLCC). We have the following corollary.

▶ **Corollary 5.** *For any $\gamma > 0$ and $\varepsilon \in (0, 1/2)$, there exist constants $\delta \in (0, 1/2)$ and $q = q(\delta, \varepsilon, \gamma)$, and non-adaptive strong $(q, \delta, 1/2+\varepsilon, 1/2)$-Insdel RLCCs $C\colon \{0,1\}^n \to \{0,1\}^m$ with $m = O(n^{1+\gamma})$.*

1.2 Overview of techniques

1.2.1 Exponential Lower Bound for Weak Hamming RLDCs with $q = 2$

To simplify the presentation, we assume a non-adaptive decoder in this overview. While the exact same arguments do not directly apply to adaptive decoders[2], with a bit more care they can be adapted to work in those settings.

[2] For standard LDCs Katz and Trevisan [55] observed that an adaptive decoder could be converted into a non-adaptive decoder by randomly guessing the output y_j of the first query j to learn the second query k. Now we non-adaptively query the received codeword for both y_j and y_k. If our guess for y_j was correct then we continue simulating the adaptive decoder. Otherwise, we simply guess the output x_i. If the adaptive decoder succeeds with probability at least $p \geq 1/2 + \epsilon$ then the non-adaptive decoder succeeds with probability $p' \geq 1/4 + p/2 \geq 1/2 + \epsilon/2$. Unfortunately, this reduction does not preserve perfect completeness as required by our proofs for relaxed 2-query Hamming RLDCs i.e., if $p = 1$ then $p' = 3/4$.

At a high level we prove our lower bound by transforming any non-adaptive 2-query weak Hamming RLDC for messages of length n and δ fraction of errors into a standard 2-query Hamming LDC for messages of length $n' = \Omega(n)$, with slightly reduced error tolerance of $\delta/2$. Kerenidis and de Wolf [56] proved that any 2-query Hamming LDC for messages of length n must have codeword length $m = \exp(\Omega(n))$. Combining this result with our transformation, it immediately follows that any 2-query weak Hamming RLDC must also have codeword length $m = \exp(\Omega(n))$. While our transformation does not need the third property (success rate) of a strong RLDC, we crucially rely on the property of *perfect completeness*, and that the decoder only makes $q = 2$ queries.

Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be a weak $(2, \delta, 1/2 + \varepsilon)$-RLDC. For simplicity (and without loss of generality), let us assume the decoder Dec works as follows. For message x and input $i \in [n]$, the decoder non-adaptively makes 2 random queries $j, k \in [m]$, and outputs $f^i_{j,k}(y_j, y_k) \in \{0, 1, \perp\}$, where y_j, y_k are answers to the queries from a received word y, and $f^i_{j,k}\colon \{0,1\}^2 \to \{0,1,\perp\}$ is a deterministic function. When there is no error, we have $y_j = C(x)_j$ and $y_k = C(x)_k$.

We present the main ideas below, and refer the readers to Section 4 for full details.

1.2.1.1 Fixable codeword bits

The starting point of our proof is to take a closer look at those functions $f^i_{j,k}$ with $\perp$ entries in their truth tables. It turns out that when $f^i_{j,k}$ has at least one $\perp$ entry in the truth table, $C(x)_j$ can be fixed to a constant by setting either $x_i = 0$ or $x_i = 1$, and same for $C(x)_k$. To see this, note that the property of perfect completeness forces $f^i_{j,k}$ to be 0 or 1 whenever $x_i = 0$ or $x_i = 1$ and there is no error. Thus if neither $x_i = 0$ nor $x_i = 1$ fixes $C(x)_j$, then there must be two entries of 0 and two entries of 1 in the truth table of $f^i_{j,k}$, which leaves no space for $\perp$ (see Claim 13). Thus, when there is at least one $\perp$ entry in the truth table of $f^i_{j,k}$, we say that $C(x)_j$ and $C(x)_k$ are *fixable* by x_i.

This motivates the definition of the set S_i, which contains all indices $j \in [m]$ such that the codeword bits $C(x)_j$ are fixable by x_i; and the definition of T_j, the set of all indices $i \in [n]$ such that $C(x)_j$ is fixable by the message bits x_i. It is also natural to pay special attention to queries j, k that are not both contained in S_i, since in this case the function $f^i_{j,k}$ never outputs $\perp$.

1.2.1.2 The query structure

In general, a query set $\{j, k\}$ falls into one of the following three cases: (1) both j, k lie inside S_i; (2) both j, k lie outside of S_i; (3) one of them lies inside S_i and the other lies outside of S_i. It turns out that case (3) essentially never occurs for a decoder with perfect completeness. The reason is that when, say, $j \in S_i$ and $k \notin S_i$, one can effectively pin down every entry in the truth table of $f^i_{j,k}$ by using the perfect completeness property, and observe that the output of $f^i_{j,k}$ does not depend on y_k at all (see Claim 14). Thus in this case we can equivalently view the decoder as only querying y_j where $j \in S_i$, which leads us back to case (1). In what follows, we denote by E_1 the event that case (1) occurs, and by E_2 the event that case (2) occurs.

1.2.1.3 The transformation by polarizing conditional success probabilities

We now give a high level description of our transformation from a weak RLDC to a standard LDC. Let y be a string which contains at most $\delta m/2$ errors from the codeword $C(x)$. We have established that the success probability of the weak RLDC decoder on y is an average of two conditional probabilities

$$\Pr[\mathsf{Dec}(i,y) \in \{x_i, \perp\}] = p_1 \cdot \Pr[\mathsf{Dec}(i,y) \in \{x_i, \perp\} \mid E_1] + p_2 \cdot \Pr[\mathsf{Dec}(i,y) \in \{x_i, \perp\} \mid E_2],$$

where $p_1 = \Pr[E_1]$ and $p_2 = \Pr[E_2]$. Let us assume for the moment that S_i has a small size, e.g., $|S_i| \leq \delta m/2$. The idea in this step is to introduce additional errors to the S_i-portion of y, in a way that drops the conditional success probability $\Pr[\mathsf{Dec}(i,y) \in \{x_i, \perp\} \mid E_1]$ to 0 (see Lemma 15). In particular, we modify the bits in S_i to make it consistent with the encoding of any message $\hat{x}$ with $\hat{x}_i = 1 - x_i$. Perfect completeness thus forces the decoder to output $1 - x_i$ conditioned on E_1. Note that we have introduced at most $\delta m/2 + |S_i| \leq \delta m$ errors in total, meaning that the decoder should still have an overall success probability of $1/2 + \varepsilon$. Furthermore, now the conditional probability $\Pr[\mathsf{Dec}(i,y) \in \{x_i, \perp\} \mid E_2]$ takes all credits for the overall success probability. Combined with the observation that Dec never outputs $\perp$ given E_2, this suggests the following natural way to decode x_i in the sense of a standard LDC: sample queries j, k according to the conditional probability given E_2 (i.e., both j, k lie outside S_i) and output $f^i_{j,k}(y_j, y_k)$. This gives a decoding algorithm for standard LDC, with success probability $1/2 + \varepsilon$ and error tolerance $\delta m/2$ (see Lemma 16), modulo the assumption that $|S_i| \leq \delta m/2$.

1.2.1.4 Upper bounding $|S_i|$

The final piece in our transformation from weak RLDC to standard LDC is to address the assumption that $|S_i| \leq \delta m/2$. This turns out to be not true in general, but it would still suffice to prove that $|S_i| \leq \delta m/2$ for $n' = \Omega(n)$ of the message bits i. If we could show that $|T_j|$ is small for most $j \in [m]$, then a double counting argument shows that $|S_i|$ is small for most $i \in [n]$. Unfortunately, if we had $C(x)_j = \bigwedge_{i=1}^n x_i$ for $m/2$ of the codeword bits j then we also have $|T_j| = n$ for $m/2$ codeword bits and $|S_i| \geq m/2 \geq \delta m/2$ for all message bits $i \in [n]$. We address this challenge by first arguing that any weak RLDC for n-bit messages can be transformed into another weak RLDC for $\Omega(n)$-bit messages for which we have $|T_j| \leq 3\ln(8/\delta)$ for all but $\delta m/4$ codeword bits. The transformation works by fixing some of the message bits and then eliminating codeword bits that are fixed to constants. Intuitively, if some $C(x)_j$ is fixable by many message bits, it will have very low entropy (e.g., $C(x)_j$ is the AND of many message bits) and hence contain very little information and can (likely) be eliminated. We make this intuition rigorous through the idea of random restriction: for each $i \in [n]$, we fix $x_i = 0$, $x_i = 1$, or leave x_i free, each with probability $1/3$. The probability that $C(x)_j$ is not fixed to a constant is at most $(1 - 1/3)^{|T_j|} \leq \delta/8$, provided that $|T_j| \geq 3\ln(8/\delta)$. After eliminating codeword bits that are fixed to constants, we show that with probability at least $1/2$ at most $\delta m/4$ codeword bits $C(x)_j$ with $|T_j| \geq 3\ln(8/\delta)$ survived[3]. Note that with high probability the random restriction leaves at least $n/6$ message bits free. Thus, there must exist a restriction which leaves at least $n/6$ message bits free ensuring that $|T_j| \geq 3\ln(8/\delta)$ for at most $\delta m/4$ of the remaining codeword bits $C(x)_j$. We can now apply the double counting argument to conclude that $|S_i| \leq \delta m/2$ for $\Omega(n)$ message bits, completing the transformation.

[3] We are oversimplifying a bit for ease of presentation. In particular, the random restriction process may cause a codeword bit $C(x)_j$ to be fixable by a new message bit x_i that did not belong to T_j before the restriction – We thank an anonymous reviewer for pointing this out to us. Nevertheless, for our purpose it is sufficient to eliminate codeword bits that initially have a large $|T_j|$. See the formal proof for more details.

1.2.1.5 Adaptive decoders

For possibly adaptive decoders, we are going to follow the same proof strategy. The new idea and main difference is that we focus on the first query made by the decoder, which is always non-adaptive. We manage to show that the first query determines a similar query structure, which is the key to the transformation to a standard LDC. More details can be found in Section 4.2.

1.2.2 Lower Bounds for Strong Insdel RLDCs

We recall that a strong Insdel RLDC C is a weak Insdel RLDC which satisfies an additional property: for every $x \in \{0,1\}^n$ and $y \in \{0,1\}^{m'}$ such that $\mathsf{ED}(C(x), y) \leq \delta \cdot 2m$, there exists a set $I_y \subseteq [n]$ of size $|I_y| \geq \rho n$ such that for every $i \in I_y$, we have $\Pr[\mathsf{Dec}(i, y) = x_i] \geq \alpha$. In other words, for ρ-fraction of the message bits, the decoder can correctly recover them with high probability, just like in a standard Insdel LDC. Towards obtaining a lower bound on the codeword length m, a natural idea would be to view C as a standard Insdel LDC just for that ρ-fraction of message bits, and then apply the exponential lower bound for standard Insdel LDCs from [14]. This idea would succeed if the message bits correctly decoded with high probability were the same for all potential corrupted codewords y. However, it could be the case that $i \in I_y$ for some strings y, whereas $i \notin I_{y'}$ for other strings y'. Indeed, allowing the set I_y to depend on y is the main reason why very short constant-query Hamming RLDCs exist.

We further develop this observation to obtain our lower bound. We use an averaging argument to show the existence of a *corruption-independent* set I of message bits with $|I| = \Omega(n)$, which the decoder can recover with high probability. To this end, we need to open the "black box" of the lower bound result of Blocki et al. [14]. The proof in [14] starts by constructing an error distribution $\mathcal{E}$ with several nice properties, and deduce the exponential lower bound based solely on the fact that the Insdel LDC should, on average (i.e., for a uniformly random message x), correctly recover each bit with high probability under $\mathcal{E}$. One of the nice properties of $\mathcal{E}$ is that it is oblivious to the decoding algorithm Dec. Therefore, it makes sense to consider the average success rate against $\mathcal{E}$, i.e., $\Pr[\mathsf{Dec}(i, y) = x_i]$, where $i \in [n]$ is a uniformly random index, $x \in \{0,1\}^n$ is a uniformly random string, and y is a random string obtained by applying $\mathcal{E}$ to $C(x)$. By replacing $\perp$ with a uniformly random bit in the output of Dec, the average success rate is at least $\rho\alpha + (1-\rho)\alpha/2 = (1+\rho)\alpha/2$, since there is a ρ-fraction of indices for which Dec can correctly recover with probability α, and for the remaining $(1-\rho)$-fraction of indices the random guess provides an additional success rate of at least $\alpha/2$. Assuming α is sufficiently close to 1, which we can achieve by repeating the queries independently for a constant number of times and doing something similar to a majority vote, the average success rate against $\mathcal{E}$ is strictly above $1/2$. Therefore, there exist a constant fraction of indices for which the success rate against $\mathcal{E}$ is still strictly above $1/2$, and the number of queries remains a constant. This is sufficient for the purpose of applying the argument in [14] to get an exponential lower bound.

1.2.3 Constant-Query Weak Insdel RLDC

Our construction of a constant query weak Insdel RLDC uses code concatenation and two building blocks: a weak Hamming RLDC (as the outer code) with constant query complexity, constant error-tolerance, and codeword length $k = O(n^{1+\gamma})$ for any $\gamma > 0$ [7], and the

Schulman-Zuckerman [69] (from now on denoted by SZ) Insdel codes[4] (as the inner code). We let $C_{\text{out}}\colon \{0,1\}^n \to \{0,1\}^k$ and $C_{\text{in}}\colon [k] \times \{0,1\} \to \{0,1\}^t$ denote the outer and inner codes, respectively. Our final concatenation code C will have codewords in $\{0,1\}^m$ for some m (to be determined shortly), will have constant query complexity, and will tolerate a constant fraction of Insdel errors.

1.2.3.1 Code construction

Given a message $x \in \{0,1\}^n$, we first apply the outer code to obtain a Hamming codeword $y = y_1 \circ \cdots \circ y_k = C_{\text{out}}(x)$ of length k, where each $y_i \in \{0,1\}$ denotes a single bit of the codeword. Then for each index i, we compute $c_i = C_{\text{in}}(i, y_i) \in \{0,1\}^t$ as the encoding of the message (i, y_i) via the inner code. Finally, we output the codeword $C(x) \coloneqq c_1 \circ 0^t \circ c_2 \circ \cdots \circ 0^t \circ c_k$, where 0^t denotes a string of t zeros (which we later refer to as a buffer). Note that the inner code is a constant-rate code, i.e., $t = O(\log(k))$, and has constant error-tolerance $\delta_{\text{in}} \in (0, 1/2)$. Thus, the final codeword has length $m \coloneqq (2t-1)k = O(k \log(k))$ bits. For any constant $\gamma > 0$ we have a constant query outer code with length $k = O(n^{1+\gamma})$. Plugging this into our construction we have codeword length $m = O(n^{1+\gamma} \log n)$ which is $O(n^{1+\gamma'})$ for any constant $\gamma' > \gamma$.

1.2.3.2 Decoding algorithm: intuition and challenges

Intuitively, our relaxed decoder will simulate the outer decoder. When the outer decoder requests y_i, the natural approach would be to find and decode the block c_i to obtain (i, y_i). There are two challenges in this approach. First, if there were insertions or deletions, then we do not know where the block c_i is located; moreover, searching for this block can potentially blow-up the query complexity by a multiplicative polylog(m) factor [13, 68]. Second, even if we knew where c_i were located, because $t = O(\log k)$ and we want the decoder to have constant locality, we cannot afford to recover the entire block c_i.

We address the first challenge by attempting to locate block c_i under the optimistic assumption that there are no corruptions. If we detect any corruptions, then we may immediately abort and output $\perp$ since our goal is only to obtain a weak Insdel RLDC. Assuming that there were no corruptions, we know exactly where the block c_i is located, and we know that c_i can only take on two possible values: it is either the inner encoding of $(i, 0)$ or the inner encoding of $(i, 1)$. If we find anything inconsistent with the inner encoding of either $(i, 0)$ or $(i, 1)$, then we can immediately output $\perp$.

Checking consistency with the inner encodings of $(i, 0)$ and $(i, 1)$ is exactly how we address the second challenge. In place of reading the entire block c_i, we instead only need to determine whether (1) c_i is (close to) the inner encoding of $(i, 0)$, (2) c_i is (close to) the inner encoding of $(i, 1)$, or (3) c_i is not close to either string. In either case (1) or case (2), we simply output the appropriate bit, and in case (3), we simply output $\perp$. Thus, our Insdel RLDC decoder simulates the outer decoder. Whenever the outer decoder request y_i, we determine the expected location for c_i, randomly sub-sample a constant number of indices from this block and compare with the inner encodings of $(i, 0)$ and $(i, 1)$ at the corresponding indices. To ensure perfect completeness, we always ensure that *at least one* of the sub-sampled indices is for a bit where the inner encodings of $(i, 0)$ and $(i, 1)$ differ. If there are no corruptions, then whenever the simulated outer decoder requests y_i we will always respond with the correct bit. Perfect completeness of our Insdel RLDC now follows

[4] In particular, these are classical/non-local codes.

immediately from the perfect completeness of the outer decoder. Choosing a constant number of indices to sub-sample ensures that the locality of our weak Insdel RLDC decoder is a constant multiplicative factor larger than the outer decoder, which gives our Insdel RLDC decoder constant locality overall.

1.2.3.3 Analysis of the decoding algorithm

The main technical challenge is proving that our Insdel RLDC still satisfies the second condition of Definition 1, when the received word is not a correct encoding of the message x. Recall that $c_i = C_{\text{in}}(i, y_i)$, and suppose $\widetilde{c}_i \neq c_i$ is the block of the received word that we are going to check for consistency with the inner encodings of $(i, 0)$ and $(i, 1)$. Then, the analysis of our decoder falls into three cases. In the first case, if $\widetilde{c}_i$ is not too corrupted (i.e., $\mathsf{ED}(\widetilde{c}_i, c_i)$ is not too large), then we can argue that the decoder outputs the correct bit y_i or $\perp$ with good probability. In the second case, if $\widetilde{c}_i$ has high edit distance from both $C_{\text{in}}(i, 0)$ and $C_{\text{in}}(i, 1)$, then we can argue that the decoder outputs $\perp$ with good probability. The third case is the most difficult case, which we describe as "dangerous". We say that the block $\widetilde{c}_i$ is *dangerous* if the edit distance between $\widetilde{c}_i$ and $C_{\text{in}}(i, 1 - y_i)$ is not too large; i.e., $\widetilde{c}_i$ is close to the encoding of the opposite bit $1 - y_i$.

The key insight to our decoding algorithm is that as long as the number of dangerous blocks $\widetilde{c}_i$ is upper bounded, then we can argue the overall probability that our decoder outputs y_i or $\perp$ satisfies the relaxed decoding condition of Definition 1. Intuitively, we can we think of our weak Insdel RLDC decoder as running the outer decoder on a string $\tilde{y} = \tilde{y}_1 \circ \ldots \circ \tilde{y}_k$, where each $\tilde{y}_i \in \{0, 1, \perp\}$ and the outer decoder has been modified to output $\perp$ whenever it queries for y_i and receives $\perp$. Observe that if δ_{out} is the error-tolerance of the outer decoder, then as long as the set $\left|\{i \ : \ \widetilde{y}_i \neq \perp \wedge \widetilde{y}_i \neq y_i\}\right| \leq \delta_{\text{out}} k$, the modified outer decoder, on input $j \in [n]$, will output either the correct value x_j or $\perp$ with high probability (for appropriate choices of parameters). Intuitively, if a block is "dangerous" then we can view $\widetilde{y}_i = 1 - y_i$, and otherwise we have $\widetilde{y}_i \in \{y_i, \perp\}$ with reasonably high probability. Thus, as long as the number of "dangerous" block is at most $\delta_{\text{out}} k/2$, then our relaxed Insdel decoder will satisfy the second property of Definition 1 and output either x_j or $\perp$ with high probability for any $j \in [n]$.

1.2.3.4 Upper bounding the number of dangerous blocks

To upper bound the number of "dangerous" blocks we utilize a matching argument based on the longest common sub-sequence (LCS) between the original codeword and the received (corrupted) word. Our matching argument utilizes a key feature of the SZ Insdel code. In particular, the Hamming weight (i.e., number of non-zero symbols) of every substring c' of an SZ codeword is at least $\lfloor |c'|/2 \rfloor$. This ensures that the buffers 0^t cannot be matched with large portions of any SZ codeword. We additionally leverage a key lemma (full version, Lemma 9) which states that the edit distance between the codeword $C_{\text{in}}(i, 1 - y_i)$ and *any* substring of length less than $2t$ of the uncorrupted codeword $C(x)$ has relative edit distance at least $\delta_{\text{in}}/2$. We use these two properties, along with key facts about the LCS matching, to yield an upper bound on the number of dangerous blocks, completing the analysis of our decoder.

1.2.3.5 Extension to relaxed locally correctable codes for insdel errors

Our construction also yields a strong Insdel Relaxed Locally Correctable Code (RLCC) with constant locality if the outer code is a weak Hamming RLCC. First, observe that bits of the codeword corresponding to the 0^t buffers are very easy to predict without even making

any queries to the corrupted codeword. Thus, if we are asked to recover the j'th bit of the codeword and j corresponds to a buffer 0^t, we can simply return 0 without making any queries to the received word. Otherwise, if we are asked to recover the j'th bit of the codeword and j corresponds to block c_i, we can simulate the Hamming RLCC decoder (as above) on input i to obtain y_i (or $\perp$). Assuming that $y_i \in \{0,1\}$, we can compute the corresponding SZ encoding of (i, y_i) and obtain the original value of the block c_i and then recover the j'th bit of the original codeword. The analysis of the RLCC decoder is analogous to the RLDC decoder. See Section 6 in the full version for details on both our weak Insdel RLDC and strong Insdel RLCC constructions.

▶ Remark 6. The "adaptiveness" of our constructed Insdel RLDC/RLCC decoder is identical to that of the outer Hamming RLDC/RLCC decoder. In particular, the weak Hamming RLDC of Ben-Sasson et al. [7] has a non-adaptive decoder, making our final decoder non-adaptive as well. Similarly, we use a weak Hamming RLCC due to Asadi and Shinkar [3] for our Insdel LCC, which is also a non-adaptive decoder.

2 Open Questions

Exact "phase-transition" thresholds

Our results show that both in the Hamming and Insdel setting there is a constant q such that every q-query RLDC requires super-polynomial codeword length, while there exists a $(q+1)$-query RLDC of polynomial codeword length. Finding the precise q remains an intriguing open question. Further, a more refined understanding of codeword length for RLDCs making $3, 4, 5$ queries is another important question, which has lead to much progress in the understanding of the LDC variants.

Constant-query strong Insdel RLDCs/RLCCs

While we do construct the first weak RLDCs in the Insdel setting, the drawback of our constructions is the fact that our codes do not satisfy the third property of Definition 1. Building strong Insdel RLDCs remains an open question. We note that our lower bounds imply that for a constant number of queries, such codes (if they exist) must have exponential codeword length.

Applications of local Insdel codes

As previously mentioned, Hamming LDCs/RLDCs have so far found many applications such as private information retrieval, probabilistically checkable proofs, self-correction, fault-tolerant circuits, hardness amplification, and data structures. Are there analogous or new applications of the Insdel variants in the broader computing area?

Lower bounds for Hamming RLDCs/LDCs

Our 2-query lower bound for Hamming RLDCs crucially uses the perfect completeness property of the decoder. An immediate question is whether the bound still holds if we allow the decoder to have imperfect completeness. We also note that the argument in our exponential lower bounds for 2-query Hamming RLDCs fail to hold for alphabets other than the binary alphabet, and we leave the extension to larger alphabet sizes as an open problem. Another related question is to understand if one can leverage perfect completeness and/or random restrictions to obtain improved lower bounds for $q \geq 3$-query standard Hamming LDCs. Perfect completeness has been explicitly used before to show exponential lower bounds for 2-query LCCs [11].

2.1 Further discussion about related work

Insdel codes

The study of error correcting codes for insertions and deletions was initiated by Levenstein [59]. While progress has been slow because constructing codes for insdel errors is strictly more challenging than for Hamming errors, strong interest in these codes lately has led to many exciting results [19, 21–25, 41–43, 45–49, 51, 61, 63, 69] (See also the excellent surveys of [50, 64, 66, 71]).

Insdel LDCs

[67] gave private-key constructions of LDCs with $m = \Theta(n)$ and locality polylog(n). [16] extended the construction from [67] to settings where the sender/decoder do not share randomness, but the adversarial channel is resource bounded. [12] applied the [13] compiler to the private key Hamming LDC of [67] (resp. resource bounded LDCs of [16]) to obtain private key Insdel LDCs (resp. resource bounded Insdel LDCs) with constant rate and polylog(n) locality.

Insdel LDCs have also been recently studied in *computationally bounded channels*, introduced in [60]. Such channels can perform a bounded number of adversarial errors, but do not have unlimited computational power as the general Hamming channels. Instead, such channels operate with bounded resources. As expected, in many such limited-resource settings one can construct codes with strictly better parameters than what can be done generally [31, 44, 65, 70]. LDCs in these channels under Hamming error were studied in [15, 16, 52–54, 67]. [12] applied the [13] compiler to the Hamming LDC of [16] to obtain a constant rate Insdel LDCs with polylog(n) locality for resource bounded channels. The work of [26] proposes the notion of locally decodable codes with randomized encoding, in both the Hamming and edit distance regimes, and in the setting where the channel is oblivious to the encoded message, or the encoder and decoder share randomness. For edit error they obtain codes with $m = O(n)$ or $m = n \log n$ and polylog(n) query complexity. However, even in settings with shared randomness or where the channel is oblivious or resource bounded, there are no known constructions of Insdel LDCs with constant locality.

Locality in the study of insdel codes was also considered in [49], which constructs explicit synchronization strings that can be locally decoded.

2.2 Organization

The remainder of the paper is organized as follows. We give general preliminaries and recall some prior results used in our results in Section 3. Due to space limit, we only present the proof of Theorem 2 in Section 4. The readers are pointed to the full version for proofs of Theorem 3, Theorem 4 and Corollary 5.

3 Preliminaries

For natural number $n \in \mathbb{N}$, we let $[n] \coloneqq \{1, 2, \ldots, n\}$. We let "$\circ$" denote the standard string concatenation operation. For a string $x \in \{0,1\}^*$ of finite length, we let $|x|$ denote the length of x. For $i \in [|x|]$, we let $x[i]$ denote the i-th bit of x. Furthermore, for $I \subseteq [|x|]$, we let $x[I]$ denote the subsequence $x[i_1] \circ x[i_2] \circ \cdots \circ x[i_\ell]$, where $i_j \in I$ and $\ell = |I|$. For two strings $x, y \in \{0,1\}^n$ of length n, we let $\mathsf{HAM}(x, y)$ denote the *Hamming Distance* between x and y; i.e., $\mathsf{HAM}(x, y) \coloneqq \big|\{i \in [n] \colon x_i \neq y_i\}\big|$. Similarly, we let $\mathsf{ED}(x, y)$ denote the *Edit*

Distance between x and y; i.e., $\mathsf{ED}(x, y)$ is the minimum number of insertions and deletions needed to transform string x into string y. We often discuss the *relative Hamming Distance* (resp., *relative Edit Distance*) between x and y, which is simply the Hamming Distance normalized by n, i.e., $\mathsf{HAM}(x, y)/n$ (resp., the Edit Distance normalized by $|x| + |y|$, i.e., $\mathsf{ED}(x, y)/(|x| + |y|)$). Finally, the *Hamming weight* of a string x is the number of non-zero entries of x, which we denote as $\mathsf{wt}(x) := |\{i \in [|x|] : x_i \neq 0\}|$.

For completeness, we recall the definition of a classical locally decodable code, or just a *locally decodable code*.

▶ **Definition 7** (Locally Decodable Codes). *A (q, δ, α)-Locally Decodable Code $C : \Sigma^n \to \Sigma^m$ is a code for which there exists a randomized decoder that makes at most q queries to the received word y and satisfies the following property: for every $i \in [n]$, if y is such that $\mathrm{dist}(y, C(x)) \leq \delta$ for some unique $C(x)$, then the decoder, on input i, outputs x_i with probability $\geq \alpha$. Here, the randomness is taken over the random coins of the decoder, and* dist *is a normalized metric.*

If dist *is the relative Hamming distance, then we say that the code is a Hamming LDC; similarly, if* dist *is the relative edit distance, then we say that the code is an Insdel LDC.*

We recall the general 2-query Hamming LDC lower bound [6, 56].

▶ **Theorem 8** ([6, 56]). *For constants $\delta, \varepsilon \in (0, 1/2)$ there exists a constant $c = c(\delta, \varepsilon) \in (0, 1)$ such that if $C : \{0,1\}^n \to \{0,1\}^m$ is a $(2, \delta, 1/2 + \varepsilon)$ Hamming LDC then $m \geq 2^{cn-1}$.*

In our weak Insdel RLDC construction, we utilize a weak Hamming RLDC due to [7].

▶ **Lemma 9** ([7]). *For constants $\varepsilon, \delta \in (0, 1/2)$ and $\gamma \in (0, 1)$, there exists a constant $q = O_{\delta,\varepsilon}(1/\gamma^2)$ and a weak $(q, \delta, 1/2 + \varepsilon)$-Hamming RLDC $C : \{0,1\}^n \to \{0,1\}^m$ with $m = O(n^{1+\gamma})$. Moreover, the decoder of this code is non-adaptive.*

Our construction additionally utilizes the well-known Schulman-Zuckerman Insdel codes [69].

▶ **Lemma 10** (Schulman-Zuckerman (SZ) Code [69]). *There exists constants $\beta \geq 1$ and $\delta > 0$ such that for large enough values of $t > 0$, there exists a code $C : \{0,1\}^t \to \{0,1\}^{\beta t}$ capable of decoding from δ-fraction of Insdel errors and the additional property that for every $x \in \{0,1\}^t$ and $y = C(x)$, every substring y' of y with length at least 2 has Hamming weight $\geq \lfloor |y'|/2 \rfloor$.*

Our strong Insdel RLCC construction relies on a weak Hamming RLCC. We utilize the following weak Hamming RLCC implicit in [3].

▶ **Lemma 11** (Implied by Theorem 1 of [3]). *For every sufficiently large $q \in \mathbb{N}$ and $\varepsilon \in (0, 1/2)$, there is a constant δ such that there exists a weak $(q, \delta, 1/2 + \varepsilon)$-relaxed Hamming Locally Correctable Code $C : \{0,1\}^n \to \{0,1\}^m$ with $m = n^{1+O(1/q)}$. Moreover, the decoder of this code is non-adaptive.*

4 Lower Bounds for 2-Query Hamming RLDCs

We prove Theorem 2 in this section. As a reminder, a weak (q, δ, α)-RLDC satisfies the first two conditions in Definition 1, and non-adaptive means the decoder makes queries according to a distribution which is independent of the received string y. Here we are interested in the case $q = 2$ and $\alpha = 1/2 + \varepsilon$.

To avoid overloading first-time readers with heavy notations, we first present a proof of the lower bound for *non-adaptive* decoders, i.e., decoders with a query distribution independent of the received string. This proof will be easier to follow, while the crucial ideas behind it remain the same. The proof for the most general case is presented in the last subsection, with an emphasis on the nuances in dealing with adaptivity.

4.1 A Warm-up: the lower bound for non-adaptive decoders

In the following, we fix a relaxed decoder Dec for C. In this subsection, we assume that Dec is non-adaptive, and that it has the first two properties specified in Definition 1. To avoid technical details, we also assume Dec always makes exactly 2 queries (otherwise add dummy queries to make the query count exactly 2).

Given an index $i \in [n]$ and queries j, k made by $\mathsf{Dec}(i, \cdot)$, in the most general setting the output could be a random variable which depends on i and y_j, y_k, where y_j, y_k are the answers to queries j, k, respectively. An equivalent view is that the decoder picks a random function f according to some distribution and outputs $f(y_j, y_k)$. Let $\mathtt{DF}^i_{j,k}$ be the set of all decoding functions $f\colon \{0,1\}^2 \to \{0,1,\perp\}$ which are selected by $\mathsf{Dec}(i,\cdot)$ with non-zero probability when querying j, k. We partition the queries into the following two sets

$$F^0_i \coloneqq \left\{\{j,k\} \subseteq [m]\colon \forall f \in \mathtt{DF}^i_{j,k} \text{ the truth table of } f \text{ contains no "}\perp\text{"}\right\},$$
$$F^{\geq 1}_i \coloneqq \left\{\{j,k\} \subseteq [m]\colon \exists f \in \mathtt{DF}^i_{j,k} \text{ the truth table of } f \text{ contains at least 1 "}\perp\text{"}\right\}.$$

Notations

Given a string $w \in \{0,1\}^m$ and a subset $S \subseteq [m]$, we denote $w[S] \coloneqq (w_i)_{i\in S} \in \{0,1\}^{|S|}$. Given a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, and $\sigma \in \{0,1\}$, we write $f \restriction_{x_i=\sigma}$ to denote the restriction of f to the domain $\left\{\mathbf{x} \in \{0,1\}^n : x_i = \sigma\right\}$. For a sequence of restrictions, we simply write $f \restriction_{(x_{j_1},\ldots,x_{j_k})=(\sigma_1,\ldots,\sigma_k)}$, or $f_{J|\sigma}$ where $J = [n] \setminus \{j_1, \ldots, j_k\}$ and $\sigma = (\sigma_1, \ldots, \sigma_k)$. Note that $f_{J|\sigma}$ is a Boolean function over the domain $\{0,1\}^J$.

We will identify the encoding function of C as a collection of m Boolean functions

$$\mathcal{C} \coloneqq \left\{C_1, \ldots, C_m\colon \forall j \in [m], C_j\colon \{0,1\}^n \to \{0,1\}\right\}.$$

Namely, $C(x) = (C_1(x), C_2(x), \ldots, C_m(x))$ for all $x \in \{0,1\}^n$.

For $j \in [m]$, we say C_j is *fixable* by x_i if at least one of the restrictions $C_j \restriction_{x_i=0}$ and $C_j \restriction_{x_i=1}$ is a constant function. Denote

$$S_i \coloneqq \left\{j \in [m]\colon C_j \text{ is fixable by } x_i\right\}, \quad T_j \coloneqq \left\{i \in [n]\colon C_j \text{ is fixable by } x_i\right\},$$

and $w_j \coloneqq |T_j|$. Let

$$W \coloneqq \left\{j \in [m]\colon w_j \geq 3\ln(8/\delta)\right\}.$$

For $i \in [n]$ define the sets $S_{i,+} \coloneqq S_i \cap W$, and $S_{i,-} \coloneqq S_i \cap \overline{W}$.

Let $J \subseteq [n]$ and $\rho \in \{0,1\}^{\overline{J}}$. A code $C\colon \{0,1\}^n \to \{0,1\}^m$ restricted to $\mathbf{x}_{\overline{J}} = \rho$, denoted by $C_{J|\rho}$, is specified by the following collection of Boolean functions

$$\mathcal{C}_{J|\rho} \coloneqq \left\{C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho}\colon j \in [m], C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho} \text{ is not a constant function}\right\}.$$

Namely, we restrict each function C_j in $\mathcal{C}$ to $\mathbf{x}_{\overline{J}} = \rho$, and eliminate those that have become constant functions. $C_{J|\rho}$ encodes n'-bit messages into m'-bit codewords, where $n' = |J|$ and $m' = \left|\mathcal{C}_{J|\rho}\right| \leq m$.

We note that the local decoder Dec for C can also be used as a local decoder for $C_{J|\rho}$, while preserving all the parameters. This is because, Dec never needs to really read a codeword bit which has become a constant function under the restriction $J|\rho$.

The lemma below will be useful later in the proof. It shows that a constant fraction of the message bits can be fixed so that most codeword bits C_j with large w_j become constants.

▶ **Lemma 12.** *There exist a set $J \subseteq [n]$ and assignments $\rho \in \{0,1\}^{\overline{J}}$ such that $|J| \geq n/6$, and $|W \setminus A| \leq \delta m/4$, where $A \subseteq W$ collects all codeword bits $j \in W$ such that $C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho}$ is a constant function.*

Proof. Let J be a random subset formed by selecting each $i \in [n]$ independently with probability $1/3$. For each $j \in \overline{J}$, set $\rho_j = 0$ or $\rho_j = 1$ with probability $1/2$. We have $\mathbb{E}[|J|] = n/3$, and hence the Chernoff bound shows that $|J| < n/6$ with probability $\exp(-\Omega(n))$. Furthermore, for each $j \in W$, $C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho}$ becomes a constant function except with probability $\delta/8$. This is because for each $i \in T_j$, $C_j \restriction_{x_i=0}$ or $C_j \restriction_{x_i=1}$ is a constant function, and either case happens with probability $1/3$. Therefore

$$\Pr\left[C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho} \text{ is not constant}\right] \leq \left(1 - \frac{1}{3}\right)^{|T_j|} < e^{-|T_j|/3} \leq \frac{\delta}{8},$$

where the last inequality is due to $w_j = |T_j| \geq 3\ln(8/\delta)$, since $j \in W$.

By linearity of expectation and Markov's inequality, we have

$$\begin{aligned}
&\Pr\left[\sum_{j\in W} \mathbf{1}\left\{C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho} \text{ is not constant}\right\} \geq \frac{\delta}{4}|W|\right] \\
\leq &\frac{\mathbb{E}\left[\sum_{j\in W} \mathbf{1}\left\{C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho} \text{ is not constant}\right\}\right]}{\delta|W|/4} \\
= &\frac{\sum_{j\in W} \Pr\left[C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho} \text{ is not constant}\right]}{\delta|W|/4} \\
\leq &\frac{\delta/8 \cdot |W|}{\delta|W|/4} \leq \frac{1}{2}.
\end{aligned}$$

Applying a union bound gives

$$\begin{aligned}
&\Pr\left[(|J| < n/6) \vee \left(\sum_{j\in W} \mathbf{1}\left\{C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho} \text{ is not constant}\right\} \geq \frac{\delta}{4}|W|\right)\right] \\
\leq &\exp\left(-\Omega(n)\right) + \frac{1}{2} < 1.
\end{aligned}$$

Finally, we can conclude that there exist $J \subseteq [n]$ and $\rho \in \{0,1\}^{\overline{J}}$ such that $|J| \geq n/6$, and $C_j \restriction_{\mathbf{x}_{\overline{J}}=\rho}$ becomes a constant function for all but $\delta/4$ fraction of $j \in W$. ◀

Let $J \subseteq [n]$ and $\rho \in \{0,1\}^{\overline{J}}$ be given by the Lemma 12, and consider the restricted code $C_{J|\rho}$. By rearranging the codeword bits, we may assume $J = [n']$ where $n' = |J| \geq n/6$.

Let $A \subseteq [m]$ be the set of codeword bits which get fixed to constants under $J|\rho$. We denote $W' \coloneqq W \setminus A$, $S'_i \coloneqq S_i \setminus A$, $S'_{i,-} \coloneqq S_{i,-} \setminus A$, and $S'_{i,+} \coloneqq S_{i,+} \setminus A$. Note that $|W'| = |W \setminus A| \leq \delta m/4$, and thus $|S'_{i,+}| = |S_{i,+} \cap W'| \leq \delta m/4$ for all $i \in [n']$. We emphasize that S'_i does not necessarily contain all codeword bits fixable by x_i in the restricted code $C_{J|\rho}$, as fixing some message bits may cause more codeword bits to be fixable by x_i.

We first show that the queries of C must have certain structures. The following claim characterizes the queries in $F_i^{\geq 1}$.

▷ Claim 13. Suppose $\{j,k\} \in F_i^{\geq 1}$. Then we must have $j,k \in S_i$.

Proof. Let $\{j,k\} \in F_i^{\geq 1}$. Suppose for the sake of contradiction that $j \notin S_i$. This implies there are partial assignments $\sigma_{00}, \sigma_{01}, \sigma_{10}, \sigma_{11} \in \{0,1\}^{n-1}$ such that

$$C_j\left(\mathbf{x}_{-i} = \sigma_{00}, x_i = 0\right) = 0, \quad C_j\left(\mathbf{x}_{-i} = \sigma_{01}, x_i = 1\right) = 0,$$
$$C_j\left(\mathbf{x}_{-i} = \sigma_{10}, x_i = 0\right) = 1, \quad C_j\left(\mathbf{x}_{-i} = \sigma_{11}, x_i = 1\right) = 1,$$

where $\mathbf{x}_{-i}$ is defined as $\left(x_t \colon t \in [n] \setminus \{i\}\right)$.

Let $C_{00}, C_{01}, C_{10}, C_{11}$ be encodings of the corresponding assignments mentioned above. Since the relaxed decoder has perfect completeness, when $\mathsf{Dec}(i,\cdot)$ is given access to C_{00} or C_{10} it must output $x_i = 0$. Note that the j-th bit is different in C_{00} and C_{10}. Similarly, when $\mathsf{Dec}(i,\cdot)$ is given access to C_{01} or C_{11} it must output $x_i = 1$. However, this already takes up 4 entries in the truth table of any decoding function $f \in \mathtt{DF}^i_{j,k}$, leaving no space for any "$\perp$" entry. This contradicts with the assumption $\{j,k\} \in F_i^{\geq 1}$. ◁

Here is another way to view Claim 13 which will be useful later: Suppose $\{j,k\}$ is a query set such that $j \notin S_i$ (or $k \notin S_i$), then $\{j,k\} \in F_i^0$. In other words, conditioned on the event that some query is not contained in S_i, the decoder never outputs $\perp$.

The following claim characterizes the queries in F_i^0.

▷ Claim 14. Suppose $\{j,k\} \in F_i^0$, and $j \in S_i$. Then one of the following three cases occur: (1) $k \in S_i$, (2) $C_j = x_i$, or (3) $C_j = \neg x_i$.

Proof. Since $j \in S_i$, we may, without loss of generality, assume that $C_j \restriction_{x_i=0}$ is a constant function. Let us further assume it is the constant-zero function. The proofs for the other cases are going to be similar.

Denote by $f(y_j, y_k)$ the function returned by $\mathsf{Dec}(i,\cdot)$ conditioned on reading $\{j,k\}$. Any function $f \in \mathtt{DF}^i_{j,k}$ takes values in $\{0,1\}$ since $\{j,k\} \in F_i^0$. Suppose case (1) does not occur, meaning that $C_k \restriction_{x_i=0}$ is not a constant function. Then there must be partial assignments $\sigma_{00}, \sigma_{01} \in \{0,1\}^{n-1}$ such that

$$C_k(x_i = 0, \mathbf{x}_{-i} = \sigma_{00}) = 0, \quad C_k(x_i = 0, \mathbf{x}_{-i} = \sigma_{01}) = 1.$$

Let C_{00} and C_{01} be the encodings of the corresponding assignments mentioned above. Due to perfect completeness of Dec, it must always output $x_i = 0$ when given access to C_{00} or C_{01}. That means $f(0,0) = f(0,1) = 0$.

Now we claim that $C_j \restriction_{x_i=1}$ must be the constant-one function. Otherwise there is a partial assignment $\sigma_{10} \in \{0,1\}^{n-1}$ such that

$$C_j(x_i = 1, \mathbf{x}_{-i} = \sigma_{10}) = 0.$$

Let C_{10} be the encoding of this assignment. On the one hand, due to perfect completeness $\mathsf{Dec}(i,\cdot)$ should always output $x_i = 1$ when given access to C_{10}. On the other hand, $\mathsf{Dec}(i,\cdot)$ outputs $f((C_{10})_j, 0) = f(0,0) = 0$. This contradiction shows that $C_j \restriction_{x_i=1}$ must be the constant-one function. Therefore $C_j = x_i$, i.e., case (2) occurs.

Similarly, when $C_j \restriction_{x_i=0}$ is the constant-one function, we can deduce that $C_j = \neg x_i$, i.e., case (3) occurs. ◁

We remark that Claim 13 and Claim 14 jointly show that for any query set $\{j,k\}$ made by $\mathsf{Dec}(i,\cdot)$ there are 2 essentially different cases: (1) both j,k lie inside S_i, and (2) both j,k lie outside S_i. The case $j \in S_i, k \notin S_i$ ($k \in S_i, j \notin S_i$, resp.) means that k (j, resp.) is a dummy query which is not used for decoding. Furthermore, conditioned on case (2), the decoder never outputs $\perp$.

Another important observation is that all properties of the decoder discussed above hold for the restricted code $C_{J|\rho}$, with S_i replaced by S'_i. This is because $C_{J|\rho}$ uses essentially the same decoder, except that it does not actually query any codeword bit which became a constant.

For a subset $S \subseteq [m]$, we say "$\mathsf{Dec}(i,\cdot)$ reads S" if the event "$j \in S$ and $k \in S$" occurs where $j, k \in [m]$ are the queries made by $\mathsf{Dec}(i,\cdot)$. The following lemma says that conditioned on $\mathsf{Dec}(i,\cdot)$ reads some subset S, there is a way of modifying the bits in S that flips the output of the decoder.

▶ **Lemma 15.** *Let $S \subseteq [m]$ be a subset such that $\Pr[\mathsf{Dec}(i,\cdot) \text{ reads } S] > 0$. Then for any string $s \in \{0,1\}^m$ and any bit $b \in \{0,1\}$, there exists a string $z \in \{0,1\}^m$ such that $z[[m] \setminus S] = s[[m] \setminus S]$, and*

$$\Pr\left[\mathsf{Dec}(i,z) = 1 - b \mid \mathsf{Dec}(i,\cdot) \text{ reads } S\right] = 1.$$

Proof. Let $x \in \{0,1\}^n$ be a string with $x_i = 1 - b$. Let $z \in \{0,1\}^m$ be the string satisfying

$$z[S] = C(x)[S], \quad z[[m] \setminus S] = s[[m] \setminus S].$$

Since Dec has perfect completeness, we have

$$1 = \Pr\left[\mathsf{Dec}(i, C(x)) = x_i \mid \mathsf{Dec}(i,\cdot) \text{ reads } S\right] = \Pr\left[\mathsf{Dec}(i,z) = 1 - b \mid \mathsf{Dec}(i,\cdot) \text{ reads } S\right].$$

◀

The next lemma is a key step in our proof. It roughly says that there is a local decoder for x_i in the standard sense as long as the size of S_i is not too large.

▶ **Lemma 16.** *Suppose $i \in [n]$ is such that $|S_i| \le \delta m/2$. Then there is a $(2, \delta/2, 1/2 + \varepsilon)$-local decoder D_i for i. In other words, for any $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$ such that $\mathsf{HAM}(C(x), y) \le \delta m/2$, we have*

$$\Pr\left[D_i(y) = x_i\right] \ge \frac{1}{2} + \varepsilon,$$

and D_i makes at most 2 queries into y.

Proof. Let $i \in [n]$ be such that $|S_i| \le \delta m/2$. The local decoder D_i works as follows. Given $x \in \{0,1\}^n$ and $y \in \{0,1\}^m$ such that $\mathsf{HAM}(C(x), y) \le \delta m/2$, D_i obtains a query set Q according to the query distribution of $\mathsf{Dec}(i,\cdot)$ conditioned on $Q \subseteq [m] \setminus S_i$. Then D_i finishes by outputting the result returned by $\mathsf{Dec}(i,\cdot)$.

Denote by E_i the event "$\mathsf{Dec}(i,\cdot)$ reads $[m] \setminus S_i$", i.e., both two queries made by $\mathsf{Dec}(i,\cdot)$ lie outside S_i. In order for the conditional distribution to be well-defined, we need to argue that E_i occurs with non-zero probability. Suppose this is not the case, meaning that $Q \cap S_i \neq \varnothing$ for all possible query set Q. Let $z \in \{0,1\}^m$ be the string obtained by applying Lemma 15 with $S = S_i$, $s = C(x)$ and $b = x_i$. Claim 13 and Claim 14 jointly show that either $Q \subseteq S_i$, or the decoder's output does not depend on the answers to queries in $Q \setminus S_i$. In any case, the output of $\mathsf{Dec}(i,z)$ depends only on $z[S_i]$. However, by the choice of z we now have a contradiction since

$$\frac{1}{2} + \varepsilon \le \Pr\left[\mathsf{Dec}(i,z) \in \{x_i, \perp\}\right] = \Pr\left[\mathsf{Dec}(i,z) \in \{x_i, \perp\} \mid \mathsf{Dec}(i,\cdot) \text{ reads } S_i\right] = 0,$$

where the first inequality is due to $\mathsf{HAM}(C(x), z) \le |S_i| < \delta m$ and the relaxed decoding property of Dec.

By definition of D_i, it makes at most 2 queries into y. Its success rate is given by

$$\Pr[D_i(y) = x_i] = \Pr[\mathsf{Dec}(i, y) = x_i \mid E_i].$$

Therefore it remains to show that

$$\Pr\left[\mathsf{Dec}(i, y) = x_i \mid E_i\right] \geq \frac{1}{2} + \varepsilon.$$

Let z be the string obtained by applying Lemma 15 with $S = S_i$, $s = y$ and $b = x_i$. From previous discussions we see that conditioned on $\overline{E_i}$ (i.e., the event E_i does not occur), the output of $\mathsf{Dec}(i, z)$ only depends on $z[S_i]$. Therefore

$$\Pr\left[\mathsf{Dec}(i, z) \in \{x_i, \perp\} \mid \overline{E_i}\right] = 1 - \Pr\left[\mathsf{Dec}(i, z) = 1 - x_i \mid \overline{E_i}\right] = 0. \tag{1}$$

We also have that z is close to $C(x)$ since

$$\mathsf{HAM}(z, C(x)) \leq \mathsf{HAM}(z, y) + \mathsf{HAM}(y, C(x)) \leq |S_i| + \delta m/2 \leq \delta m.$$

Thus, the relaxed decoding property of Dec gives

$$\Pr\left[\mathsf{Dec}(i, z) \in \{x_i, \perp\}\right] \geq \frac{1}{2} + \varepsilon.$$

On the other hand, we also have

$$\begin{aligned}
&\Pr\left[\mathsf{Dec}(i, z) \in \{x_i, \perp\}\right] \\
=&\Pr\left[\mathsf{Dec}(i, z) \in \{x_i, \perp\} \mid \overline{E_i}\right] \cdot \Pr\left[\overline{E_i}\right] + \Pr\left[\mathsf{Dec}(i, z) \in \{x_i, \perp\} \mid E_i\right] \cdot \Pr\left[E_i\right] \\
=&\Pr\left[\mathsf{Dec}(i, z) \in \{x_i, \perp\} \mid \overline{E_i}\right] \cdot \Pr\left[\overline{E_i}\right] + \Pr\left[\mathsf{Dec}(i, y) \in \{x_i, \perp\} \mid E_i\right] \cdot \Pr\left[E_i\right] \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (z[[m] \setminus S_i] = y[[m] \setminus S_i]) \\
=&\Pr\left[\mathsf{Dec}(i, y) \in \{x_i, \perp\} \mid E_i\right] \cdot \Pr\left[E_i\right] \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(Equation (1))} \\
\leq&\Pr\left[\mathsf{Dec}(i, y) \in \{x_i, \perp\} \mid E_i\right].
\end{aligned}$$

Note that by Claim 13, conditioned on E_i, $\mathsf{Dec}(i, \cdot)$ never outputs "$\perp$". We thus have

$$\Pr\left[\mathsf{Dec}(i, y) = x_i \mid E_i\right] \geq \frac{1}{2} + \varepsilon. \qquad \blacktriangleleft$$

We remark once again that the above lemma holds for the restricted code $C_{J|\rho}$, with S_i replaced by S'_i.

Below we prove an exponential lower bound for non-adaptive 2-query Hamming RLDCs.

▶ **Proposition 17.** *Let $C\colon \{0,1\}^n \to \{0,1\}^m$ be a non-adaptive weak $(2, \delta, 1/2 + \varepsilon)$-RLDC. Then $m = 2^{\Omega_{\delta,\varepsilon}(n)}$.*

Proof. Let $C_{J|\rho}\colon \{0,1\}^{n'} \to \{0,1\}^{m'}$ be the restricted code where $J|\rho$ is given by Lemma 12, and $A \subseteq [m]$ be the set of codeword bits which get fixed to constants. We also let $S'_i \coloneqq S_i \setminus A$, $S'_{i,-} = S_{i,-} \setminus A$, $S'_{i,+} = S_{i,+} \setminus A$.

Denote $T'_j \coloneqq \left\{i \in [n']\colon j \in S'_i\right\}$. Since $S'_i \subseteq S_i$ for each i, we also have $T'_j \subseteq T_j$ for each j. In particular, for each $j \notin W' \subseteq W$, we have $|T'_j| \leq |T_j| \leq 3\ln(8/\delta)$. Therefore

$$\mathop{\mathbb{E}}_{i \in [n']}\left[|S'_{i,-}|\right] = \frac{1}{n'}\sum_{i=1}^{n'} |S'_{i,-}| = \frac{1}{n'}\sum_{j \in [m'] \setminus W'} |T'_j| \leq 3\ln(8/\delta) \cdot \frac{m'}{n'}.$$

Therefore by Markov's inequality,

$$\Pr_{i\in[n']}\left[|S'_{i,-}| > \delta m'/4\right] \leq \frac{12\ln(8/\delta)}{\delta n'} = O_\delta\left(\frac{1}{n'}\right).$$

In other words, there exists $I \subseteq [n']$ of size $|I| \geq n' - O_\delta(1)$ such that $|S'_{i,-}| \leq \delta m'/4$ for all $i \in I$. For any such $i \in I$, we have $|S'_i| = |S'_{i,-}| + |S'_{i,+}| \leq \delta m'/4 + \delta m'/4 = \delta m'/2$. By Lemma 16, we can view $C_{J|\rho}$ as a $(2, \delta/2, 1/2+\varepsilon)$-LDC for message bits in I (for instance, we can arbitrarily fix the message bits outside I), where $|I| > n' - O_\delta(1) = \Omega(n)$. Finally, the statement of the proposition follows from Theorem 8. ◀

4.2 Lower bounds for adaptive 2-Query Hamming RLDCs

Now we turn to the actual proof, which still works for possibly adaptive decoders. Let C be a weak $(2, \delta, 1/2+\varepsilon)$-RLDC with perfect completeness. We fix a relaxed decoder Dec for C. Without loss of generality, we assume Dec works as follows: on input $i \in [n]$, $\mathsf{Dec}(i,\cdot)$ picks the first query $j \in [m]$ according to a distribution $\mathcal{D}_i$. Let $b \in \{0,1\}$ be the answer to this query. Then Dec picks the second query $k \in [m]$ according to a distribution $\mathcal{D}_{i;j,b}$, and obtains an answer $b' \in \{0,1\}$. Finally, Dec outputs a random variable $X_{i;j,b,k,b'} \in \{0,1,\perp\}$.

We partition the support of $\mathcal{D}_i$ into the following two sets:

$$F_i^0 := \left\{j \in \mathrm{supp}(\mathcal{D}_i) \colon \forall b,b' \in \{0,1\}, k \in \mathrm{supp}(\mathcal{D}_{i;j,b,k,b'}), \Pr[X_{i;j,b,k,b'} = \perp] = 0\right\},$$
$$F_i^{>0} := \left\{j \in \mathrm{supp}(\mathcal{D}_i) \colon \exists b,b' \in \{0,1\}, k \in \mathrm{supp}(\mathcal{D}_{i;j,b,k,b'}), \Pr[X_{i;j,b,k,b'} = \perp] > 0\right\}.$$

We will still apply the restriction guaranteed by Lemma 12 to C. The sets S_i, T_j, W, $S_{i,-}$, $S_{i,+}$ (are their counterparts for $C_{J|\rho}$) are defined in the exact same way.

The following claim is adapted from Claim 13.

▷ Claim 18. $(\mathrm{supp}(\mathcal{D}_i) \setminus S_i) \subseteq F_i^0$.

Proof. Let $j \in \mathrm{supp}(\mathcal{D}_i) \setminus S_i$ and we will show $j \in F_i^0$. By the definition of S_i, $j \notin S_i$ means that there are partial assignments $\sigma_{00}, \sigma_{01}, \sigma_{10}, \sigma_{11} \in \{0,1\}^{n-1}$ such that

$$C_j(\mathbf{x}_{-i} = \sigma_{00}, x_i = 0) = 0, \quad C_j(\mathbf{x}_{-i} = \sigma_{01}, x_i = 1) = 0,$$
$$C_j(\mathbf{x}_{-i} = \sigma_{10}, x_i = 0) = 1, \quad C_j(\mathbf{x}_{-i} = \sigma_{11}, x_i = 1) = 1,$$

where $\mathbf{x}_{-i}$ is defined as $\left(x_t \colon t \in [n] \setminus \{i\}\right)$.

Let $C_{00}, C_{01}, C_{10}, C_{11}$ be encodings of the corresponding assignments mentioned above. Consider an arbitrary query $k \in \mathrm{supp}(\mathcal{D}_{i;j,0})$, and let b'_1, b'_2 be the k-th bit of C_{00} and C_{01}, respectively. We note that $X_{i;j,0,k,b'_1}$ is the output of $\mathsf{Dec}(i, C_{00})$ conditioned on the queries j, k, and $X_{i;j,0,k,b'_2}$ is the output of $\mathsf{Dec}(i, C_{01})$ conditioned on the queries j, k. Due to perfect completeness of Dec, we have

$$\Pr[X_{i;j,0,k,b'_1} = 0] = 1, \quad \Pr[X_{i;j,0,k,b'_2} = 1] = 1.$$

Therefore, it must be the case that $b'_1 \neq b'_2$, which implies that $\Pr[X_{i;j,0,k,b'} = \perp] = 0$ for any $b' \in \{0,1\}$.

An identical argument shows that $\Pr[X_{i;j,1,k,b'} = \perp] = 0$ for any $k \in \mathrm{supp}(\mathcal{D}_{i;j,1})$ and $b' \in \{0,1\}$. Thus we have shown $j \in F_i^0$. ◁

We remark that the above claim also implies $F_i^{>0} \subseteq S_i$, since $\mathrm{supp}(\mathcal{D}_i)$ is a disjoint union of F_i^0 and $F_i^{>0}$. In other words, conditioned on the event that the first query j is not contained in S_i, the decoder never outputs $\perp$.

The next claim is adapted from Claim 14.

▷ Claim 19. Let $j \in \mathrm{supp}(\mathcal{D}_i) \cap S_i$. For any $b \in \{0,1\}$ one of the following three cases occurs:

1. $\mathrm{supp}(\mathcal{D}_{i;j,b}) \subseteq S_i$;
2. For any $k \in \mathrm{supp}(\mathcal{D}_{i;j,b}) \setminus S_i$, $\Pr[X_{i;j,b,k,0} = b] = \Pr[X_{i;j,b,k,1} = b] = 1$;
3. For any $k \in \mathrm{supp}(\mathcal{D}_{i;j,b}) \setminus S_i$, $\Pr[X_{i;j,b,k,0} = 1-b] = \Pr[X_{i;j,b,k,1} = 1-b] = 1$.

Proof. Since $j \in S_i$, we may, without loss of generality, assume that $C_j \restriction_{x_i=0}$ is a constant function. Let us further assume $C_j \restriction_{x_i=0} \equiv 0$. The proofs for the other cases are going to be similar.

Suppose $\mathrm{supp}(\mathcal{D}_{i;j,0}) \not\subseteq S_i$, and let $k \in \mathrm{supp}(\mathcal{D}_{i;j,0}) \setminus S_i$. By the definition of S_i, $k \notin S_i$ means that there are partial assignments $\sigma_{00}, \sigma_{01} \in \{0,1\}^{n-1}$ such that

$$C_k(x_i = 0, \mathbf{x}_{-i} = \sigma_{00}) = 0, \quad C_k(x_i = 0, \mathbf{x}_{-i} = \sigma_{01}) = 1.$$

Let C_{00} and C_{01} be the encodings of the corresponding assignments mentioned above. We note that $X_{i;j,0,k,0}$ and $X_{i;j,0,k,1}$ are the outputs of $\mathsf{Dec}(i, C_{00})$ and $\mathsf{Dec}(i, C_{01})$, respectively, conditioned on the queries j, k. Due to perfect completeness of Dec, we must have

$$\Pr[X_{i;j,0,k,0} = 0] = \Pr[X_{i;j,0,k,1} = 0] = 1,$$

since both C_{00} and C_{01} encode messages with $x_i = 0$.

Now we claim that $C_j \restriction_{x_i=1} \equiv 1$ must hold. Otherwise there is a partial assignment $\sigma_{10} \in \{0,1\}^{n-1}$ such that

$$C_j(x_i = 1, \mathbf{x}_{-i} = \sigma_{10}) = 0.$$

Let C_{10} be the encoding of this assignment, and let $b' \in \{0,1\}$ be the k-th bit of C_{10}. On the one hand, $X_{i;j,0,k,b'}$ is the output $\mathsf{Dec}(i, C_{10})$ conditioned on the queries j, k, and we have just established

$$\Pr[X_{i;j,0,k,b'} = 0] = 1.$$

On the other hand, $\mathsf{Dec}(i, C_{10})$ should output $x_i = 1$ with probability 1 due to perfect completeness. This contradiction shows that $C_j \restriction_{x_i=1} \equiv 1$.

Similarly, suppose $\mathrm{supp}(\mathcal{D}_{i;j,1}) \not\subseteq S_i$ and let $k \in \mathrm{supp}(\mathcal{D}_{i;j,1}) \setminus S_i$, meaning that there are partial assignments $\sigma_{10}, \sigma_{11} \in \{0,1\}^{n-1}$ such that

$$C_k(x_i = 1, \mathbf{x}_{-i} = \sigma_{10}) = 0, \quad C_k(x_i = 1, \mathbf{x}_{-i} = \sigma_{11}) = 1.$$

Let C_{10} and C_{11} be the corresponding encodings, and note that $X_{i;j,1,k,0}$ and $X_{i;j,1,k,1}$ are the outputs of $\mathsf{Dec}(i, C_{10})$ and $\mathsf{Dec}(i, C_{11})$, respectively, conditioned on the queries j, k. Perfect completeness of Dec implies

$$\Pr[X_{i;j,1,k,0} = 1] = \Pr[X_{i;j,1,k,1} = 1] = 1,$$

since both C_{10} and C_{11} encode messages with $x_i = 1$.

So far we have shown that for any $b \in \{0,1\}$ such that $\mathrm{supp}(\mathcal{D}_{i;j,b}) \not\subseteq S_i$, it holds that

$$\forall k \in \mathrm{supp}(\mathcal{D}_{i;j,b}) \setminus S_i, \quad \Pr[X_{i;j,b,k,0} = b] = \Pr[X_{i;j,b,k,1} = b] = 1,$$

provided that $C_j \restriction_{x_i=0} \equiv 0$. In case of $C_j \restriction_{x_i=0} \equiv 1$, we can use an identical argument to deduce that for any $b \in \{0,1\}$ such that $\mathrm{supp}(\mathcal{D}_{i;j,b}) \not\subseteq S_i$, it holds that

$$\forall k \in \mathrm{supp}(\mathcal{D}_{i;j,b}) \setminus S_i, \quad \Pr[X_{i;j,b,k,0} = 1-b] = \Pr[X_{i;j,b,k,1} = 1-b] = 1. \quad \lhd$$

Here is another way to view Claim 19: conditioned on the event that the first query j is contained in S_i, either the second query k is also contained in S_i, or the output $X_{i;j,b,k,b'}$ is independent of the answer b' to query k. In either case, the decoder's output depends solely on the S_i-portion of the received string.

Once again, the conclusions of Claim 18 and Claim 19 hold for $C_{J|\rho}$, with S_i replaced by S'_i.

Finally, we are ready to prove Theorem 2. We recall the Theorem below.

▶ **Theorem 2.** *Let* $C\colon \{0,1\}^n \to \{0,1\}^m$ *be a weak adaptive* $(2,\delta,1/2+\varepsilon)$*-RLDC. Then* $m = 2^{\Omega_{\delta,\varepsilon}(n)}$.

Proof. The proof is almost identical to the one for Proposition 17. First, we can show that there exists $I \subseteq [n']$ of size $|I| \geq n' - O_\delta(1) = \Omega(n)$ such that $|S'_{i,-}| \leq \delta m/4$ for all $i \in I$, and hence $|S'_i| = |S'_{i,-}| + |S'_{i,+}| \leq \delta m/2$. Second, similar to the proof of Lemma 16, for each $i \in I$ we can construct a decoder D_i for x_i as follows. D_i restarts $\mathsf{Dec}(i,\cdot)$ until it makes a first query $j \in [m'] \setminus S'_i$. Then D_i finishes simulating $\mathsf{Dec}(i,\cdot)$ and returns its output. With the help of Claim 18 and Claim 19, the same analysis in Lemma 16 shows that D_i never returns $\perp$, and that the probability of returning x_i is at least $1/2+\varepsilon$. Finally, the theorem follows from Theorem 8. ◀

References

1 Omar Alrabiah, Venkatesan Guruswami, Pravesh Kothari, and Peter Manohar. A near-cubic lower bound for 3-query locally decodable codes from semirandom csp refutation. *Electron. Colloquium Comput. Complex.*, 2022. URL: `https://eccc.weizmann.ac.il/report/2022/101/`.

2 Alexandr Andoni, Thijs Laarhoven, Ilya P. Razenshteyn, and Erik Waingarten. Optimal hashing-based time-space trade-offs for approximate near neighbors. In *SODA*, pages 47–66, 2017.

3 Vahid R. Asadi and Igor Shinkar. Relaxed locally correctable codes with improved parameters. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021*, volume 198 of *LIPIcs*, pages 18:1–18:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021.

4 László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking computations in polylogarithmic time. In *STOC*, pages 21–31, 1991.

5 James L. Banal, Tyson R. Shepherd, Joseph Berleant, Hellen Huang, Miguel Reyes, Cheri M. Ackerman, Paul C. Blainey, and Mark Bathe. Random access dna memory using boolean search in an archival file storage system. *Nature Materials*, 20:1272–1280, 2021. `doi:10.1101/2020.02.05.936369`.

6 Avraham Ben-Aroya, Oded Regev, and Ronald de Wolf. A hypercontractive inequality for matrix-valued functions with applications to quantum computing and ldcs. In *FOCS*, pages 477–486. IEEE Computer Society, 2008.

7 Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Robust pcps of proximity, shorter pcps, and applications to coding. *SIAM J. Comput.*, 36(4):889–974, 2006. A preliminary version appeared in the Proceedings of the 36th Annual ACM Symposium on Theory of Computing (STOC).

8 Arnab Bhattacharyya, L. Sunil Chandran, and Suprovat Ghoshal. Combinatorial lower bounds for 3-query ldcs. In *ITCS*, volume 151 of *LIPIcs*, pages 85:1–85:8. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

9 Arnab Bhattacharyya, Zeev Dvir, Shubhangi Saraf, and Amir Shpilka. Tight lower bounds for linear 2-query lccs over finite fields. *Comb.*, 36(1):1–36, 2016.

10 Arnab Bhattacharyya and Sivakanth Gopi. Lower bounds for constant query affine-invariant lccs and ltcs. *ACM Trans. Comput. Theory*, 9(2):7:1–7:17, 2017.

11 Arnab Bhattacharyya, Sivakanth Gopi, and Avishay Tal. Lower bounds for 2-query lccs over large alphabet. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2017.

12 Alexander R. Block and Jeremiah Blocki. Private and resource-bounded locally decodable codes for insertions and deletions. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 1841–1846, 2021. doi:10.1109/ISIT45174.2021.9518249.

13 Alexander R. Block, Jeremiah Blocki, Elena Grigorescu, Shubhang Kulkarni, and Minshen Zhu. Locally decodable/correctable codes for insertions and deletions. In *FSTTCS*, volume 182 of *LIPIcs*, pages 16:1–16:17, 2020.

14 Jeremiah Blocki, Kuan Cheng, Elena Grigorescu, Xin Li, Yu Zheng, and Minshen Zhu. Exponential lower bounds for locally decodable and correctable codes for insertions and deletions. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 739–750, 2022. doi:10.1109/FOCS52979.2021.00077.

15 Jeremiah Blocki, Venkata Gandikota, Elena Grigorescu, and Samson Zhou. Relaxed locally correctable codes in computationally bounded channels. *IEEE Transactions on Information Theory*, 67(7):4338–4360, 2021. doi:10.1109/TIT.2021.3076396.

16 Jeremiah Blocki, Shubhang Kulkarni, and Samson Zhou. On Locally Decodable Codes in Resource Bounded Channels. In Yael Tauman Kalai, Adam D. Smith, and Daniel Wichs, editors, *1st Conference on Information-Theoretic Cryptography (ITC 2020)*, volume 163, pages 16:1–16:23, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.ITC.2020.16.

17 Manuel Blum and Sampath Kannan. Designing programs that check their work. *J. ACM*, 42(1):269–291, 1995.

18 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. *J. Comput. Syst. Sci.*, 47(3):549–595, 1993.

19 Joshua Brakensiek, Venkatesan Guruswami, and Samuel Zbarsky. Efficient low-redundancy codes for correcting multiple deletions. *IEEE Trans. Inf. Theory*, 64(5):3403–3410, 2018.

20 Victor Chen, Elena Grigorescu, and Ronald de Wolf. Error-correcting data structures. *SIAM J. Comput.*, 42(1):84–111, 2013.

21 Kuan Cheng, Venkatesan Guruswami, Bernhard Haeupler, and Xin Li. Efficient linear and affine codes for correcting insertions/deletions. In *SODA*, pages 1–20. SIAM, 2021.

22 Kuan Cheng, Bernhard Haeupler, Xin Li, Amirbehshad Shahrasbi, and Ke Wu. Synchronization strings: Highly efficient deterministic constructions over small alphabets. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2185–2204. SIAM, 2019.

23 Kuan Cheng, Zhengzhong Jin, Xin Li, and Ke Wu. Deterministic document exchange protocols, and almost optimal binary codes for edit errors. In Mikkel Thorup, editor, *FOCS*, pages 200–211, 2018.

24 Kuan Cheng, Zhengzhong Jin, Xin Li, and Ke Wu. Block edit errors with transpositions: Deterministic document exchange protocols and almost optimal binary codes. In *ICALP*, volume 132 of *LIPIcs*, pages 37:1–37:15, 2019.

25 Kuan Cheng and Xin Li. Efficient document exchange and error correcting codes with asymmetric information. In *SODA*, pages 2424–2443. SIAM, 2021.

26 Kuan Cheng, Xin Li, and Yu Zheng. Locally decodable codes with randomized encoding. *CoRR*, abs/2001.03692, 2020. arXiv:2001.03692.

27 Alessandro Chiesa, Tom Gur, and Igor Shinkar. Relaxed locally correctable codes with nearly-linear block length and constant query complexity. In Shuchi Chawla, editor, *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1395–1411. SIAM, 2020.

28 Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45(6):965–981, 1998.

29 Gil Cohen and Tal Yankovitz. Relaxed locally decodable and correctable codes: Beyond tensoring. *Electron. Colloquium Comput. Complex.*, TR22-045, 2022. `arXiv:TR22-045`.

30 Marcel Dall'Agnol, Tom Gur, and Oded Lachish. A structural theorem for local algorithms with applications to coding, testing, and privacy. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1651–1665. SIAM, 2021.

31 Yan Ding, Parikshit Gopalan, and Richard Lipton. Error correction against computationally bounded adversaries. Manuscript, 2004.

32 Zeev Dvir, Parikshit Gopalan, and Sergey Yekhanin. Matching vector codes. *SIAM J. Comput.*, 40(4):1154–1178, 2011.

33 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Superquadratic lower bound for 3-query locally correctable codes over the reals. *Theory Comput.*, 13(1):1–36, 2017.

34 Klim Efremenko. 3-query locally decodable codes of subexponential length. *SIAM J. Comput.*, 41(6):1694–1703, 2012.

35 Anna Gál and Andrew Mills. Three-query locally decodable codes with higher correctness require exponential length. *ACM Trans. Comput. Theory*, 3(2):5:1–5:34, 2012.

36 William I. Gasarch. A survey on private information retrieval (column: Computational complexity). *Bulletin of the EATCS*, 82:72–107, 2004.

37 Oded Goldreich, Howard J. Karloff, Leonard J. Schulman, and Luca Trevisan. Lower bounds for linear locally decodable codes and private information retrieval. *Comput. Complex.*, 15(3):263–296, 2006.

38 Tom Gur and Oded Lachish. A lower bound for relaxed locally decodable codes. *arXiv preprint*, 2019. `arXiv:1904.08112`.

39 Tom Gur and Oded Lachish. On the power of relaxed local decoding algorithms. *SIAM J. Comput.*, 50(2):788–813, 2021.

40 Tom Gur, Govind Ramnarayan, and Ron Rothblum. Relaxed locally correctable codes. *Theory Comput.*, 16:1–68, 2020.

41 Venkatesan Guruswami, Bernhard Haeupler, and Amirbehshad Shahrasbi. Optimally resilient codes for list-decoding from insertions and deletions. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *STOC*, pages 524–537. ACM, 2020.

42 Venkatesan Guruswami and Ray Li. Coding against deletions in oblivious and online models. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 625–643. SIAM, 2018.

43 Venkatesan Guruswami and Ray Li. Polynomial time decodable codes for the binary deletion channel. *IEEE Trans. Inf. Theory*, 65(4):2171–2178, 2019.

44 Venkatesan Guruswami and Adam Smith. Optimal rate code constructions for computationally simple channels. *J. ACM*, 63(4):35:1–35:37, September 2016. `doi:10.1145/2936015`.

45 Venkatesan Guruswami and Carol Wang. Deletion codes in the high-noise and high-rate regimes. *IEEE Transactions on Information Theory*, 63(4):1961–1970, 2017.

46 Bernhard Haeupler. Optimal document exchange and new codes for insertions and deletions. In David Zuckerman, editor, *FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 334–347, 2019.

47 Bernhard Haeupler, Aviad Rubinstein, and Amirbehshad Shahrasbi. Near-linear time insertion-deletion codes and $(1+\epsilon)$-approximating edit distance via indexing. In Moses Charikar and Edith Cohen, editors, *STOC*, pages 697–708. ACM, 2019.

48 Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings: codes for insertions and deletions approaching the singleton bound. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *STOC*, pages 33–46. ACM, 2017.

49 Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings: explicit constructions, local decoding, and applications. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *STOC*, pages 841–854. ACM, 2018.

50 Bernhard Haeupler and Amirbehshad Shahrasbi. Synchronization strings and codes for insertions and deletions – a survey, 2021. `arXiv:2101.00711`.

51 Bernhard Haeupler, Amirbehshad Shahrasbi, and Madhu Sudan. Synchronization strings: List decoding for insertions and deletions. In Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella, editors, *ICALP*, volume 107 of *LIPIcs*, pages 76:1–76:14, 2018.

52 Brett Hemenway and Rafail Ostrovsky. Public-key locally-decodable codes. In *Advances in Cryptology – CRYPTO 2008, 28th Annual International Cryptology Conference, Proceedings*, pages 126–143, 2008.

53 Brett Hemenway, Rafail Ostrovsky, Martin J. Strauss, and Mary Wootters. Public key locally decodable codes with short keys. In *14th International Workshop, APPROX, and 15th International Workshop, RANDOM, Proceedings*, pages 605–615, 2011.

54 Brett Hemenway, Rafail Ostrovsky, and Mary Wootters. Local correctability of expander codes. *Inf. Comput.*, 243:178–190, 2015.

55 Jonathan Katz and Luca Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *STOC*, pages 80–86, 2000.

56 Iordanis Kerenidis and Ronald de Wolf. Exponential lower bound for 2-query locally decodable codes via a quantum argument. *J. Comput. Syst. Sci.*, 69(3):395–420, 2004.

57 Swastik Kopparty, Or Meir, Noga Ron-Zewi, and Shubhangi Saraf. High-rate locally correctable and locally testable codes with sub-polynomial query complexity. *J. ACM*, 64(2):11:1–11:42, 2017.

58 Swastik Kopparty and Shubhangi Saraf. Guest column: Local testing and decoding of high-rate error-correcting codes. *SIGACT News*, 47(3):46–66, 2016.

59 Vladimir Iosifovich Levenshtein. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966. Doklady Akademii Nauk SSSR, V163 No4 845-848 1965.

60 Richard J. Lipton. A new approach to information theory. In *STACS*, pages 699–708, 1994.

61 Shu Liu, Ivan Tjuawinata, and Chaoping Xing. On list decoding of insertion and deletion errors. *CoRR*, abs/1906.09705, 2019. `arXiv:1906.09705`.

62 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992.

63 Jiri Matousek Marcos Kiwi, Martin Loebl. Expected length of the longest common subsequence for large alphabets. *Advances in Mathematics*, 197(2):480–498, 2005.

64 Hugues Mercier, Vijay K. Bhargava, and Vahid Tarokh. A survey of error-correcting codes for channels with symbol synchronization errors. *IEEE Communications Surveys and Tutorials*, 12, 2010.

65 Silvio Micali, Chris Peikert, Madhu Sudan, and David A. Wilson. Optimal error correction against computationally bounded noise. In *Theory of Cryptography, Second Theory of Cryptography Conference, TCC 2005, Cambridge, MA, USA, February 10-12, 2005, Proceedings*, pages 1–16, 2005.

66 Michael Mitzenmacher. A survey of results for deletion channels and related synchronization channels. *Probability Surveys*, 6:1–3, July 2008.

67 Rafail Ostrovsky, Omkant Pandey, and Amit Sahai. Private locally decodable codes. In *ICALP*, pages 387–398, 2007.

68 Rafail Ostrovsky and Anat Paskin-Cherniavsky. Locally decodable codes for edit distance. In Anja Lehmann and Stefan Wolf, editors, *Information Theoretic Security*, pages 236–249, Cham, 2015. Springer International Publishing.

69 L. J. Schulman and D. Zuckerman. Asymptotically good codes correcting insertions, deletions, and transpositions. *IEEE Transactions on Information Theory*, 45(7):2552–2557, 1999.

70 Ronen Shaltiel and Jad Silbak. Explicit list-decodable codes with optimal rate for computationally bounded channels. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, pages 45:1–45:38, 2016.

71 N.J.A. Sloane. On single-deletion-correcting codes. *arXiv*, 2002. `arXiv:math/0207197`.

72 Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom generators without the XOR lemma (abstract). In *CCC*, page 4, 1999.

73 Luca Trevisan. Some applications of coding theory in computational complexity. *CoRR*, cs.CC/0409044, 2004. `arXiv:0409044`.

74 Stephanie Wehner and Ronald de Wolf. Improved lower bounds for locally decodable codes and private information retrieval. In *ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 1424–1436. Springer, 2005.

75 David P. Woodruff. New lower bounds for general locally decodable codes. Technical report, Weizmann Institute of Science, Israel, 2007.

76 David P. Woodruff. A quadratic lower bound for three-query linear locally decodable codes over any field. *J. Comput. Sci. Technol.*, 27(4):678–686, 2012.

77 S. M. Hossein Tabatabaei Yazdi, Ryan Gabrys, and Olgica Milenkovic. Portable and error-free dna-based data storage. *Scientific Reports*, 7:2045–2322, 2017. `doi:10.1038/s41598-017-05188-1`.

78 Sergey Yekhanin. Towards 3-query locally decodable codes of subexponential length. *J. ACM*, 55(1):1:1–1:16, 2008.

79 Sergey Yekhanin. Locally decodable codes. *Foundations and Trends in Theoretical Computer Science*, 6(3):139–255, 2012.

Near-Optimal Set-Multilinear Formula Lower Bounds

Deepanshu Kush
Department of Computer Science, University of Toronto, Canada

Shubhangi Saraf
Department of Computer Science and Department of Mathematics, University of Toronto, Canada

Abstract

The seminal work of Raz (J. ACM 2013) as well as the recent breakthrough results by Limaye, Srinivasan, and Tavenas (FOCS 2021, STOC 2022) have demonstrated a potential avenue for obtaining lower bounds for general algebraic formulas, via strong enough lower bounds for *set-multilinear* formulas.

In this paper, we make progress along this direction by proving *near-optimal* lower bounds against low-depth as well as unbounded-depth set-multilinear formulas. More precisely, we show that over any field of characteristic zero, there is a polynomial f computed by a polynomial-sized set-multilinear branching program (i.e., f is in *set-multilinear VBP*) defined over $\Theta(n^2)$ variables and of degree $\Theta(n)$, such that any product-depth Δ set-multilinear formula computing f has size at least $n^{\Omega(n^{1/\Delta}/\Delta)}$. Moreover, we show that any unbounded-depth set-multilinear formula computing f has size at least $n^{\Omega(\log n)}$.

If such strong lower bounds are proven for the iterated matrix multiplication (IMM) polynomial or rather, any polynomial that is computed by an *ordered* set-multilinear branching program (i.e., a further restriction of set-multilinear VBP), then this would have dramatic consequences as it would imply super-polynomial lower bounds for general algebraic formulas (Raz, J. ACM 2013; Tavenas, Limaye, and Srinivasan, STOC 2022).

Prior to our work, either only weaker lower bounds were known for the IMM polynomial (Tavenas, Limaye, and Srinivasan, STOC 2022), or similar strong lower bounds were known but for a hard polynomial not known to be even in set-multilinear VP (Kush and Saraf, CCC 2022; Raz, J. ACM 2009).

By known *depth-reduction* results, our lower bounds are essentially tight for f and in general, for any hard polynomial that is in set-multilinear VBP or set-multilinear VP. Any asymptotic improvement in the lower bound (for a hard polynomial, say, in VNP) would imply super-polynomial lower bounds for general set-multilinear *circuits*.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases Algebraic Complexity, Set-multilinear, Formula Lower Bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.15

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2023/017/

Acknowledgements We would like to thank Srikanth Srinivasan for several helpful and insightful discussions. Our early discussions with him were what inspired much of this work.

1 Introduction

1.1 Background on Algebraic Complexity

Algebraic Complexity Theory is the study of the complexity of computational problems which can be described as computing a multivariate polynomial $P(x_1, \ldots, x_N)$ over some elements $x_1, \ldots, x_N$ lying in a fixed field $\mathbb{F}$. Several fundamental computational tasks such as computing the determinant, permanent, matrix product, etc., can be represented using

this framework. The natural computational models that we investigate in this setting are models such as *algebraic circuits*, *algebraic branching programs*, and *algebraic formulas*, all of which employ the natural algebraic operations in $\mathbb{F}[x_1, \ldots, x_N]$ to compute P.

An *algebraic circuit* over a field $\mathbb{F}$ for a multivariate polynomial $P(x_1, \ldots, x_N)$ is a directed acyclic graph (DAG) whose internal vertices (called gates) are labeled as either $+$ (sum) or $\times$ (product), and leaves (vertices of in-degree zero) are labeled by the variables x_i or constants from $\mathbb{F}$. A special output gate (the root of the DAG) represents the polynomial P. If the DAG happens to be a tree, such a resulting circuit is called an *algebraic formula*. The size of a circuit or formula is the number of nodes in the DAG. We also consider the product-depth of the circuit, which is the maximum number of product gates on a root-to-leaf path. The class VP (respectively, VF) then is defined to be the collection of all polynomials having at most polynomially large degree which can be computed by polynomial-sized circuits (respectively, formulas).

An *algebraic branching program* is a layered DAG with two special nodes in it: a start-node and an end-node. All edges of the ABP go from layer $\ell - 1$ to layer ℓ for some ℓ (say start-node is the unique node in layer 0) and are labeled by a linear polynomial. Every directed path γ from start-node to end-node computes the monomial P_γ, which is the product of all labels on the path γ. The ABP computes the polynomial $P = \sum_\gamma P_\gamma$, where the sum is over all paths γ from start-node to end-node. Its size is simply the number of nodes in the DAG, its *depth* is the length of the longest path from the start-node to the end-node, and *width* is the maximum number of nodes in any layer. The class VBP then is defined to be the collection of all polynomials which can be computed by polynomial-sized branching programs[1].

The complexity of these models is measured by the size, which serves as an indicator of the time complexity of computing the polynomial. The product-depth measures the extent to which this computation can be made parallel. As these models are supposed to construct a formal polynomial P, they are *syntactic* models of computation. This is unlike a *Boolean* circuit, which is only required to model specific truth-table constraints. The problem of proving algebraic circuit lower bounds is therefore widely considered to be easier than its Boolean counterpart. Indeed, we know that proving VP $\neq$ VNP, the algebraic analog of the P vs NP problem, is implied by the latter separation in the non-uniform setting ([5]). Similarly, proving super-polynomial lower bounds for algebraic formulas is the algebraic analogue of the NC^1 vs NP problem and is also considered to be one of the central challenges in algebraic complexity theory. We refer the reader to [32] for a more elaborate survey of this topic and for the formal definitions of the algebraic complexity classes VF, VBP, VP, and VNP.

1.2 A Recent Breakthrough

Much like in the Boolean setting, the problem of showing lower bounds for *general* algebraic circuits (or even formulas) has remained elusive. However, some remarkable progress has been made very recently by Limaye, Srinivasan, and Tavenas ([23]) who in a spectacular breakthrough, showed the first super-polynomial lower bounds for algebraic formulas of *all* constant depths. Prior to their work, the best known lower bound ([18]) even for product-depth 1 (or $\Sigma\Pi\Sigma$ formulas) was only almost-cubic. This is in stark contrast with the Boolean setting, in which we have known strong constant-depth lower bounds for many

[1] The inclusions VF $\subseteq$ VBP $\subseteq$ VP follow.

decades [3, 11, 38, 13, 31, 34]. Constant-depth formulas are critical to the study of algebraic complexity theory, as unlike the Boolean setting, strong enough bounds against them are known to yield VP $\neq$ VNP ([2]). This helps put into perspective the importance of the work [23].

The crucial step in the proof of the [23] result is to first establish super-polynomial lower bounds for a certain restricted class of (low-depth) algebraic formulas, namely *set-multilinear* formulas which we now define along with other important circuit models. A polynomial is said to be homogeneous if each monomial has the same total degree and *multilinear* if every variable occurs at most once in any monomial. Now, suppose that the underlying variable set is partitioned into d sets $X_1, \ldots, X_d$. Then the polynomial is said to be *set-multilinear* with respect to this variable partition if each monomial in P has *exactly* one variable from each set. Note that a set-multilinear polynomial is both multilinear and homogeneous. Next, we define different models of computation corresponding to these variants of polynomials classes. An algebraic formula/branching program/circuit is set-multilinear with respect to a variable partition $(X_1, \ldots, X_d)$ if each internal node in the formula/branching program/circuit computes a set-multilinear polynomial[2]. Multilinear and homogeneous formulas/branching programs/circuits are defined analogously.

Several well-studied and interesting polynomials happen to be set-multilinear. For example, the determinant and the permanent polynomials, the study of which is profoundly consequential to the field of algebraic complexity theory, are set-multilinear (with respect to the column variables). Another well-studied polynomial, namely the Iterated Matrix Multiplication polynomial, is also set-multilinear. The polynomial $\mathrm{IMM}_{n,d}$ is defined on $N = dn^2$ variables, where the variables are partitioned into d sets $X_1, \ldots, X_d$ of size n^2, each of which is represented as an $n \times n$ matrix with distinct variable entries. The polynomial $\mathrm{IMM}_{n,d}$ is defined to be the polynomial that is the $(1,1)$-th entry of the product matrix $X_1 X_2 \cdots X_d$. Note that hence, this polynomial *precisely* captures the computational power of a branching program of width n and depth d and is "complete" for the class VBP. This polynomial has a simple divide-and-conquer-based set-multilinear formula of size $n^{O(\log d)}$, and more generally for every $\Delta \leq \log d$, a set-multilinear formula of product-depth Δ and size $n^{O(\Delta d^{1/\Delta})}$, and circuit[3] of size $n^{O(d^{1/\Delta})}$. Even without the set-multilinearity constraint, no significantly better upper bound is known. It is reasonable to conjecture that this simple upper bound is tight up to the constant in the exponent.

The lower bounds in [23] for general constant-depth algebraic circuits are shown in the following sequence of steps:

1. It is shown that general low-depth algebraic formulas can be transformed to set-multilinear algebraic formulas of low depth, and without much of a blow-up in size (as long as the degree is small). More precisely, any product-depth Δ formula of size s computing a polynomial that is set-multilinear with respect to the partition $(X_1, \ldots, X_d)$ where each $|X_i| \leq n$, can be converted to a set-multilinear formula[4] of product-depth 2Δ and size $\mathrm{poly}(s) \cdot d^{O(d)}$. Such a "set-multilinearization" of general formulas of small degree was already shown before in [28] (which we describe soon in more detail); however, the main contribution of [23] here is to prove this *depth-preserving* version of it.

[2] Of course, a non-root node need not be set-multilinear with respect to the *entire* variable partition. Nevertheless, here we demand that it must be set-multilinear with respect to some *subset* of the collection $\{X_1, \ldots, X_d\}$.

[3] Any product-depth Δ (set-multilinear) circuit of size s can be simulated by a product-depth Δ (set-multilinear) formula of size $s^{2\Delta}$. Hence, any constant-depth formula lower bound automatically yields a corresponding circuit lower bound.

[4] There is also an intermediate "homogenization" step which we skip describing here for the sake of brevity.

2. Strong lower bounds are then established for low-depth set-multilinear circuits (of small enough degree). More precisely, any set-multilinear circuit C computing $\mathrm{IMM}_{n,d}$ (where $d = O(\log n)$) of product-depth Δ must have size at least $n^{d^{\exp(-O(\Delta))}}$. This combined with the first step yields the desired lower bound for general constant-depth circuits.

Given Raz's set-multilinearization of formulas of small degree that we alluded to, and this description of the set-multilinear formula lower bounds from [23] when $d = O(\log n)$, it is evident the "small degree" regime is inherently interesting to study – as it provides an avenue, via *hardness escalation*, for tackling one of the grand challenges of algebraic complexity theory, namely proving super-polynomial lower bounds for general algebraic formulas. However, we shall now see that even the large degree regime can be equally consequential in this regard.

1.3 The Large Degree Regime

Consider a polynomial P that is set-multilinear with respect to the variable partition $(X_1, \ldots, X_d)$ where each $|X_i| \leq n$. In this paper, we shall focus on studying set-multilinear formula complexity in the regime where d and n are *polynomially* related (as opposed to say, the assumption $d = O(\log n)$ described above). We now provide some background and motivation for studying this regime.

In follow-up work [36], the same authors showed the first super-polynomial lower bound against unbounded-depth set-multilinear formulas computing $\mathrm{IMM}_{n,n}$[5]. As is astutely described in [36], studying the set-multilinear formula complexity of IMM is extremely interesting and consequential even in the setting $d = n$ because of the following reasons:

- $\mathrm{IMM}_{n,n}$ is a *self-reducible* polynomial i.e., it is possible to construct formulas for $\mathrm{IMM}_{n,n}$ by recursively using formulas for $\mathrm{IMM}_{n,d}$ (for any $d < n$). In particular, if we had formulas of size $n^{o(\log d)}$ for $\mathrm{IMM}_{n,d}$ (for some $d < n$), this would imply formulas of size $n^{o(\log n)}$ for $\mathrm{IMM}_{n,n}$. In other words, an optimal $n^{\Omega(\log n)}$ lower bound for $\mathrm{IMM}_{n,n}$ implies $n^{\omega_d(1)}$ lower bounds for $\mathrm{IMM}_{n,d}$ for any $d < n$.
- Raz in [28] showed that if an N-variate set-multilinear polynomial of degree d has an algebraic formula of size s, then it also has a set-multilinear formula of size $\mathrm{poly}(s) \cdot (\log s)^d$. In particular, for a set-multilinear polynomial P of degree $d = O(\log N / \log\log N)$, it follows that P has a formula of size $\mathrm{poly}(N)$ if and only if P has a set-multilinear formula of size $\mathrm{poly}(N)$. Thus, having $N^{\omega_d(1)}$ set-multilinear formula size lower bounds for such a low degree would imply super-polynomial lower bounds for general formulas.

In particular, proving the optimal $n^{\Omega(\log n)}$ set-multilinear formula size lower bound for $\mathrm{IMM}_{n,n}$ would have dramatic consequences as it would yield general formula lower bounds (and more specifically, the separation $\mathrm{VF} \subsetneq \mathrm{VBP}$). To this end, the authors in [36] are able to show a weaker bound of the form $(\log n)^{\Omega(\log n)}$ instead. Even though it is the case that "simply" improving the base of this exponent from $\log n$ to n yields general formula lower bounds, it seems that we are still far from achieving it. Indeed, as is observed in [36], we do not even have the optimal $n^{\Omega(\sqrt{n})}$ lower bound for $\mathrm{IMM}_{n,n}$[6] when product-depth $\Delta = 2$. For constant (or low) product-depths (i.e., when $\Delta \leq \log n$), [36] shows a set-multilinear formula size lower bound of $(\log n)^{\Omega(\Delta n^{1/\Delta})}$ for $\mathrm{IMM}_{n,n}$ (while we expect the lower bound to be $n^{\Omega(n^{1/\Delta})}$).

---

[5] Note that for $\mathrm{IMM}_{n,n}$, each X_i has size n^2, not n. But the important thing for us here is that the degree, n, is polynomially related to this parameter.

[6] This is known for set-multilinear (and even multilinear) $\Sigma\Pi\Sigma\Pi$ circuits (see [10, 19]), but those are only special cases of general product-depth 2 circuits, which are $\Sigma\Pi\Sigma\Pi\Sigma$.

The best set-multilinear lower bound we know for any explicit polynomial of degree $\Theta(n)$ and in $\text{poly}(n)$ variables and for any constant $\Delta \geq 2$ is indeed $n^{\Omega(n^{1/\Delta})}$, from a recent work by the authors ([22]). However, the polynomial for which these bounds are obtained is *not* $\text{IMM}_{n,n}$. The "hard" polynomial in this work is $\text{NW}_{n,n}$, which comes from the class of so-called Nisan-Wigderson design-based polynomials[7] and is known to be in VNP, but not known to be even in VP. The authors are also able to establish an $n^{\Omega(\log n)}$ set-multilinear formula size lower bound for $\text{NW}_{n,n}$ in the unbounded-depth setting. By far the most striking problem left open by this work is to "simplify" the hard polynomial from $\text{NW}_{n,n}$ to $\text{IMM}_{n,n}$.

We remark that such a line of simplification has been successful in other contexts in algebraic complexity theory. Indeed, for several lower bounds for algebraic circuit classes in the past, a lower bound was initially shown for the NW polynomial and then with additional effort, was shown to also hold for the IMM polynomial. For instance, [17] showed a lower bound of $n^{\Omega(\sqrt{n})}$ for the top fan-in of a $\Sigma\Pi^{[O(\sqrt{n})]}\Sigma\Pi^{[\sqrt{n}]}$ circuit computing the NW polynomial, which was subsequently shown for IMM by [10]. Similarly, [15] showed an $n^{\Omega(\sqrt{d})}$ size lower bound for homogeneous depth-4 algebraic formulas for the NW polynomial, which was then shown for IMM later in [21]. In our context of set-multilinear formula lower bounds, such a simplification from NW to IMM would be especially momentous as it would directly lead to general formula lower bounds. In this paper, although we are presently unable to simplify all the way down to IMM, we manage to make significant progress along this direction.

1.4 Our Results

The main result of this work is the following statement.

▶ **Theorem 1.** *Let N be a growing parameter and Δ be a constant integer. Then, over any field of characteristic zero, there is an explicit polynomial P_N defined over $N = \Theta(n^2)$ variables with degree $d = \Theta(n)$ that is set-multilinear with respect to the variable partition $X = (X_1, \ldots, X_d)$ where each $|X_i| = n$ and such that:*

- *there is a $\text{poly}(N)$-size set-multilinear branching program computing P_N (i.e., its every internal node computes a set-multilinear polynomial),*
- *any set-multilinear formula of product-depth Δ computing P_N must have size at least $N^{\Omega(d^{1/\Delta})}$, and*
- *further, any set-multilinear formula of arbitrary product-depth P_N must have size at least $N^{\Omega(\log d)}$.*

▶ Remark 2. Similar to [22], the lower bound in Theorem 1 is actually $d^{\Omega(d^{1/\Delta}/\Delta)}$, where d is the degree of the underlying polynomial, and it holds as long as degree $d \leq n$ and the product-depth $\Delta \leq \log d / \log\log d$ (the details are deferred to the proof of Theorem 19 in Section 4).

A few more remarks are in order. First, given that $\text{IMM}_{n,n}$ is complete for the class VBP as described in Section 1.2, one might expect that Theorem 1 immediately implies such a lower bound for $\text{IMM}_{n,n}$ as well, thereby obtaining general formula lower bounds. Curiously, however, this is not the case. This is because the underlying reduction from P_N to $\text{IMM}_{n,n}$ destroys the set-multilinearity of the formula, and hence the set-multilinear formula lower bounds no longer apply. Nevertheless, we observe that if we can make the hard polynomial in

[7] The name is inspired from [24].

Theorem 1 be computable by a polynomial-sized *ordered* set-multilinear branching program[8], then that does yield the desired lower bounds for $\mathrm{IMM}_{n,n}$. More precisely, given a variable partition $(X_1, \ldots, X_d)$ (say with each $|X_i| \leq n$), we say that a set-multilinear branching program of width n and depth d is *ordered* with respect to $(X_1, \ldots, X_d)$ if for each $\ell \in [d]$, all edges of the ABP from layer $\ell - 1$ to layer ℓ are labeled using a linear form *in* X_ℓ. Let $f(X_1, \ldots, X_d)$ be a polynomial that can be computed by an ordered set-multilinear ABP A. Then, given any set-multilinear formula computing $\mathrm{IMM}_{n,d}$ (say over the variables $\{x_{i,j}^{(\ell)}\}$ where $i, j \in [n]$ and $1 \leq \ell \leq d$), we immediately obtain a set-multilinear formula of the same size for f by replacing $x_{i,j}^{(\ell)}$ with the linear form $e_{i,j}^{(\ell)}$, where $e_{i,j}^{(\ell)}$ is the label of the edge in A between the i-th node of layer $\ell - 1$ and j-th node of layer ℓ. As a consequence, any lower bound on the size of a set-multilinear formula computing f yields a lower bound for one computing $\mathrm{IMM}_{n,d}$. We conclude that replacing P_N in Theorem 1 by any polynomial that is computable by an ordered set-multilinear ABP would yield general formula lower bounds! This observation raises the question of the relative power of ordered vs general set-multilinear ABPs – we leave this as an intriguing open problem (see Section 5).

We also remark that obtaining this precise bound is interesting also when viewed through the lens of *depth-reduction*. Tavenas ([35]), building on several prior works ([2, 20]), showed that any algebraic circuit of $\mathrm{poly}(N)$ size computing a homogeneous N-variate polynomial of degree d can be converted to a homogeneous circuit of product-depth[9] Δ of size $N^{O(d^{1/\Delta})}$. It easily follows from the proof that this depth reduction preserves syntactic restrictions. That is, if we start with a syntactically set-multilinear circuit, the resulting product-depth Δ circuit is also syntactically set-multilinear. Therefore, because P_N has a polynomial-sized set-multilinear circuit (in particular, a set-multilinear ABP), it follows that it has a product-depth Δ set-multilinear formula of size $N^{O(d^{1/\Delta})}$. Furthermore, by classical depth-reduction results[10], it follows that a size s, degree d set-multilinear circuit can be simulated by a set-multilinear formula of size $s^{O(\log d)}$. Hence, the lower bounds we obtain for P_N in Theorem 1 – in both, the constant and unbounded-depth settings – are *asymptotically optimal*. In fact, the precise bound in Theorem 1 is also *sharp* in the sense that any asymptotic improvement in its exponent for *any* constant Δ (say, for a set-multilinear polynomial in VNP) would imply super-polynomial set-multilinear circuit lower bounds (i.e., set-multilinear VP $\neq$ set-multilinear VNP), which would be quite a strong and exciting result, as it would demonstrate considerable progress towards the VP vs VNP problem.

On a related note, in [22], the authors posed a question about the possibility of obtaining *improved* depth-reduction bounds for set-multilinear circuits. More specifically, it was observed that if any asymptotic improvement in the exponent on the $N^{O(d^{1/\Delta})}$ bound for general circuits from [35] could be shown to hold for set-multilinear circuits in the setting of Theorem 1 (i.e., when $N = \Theta(d^2)$), then combined with the lower bounds for $\mathrm{NW}_{n,n}$, this would imply super-polynomial set-multilinear circuit lower bounds. It was noted that [10] rules out the possibility of obtaining a stronger reduction to depth-4, or $\Sigma\Pi\Sigma\Pi$ circuits, as it shows an $n^{\Omega(\sqrt{n})}$ size lower bound for set-multilinear depth-4 circuits computing $\mathrm{IMM}_{n,n}$, which of course has small polynomial-sized set-multilinear circuits. Nevertheless, there could still be the possibility of obtaining improved depth-reduction statements for product-depths 2 (which is $\Sigma\Pi\Sigma\Pi\Sigma$ and hence more general than depth-4) or higher, and combining it

[8] Interestingly, this model, along with that of read-once oblivious ABPs (or ROABPs), has been studied quite extensively in the polynomial identity testing (PIT) literature; see [9, 1, 12].

[9] The result is stated in [35] for $\Sigma\Pi\Sigma\Pi$ circuits but the proof can be appropriately modified for larger product-depths.

[10] See [37] and then, [4] for an adaptation to the set-multilinear setting.

with the lower bound for $\mathrm{NW}_{n,n}$ to obtain general set-multilinear circuit lower bounds. We answer this question in the negative and remark that Theorem 1 implies that an improved depth-reduction bound for set-multilinear circuits is impossible (at least when N and d are polynomially related).

We also point out the differences in the quality of the best lower bounds known in the closely related (and more general) *multilinear* setting. Despite the multilinear formula model receiving significant attention in the literature[11], to the best of our knowledge, the best known lower bound for a polynomial of degree n over $\mathrm{poly}(n)$ variables even for product-depth 2 multilinear formulas[12] is only $2^{\Omega(\sqrt{n})}$ ([7]), and generalizes as $2^{\Omega(\Delta n^{1/\Delta})}$ for higher $\Delta \leq \log n$. In contrast, using the terminology of [23], the lower bounds that we obtain for constant product-depth *set-multilinear* formulas in this paper (and indeed, in [22]) are stronger, *non-FPT bounds*. Furthermore, we point out that even solely the third item of Theorem 1 i.e., an $n^{\Omega(\log n)}$ lower bound for a set-multilinear polynomial of degree n over $\mathrm{poly}(n)$ variables computable by a small set-multilinear branching program, is a new result – as far as we know, it is not implied by any prior work. For example, though [27] shows that the $n \times n$ determinant and permanent require $n^{\Omega(\log n)}$ multilinear formula size, these polynomials are not actually known to have small set-multilinear (or even multilinear) circuits – in fact, they are conjectured not to ([25]).

We now move on to the second result of this paper. As noted earlier, prior to this work, the "hard" polynomial for which we had the same lower bounds as Theorem 1 was not known to be even in VP. In the result below, we construct a set-multilinear polynomial in VP matching the bounds of [22].

▶ **Theorem 3.** *Let N be a growing parameter and Δ be a constant integer. Then, over any field of characteristic zero, there is an explicit polynomial Q_N defined over $N = \Theta(n^2)$ variables with degree $d = \Theta(n)$ that is set-multilinear with respect to the variable partition $X = (X_1, \ldots, X_d)$ where each $|X_i| = n$ and such that:*

- *there is a $\mathrm{poly}(N)$-size set-multilinear circuit computing Q_N,*
- *any set-multilinear formula of product-depth Δ computing Q_N must have size at least $N^{\Omega(d^{1/\Delta})}$, and*
- *further, any set-multilinear formula of arbitrary product-depth Q_N must have size at least $N^{\Omega(\log d)}$.*

▶ Remark 4. Similar to Theorem 1, the lower bound in Theorem 3 is actually $d^{\Omega(d^{1/\Delta}/\Delta)}$, where d is the degree of the underlying polynomial, and it holds as long as degree $d \leq n$ and the product-depth $\Delta \leq \log d / \log\log d$ (the details are deferred to the proof of Theorem 10 in Section 3).

Evidently, despite already being a new result, Theorem 3 is subsumed by Theorem 1. However, as we shall see in Section 3 and also in the proof overview below, this construction and the associated lower bound argument is simpler than that of Theorem 1. Moreover, this argument will be quite instructive and helpful for the reader to ease into the proof of the main result (Theorem 19 in Section 4).

[11] See [25, 30, 7, 6, 16] for results in the bounded-depth setting and [27, 8, 14, 19] for results in the unbounded-depth setting. Note that however, in many of these works, the "hard" polynomial is not set-multilinear and as such, the corresponding lower bounds do not even apply in our setting.

[12] The situation is significantly better for $\Delta = 1$ (or multilinear $\Sigma\Pi\Sigma$ formulas) as [16] shows a lower bound of $n^{\Omega(d)}$ – in fact, for $\mathrm{IMM}_{n,d}$.

1.5 Proof Overview and Relation to Prior Work

In this subsection, we give an overview of the proof techniques used in both Theorems 1 and 3. We divide the subsection into two parts: the first part discusses the construction of our hard polynomial in VP (which is mainly inspired from a result ([29]) of Raz and Yehudayoff) and the second part discusses the construction of our hard polynomial in VBP (which relies upon the *arc-partition* framework of Dvir, Malod, Perifel, and Yehudayoff ([8])).

VP Construction

We shall first discuss an overview of the proof of Theorem 3. At a high-level, our overall proof techniques are similar to that of many known lower bounds. We work with a measure that is known to be small for all polynomials computed by small enough set-multilinear formulas (suitably so in the bounded and unbounded-depth settings) from the work [22], where it is also shown to be large for the NW polynomial. These *partial derivative measures* were introduced by Nisan and Wigderson in [25], who used them to prove the constant-depth set-multilinear formula lower bounds we discussed earlier. [23, 36] use a particular variant of this measure and this measure is in turn inspired from these works.

Given a variable partition $(X_1, \ldots, X_d)$, the idea is to label each set of variables X_i as "positive" or "negative" uniformly at random. Let $\mathcal{P}$ and $\mathcal{N}$ denote the set of positive and negative indices respectively, and let $\mathcal{M}^{\mathcal{P}}$ and $\mathcal{M}^{\mathcal{N}}$ denote the sets of all set-multilinear monomials over $\mathcal{P}$ and $\mathcal{N}$ respectively. For a polynomial f that is set-multilinear over the given variable partition $(X_1, \ldots, X_d)$, the measure then is simply the rank of the "partial derivative matrix" $\mathcal{M}(f)$ whose rows are indexed by the elements of $\mathcal{M}^{\mathcal{P}}$ and columns indexed by $\mathcal{N}^{\mathcal{P}}$, and the entry of this matrix corresponding to a row m_1 and a column m_2 is the coefficient of the monomial $m_1 \cdot m_2$ in the given polynomial. We remark that though this was inspired by the measure and the techniques from [23], it is also reminiscent of the measure used in [26, 27] to prove multilinear formula lower bounds. [22] shows that indeed for the NW polynomial, the matrix $\mathcal{M}(NW)$ *always* has full-rank (at least when conditioned on the event $|\mathcal{P}| = |\mathcal{N}|$).

In proving Theorem 3, our main contribution is to construct a set-multilinear polynomial Q such that $\mathcal{M}(Q)$ always has full-rank – but in addition, Q is computable by a small set-multilinear circuit. For this, we turn to the literature on the multilinear setting for inspiration. In [26], Raz constructed a multilinear polynomial g computable by a small multilinear circuit and showed a super-polynomial (general-depth) multilinear formula size bound for it. The measure used was the rank of a matrix defined analogously to $\mathcal{M}(\cdot)$. Our starting point for constructing Q was a *simplification* of g (which we call h) by Raz and Yehudayoff ([29]) using Dyck words[13], which we shall describe in more detail in Section 3. One idea that is key in these constructions is the introduction of *auxiliary* variables: $h = h(X, \Lambda)$ is defined over an *original* variable set $X = \{x_1, \ldots, x_n\}$ and an *auxiliary* variable set Λ of poly(n) size. It is then shown that the matrix associated to $h(X)$ has full-rank (i.e., h has "large" measure), when viewed as a matrix over the extended field $\mathbb{F}(\Lambda)$. In other words, the auxiliary variables assist in showing that its matrix (whose entries are now polynomials over variables in Λ) is non-singular.

While attempting to "set-multilinearize" the construction of $h(X, \Lambda)$ in order to define our $Q(X_Q, \Lambda_Q)$ (say where $X_Q = (X_1, \ldots, X_d)$ and each $|X_i| = n$), we were able adapt the correct (i.e., a set-multilinear) dependence of Q on the X_Q-variables (from that of h

[13] [29] does not actually explicitly use Dyck words in its construction, but we benefited from its exposition given in [32].

on X) in a relatively straightforward manner – this involved picking the right "gadget" or "building block" in the set-multilinear setting, which turns out to the *inner product gadget* (see Observation 8). Essentially, the simple observation that if X_1 is labeled "positive" and X_2 is labeled "negative", then the $n \times n$ matrix corresponding to the inner product polynomial $X_1 \cdot X_2$ is full-rank allows us to "build" more complicated and higher-degree full-rank polynomials, similar to how h is "built" by [29]. However, the main hurdle that we encountered while trying to construct Q using the construction of h was to achieve the correct dependence on the auxiliary variables. The issue is that if we introduce too many *sets* of auxiliary variables (i.e., if $\Lambda_Q = (\Lambda_1, \ldots, \Lambda_{d'})$ and $d' = \omega(d)$), then the degree of the polynomial blows up and because we work over the extended field $\mathbb{F}(\Lambda)$, the final quantitative expression for the lower bound in the constant-depth case of Theorem 3 suffers (in fact, it becomes even worse than the aforementioned lower bound of [7] in the constant-depth multilinear formula model). As a consequence, we need to be judicious in our use of the auxiliary variables – we highlight some of the finer details later on in Section 3. For this reason, the analysis of our hard polynomial being full-rank ends up being more intricate than [29]. In turn, this leads to the demand that the characteristic of $\mathbb{F}$ be zero – although we suspect that this assumption should not be necessary; see the discussion in Section 5.

VBP Construction

For proving Theorem 1, we build upon the work of Dvir, Malod, Perifel, and Yehudayoff ([8]), who showed the first separation between multilinear branching programs and formulas. That is, they constructed an n-variate polynomial F that can be computed by a small multilinear branching program, but needs multilinear formulas of size $n^{\Omega(\log n)}$ to compute. Our overall strategy is to adapt their approach to our set-multilinear setting – however, there are some inherent difficulties in doing so because of the nature of the very strong bounds sought in the low-depth setting (which was not an issue for [8] as this setting was not considered in that work). In what follows, we provide an overview of the *arc-partition* framework of [8], state it in our set-multilinear setting, and describe the additional challenges we face with the adaptation.

The proof of Theorem 1 consists of two parts: (i) constructing a small set-multilinear ABP computing a polynomial G and (ii) showing that any set-multilinear multilinear formula computing G must be very large (appropriately so in the constant and general-depth settings). The two parts have opposing demands: In part (i) we wish to make the polynomial G simple enough so that a small ABP can compute it, whereas in part (ii) we will need to rely on the hardness of G to prove lower bounds. One might wonder if we can get away with using the same rank measure that was defined above for the VP construction in order to meet these two demands. However, as far as we know, full-rank polynomials (in the sense described above) may also require super-polynomial sized set-multilinear ABPs. [8] were faced with a similar challenge: full-rank multilinear polynomials (say with respect to the aforementioned analogous measure of [26]) may also require super-polynomial sized multilinear ABPs. Thus, in order to prove a separation between multilinear ABPs and formulas, they sought a property which is *weaker* than being full-rank but is still useful enough for proving lower bounds. One of the main ideas in their proof is an ingenious construction of a special subset of partitions, called *arc-partitions*, which is sufficiently powerful to carry through the lower bound proof and, at the same time, simple enough to carry part (i) of the proof. In this context, a partition simply refers to a particular "positive"/"negative" labelling of the variable sets $X_1, \ldots, X_d$. The point is that the support of the distribution over these *arc-partitions* turns out to be much smaller than the support of the uniform distribution over such labellings that

was used as our measure in the VP construction. Nevertheless, after overcoming some hurdles that we soon describe, we are able to adapt their argument to show that every *arc-full-rank* polynomial f (i.e., the matrix $\mathcal{M}(f)$ is always full-rank, but now defined only with respect to the labellings coming from this special *arc-partition* distribution, instead of the uniform distribution) must have very large set-multilinear formulas – appropriately so in the constant and general-depth settings.

Let us now describe this family of partitions (stated in our set-multilinear setting) and its advantages. More specifically, we will describe a distribution over partitions (or labellings, as explained above). The partitions that will have positive probability of being obtained in this distribution will be called arc-partitions. The distribution is defined according to the following (iterative) sampling algorithm. Position the d variable sets on a cycle with d nodes so that there is an edge between i and $i+1$ modulo d. Start with the arc $[L_1, R_1] = \{1, 2\}$ (an arc is a connected path on the cycle). At step $t > 1$ of the process, maintain a partition of the arc $[L_t, R_t]$. "Grow" this partition by first picking a pair uniformly at random out of the three possible pairs $\{L_t - 2, L_t - 1\}, \{L_t - 1, R_t + 1\}, \{R_t + 1, R_t + 2\}$, and then choosing a labelling (or partition) Π on this pair i.e., assigning one of them "positive" and the other "negative" uniformly at random. After $d/2$ steps, we have chosen a partition of the d variable sets into two disjoint, equal-size sets of variables $\mathcal{P}$ and $\mathcal{N}$.

Given these arc-partitions of [8], let us now briefly describe how we obtain the desired optimal lower bounds in the constant-depth setting. In part (i) of the proof, in order to construct an arc-full-rank set-multilinear branching program, we face similar challenges as we did in the VP construction – but similarly, a more careful use of the auxiliary variables comes to the rescue. Next, we show that with high probability over the arc-partition distribution, the rank of a polynomial computed by a product-depth Δ set-multilinear formula is (appropriately) small. This is done via a proof by induction on Δ. We separately show that each summand C_i of $C = C_1 + \cdots + C_t$ for a product-depth Δ formula C has small enough rank, yielding the desired bound by the sub-additivity of rank. There are two cases: either C_i already has a factor of very large degree (i.e., at least $\sim d^{\frac{\Delta-1}{\Delta}}$, which allows us to use the inductive hypothesis for $\Delta - 1$) or otherwise, we argue that we may assume that it has many factors (roughly $K \sim d^{1/\Delta}$ many) of a similar degree. It is this inductive argument (and specifically, the first case) that forces us to work with an arc-partition over a larger D-cycle (where $D \geq d$) – one of the reasons contributing to a more nuanced analysis than [8]. In the second case, the many factors then define a "non-redundant" K-coloring of the d variable sets. This is simply a (partial) mapping $C_i : [D] \to [K]$ so that the pre-images of every color $k \in [K]$ are not too small (and of similar sizes). A color k is said to be "balanced" with respect to a partition Π if the number of "positive" variable sets of color k is roughly the same as the number of "negative" variable sets of color k. Now, for a given coloring C_i, if we choose a random partition Π from the set of all partitions, simple properties of the hyper-geometric distribution imply that the probability that all colors in C_i are "balanced" is at most $p = d^{-\Omega(K)} = d^{-\Omega(d^{1/\Delta})}$. This bound, in turn, proves a roughly $1/p = d^{\Omega(d^{1/\Delta})}$ lower bound for the size of product-depth Δ set-multilinear formulas for the VP construction (Theorem 3). Following a similar overall outline, we adapt the [8] argument to show that for any "non-redundant" partial K-coloring C_i, for a random arc-partition, the probability that all colors in C_i are "balanced" is at most $d^{-\Omega(K)}$ as well. This turns out to be significantly more difficult than showing it for a random partition (from the set of all partitions). Furthermore, because we seek such strong and optimal bounds in the low-depth setting, the analysis turns out to be more intricate (Section 4.4 in particular). Throughout Section 4 where we formally prove Theorem 1, we have suitably placed remarks to point out the locations where we require a different technical or conceptual argument than [8].

2 Preliminaries

2.1 Relative Rank and its Properties

We first describe the notation that we need to define the measures that we use to prove Theorems 1 and 3.

▶ **Definition 5** (Relative Rank Measure of [23, 36]). *Let f be a polynomial that is set-multilinear with respect to the variable partition $(X_1, X_2, \ldots, X_d)$ where each set is of size n. Let $w = (w_1, w_2, \ldots, w_d)$ be a tuple (or word) of non-zero real numbers such that $2^{|w_i|} \in [n]$ for all $i \in [d]$. For each $i \in [d]$, let $X_i(w)$ be the variable set obtained by removing arbitrary variables from the set X_i such that $|X_i(w)| = 2^{|w_i|}$, and let $\overline{X}(w)$ denote the tuple of sets of variables $(X_1(w), \ldots, X_d(w))$. Corresponding to a word w, define $\mathcal{P}_w := \{i \mid w_i > 0\}$ and $\mathcal{N}_w := \{i \mid w_i < 0\}$. Let $\mathcal{M}_w^{\mathcal{P}}$ be the set of all set-multilinear monomials over a subset of the variable sets $X_1(w), X_2(w), \ldots, X_d(w)$ indexed by $\mathcal{P}_w$, and similarly let $\mathcal{M}_w^{\mathcal{N}}$ be the set of all set-multilinear monomials over these variable sets indexed by $\mathcal{N}_w$.*

Define the 'partial derivative matrix' matrix $\mathcal{M}_w(f)$ whose rows are indexed by the elements of $\mathcal{M}_w^{\mathcal{P}}$ and columns indexed by the elements of $\mathcal{N}_w^{\mathcal{P}}$ as follows: the entry of this matrix corresponding to a row m_1 and a column m_2 is the coefficient of the monomial $m_1 \cdot m_2$ in f. We define

$$\mathrm{relrk}_w(f) := \frac{\mathrm{rank}(\mathcal{M}_w(f))}{\sqrt{|\mathcal{M}_w^{\mathcal{P}}| \cdot |\mathcal{M}_w^{\mathcal{N}}|}} = \frac{\mathrm{rank}(\mathcal{M}_w(f))}{2^{\frac{1}{2}\sum_{i\in[d]}|w_i|}}.$$

▶ **Definition 6.** *For any tuple $w = (w_1, \ldots, w_t)$ and a subset $S \subseteq [t]$, we shall refer to the sum $\sum_{i\in S} w_i$ by w_S. And by $w|_S$, we will refer to the tuple obtained by considering only the elements of w that are indexed by S. We denote by $\mathbb{F}_{\mathrm{sm}}[\mathcal{T}]$ the set of set-multilinear polynomials over the tuple of sets of variables $\mathcal{T}$.*

The following is a simple result that establishes various useful properties of the relative rank measure.

▷ Claim 7 ([23]).

1. (Imbalance) Say $f \in \mathbb{F}_{\mathrm{sm}}[\overline{X}(w)]$. Then, $\mathrm{relrk}_w(f) \leq 2^{-|w_{[d]}|/2}$.
2. (Sub-additivity) If $f, g \in \mathbb{F}_{\mathrm{sm}}[\overline{X}(w)]$, then $\mathrm{relrk}_w(f+g) \leq \mathrm{relrk}_w(f) + \mathrm{relrk}_w(g)$.
3. (Multiplicativity) Say $f = f_1 f_2 \cdots f_t$ and assume that for each $i \in [t]$, $f_i \in \mathbb{F}_{\mathrm{sm}}[\overline{X}(w|_{S_i})]$, where $(S_1, \ldots, S_t)$ is a partition of $[d]$. Then

$$\mathrm{relrk}_w(f) = \prod_{i\in[t]} \mathrm{relrk}_{w|_{S_i}}(f_i).$$

2.2 Inner Product Gadget

We crucially need the following observation to construct the hard polynomials in Theorems 1 and 3.

▶ **Observation 8.** *Let $n = 2^k$ and $X_1 = \{x_{1,1}, \ldots, x_{1,n}\}$ and $X_2 = \{x_{2,1}, \ldots, x_{2,n}\}$ be two disjoint sets of variables. Then, for any* symmetric *word $w \in \{k, -k\}^2$ (i.e., where $w_1 + w_2 = 0$) and for the inner product "gadget" $f = X_1 \cdot X_2 = \sum_{i=1}^{n} x_{1,i}x_{2,i}$, $\mathrm{relrk}_w(f) = 1$ i.e., $\mathcal{M}_w(f)$ is full-rank.*

3 A Hard Set-multilinear Polynomial in VP

3.1 Description of the Polynomial

Let d be an even integer and let $X = (X_1, \ldots, X_d)$ be a collection of sets of variables where each $|X_i| = n$, and similarly, let $Y = (Y_1, \ldots, Y_d)$ be a distinct collection of sets of variables where each $|Y_i| = n$. We shall refer to the Y-variables as the *auxiliary* variables. For i and $j \in \{1, \ldots, d\}$, let $X_i \cdot X_j$ denote the inner-product quadratic form $\sum_{k=1}^{n} x_{ik}x_{jk}$. Here, we shall assume that $X_i = \{x_{i,1}, \ldots, x_{i,n}\}$ and $Y_i = \{y_{i,1}, \ldots, y_{i,n}\}$.

For two integers $i \in \mathbb{N}$ and $j \in \mathbb{N}$, we denote $[i, j] = \{k \in \mathbb{N} : i \leq k \text{ and } k \leq j\}$ and call such a set an *interval*. For every interval $[i, j] \subseteq [d]$, we define a polynomial $f_{i,j}(X, Y) \in \mathbb{F}_{\text{sm}}[X_i, \ldots, X_j, Y_i, \ldots, Y_j]$ as follows:

$$f_{i,j} = \begin{cases} y_{i,j}y_{j,i}(X_i \cdot X_j) & \text{if } j = i + 1 \\ 0 & \text{if } j - i \text{ is even} \\ y_{i,j}y_{j,i}(X_i \cdot X_j) \cdot f_{i+1,j-1} + \sum_{r=i+1}^{j-1} f_{i,r}f_{r+1,j} & \text{otherwise} \end{cases}$$

▶ Remark 9. As described in Section 1.5, other than the use of the inner product gadget, one key difference between $f_{i,j}$ and the construction in [29] is that it uses fewer auxiliary variables. More specifically, while [29] had a "fresh" auxiliary variable for every choice of i, r, j in the sum, we are unable to afford that not only because it destroys the set-multilinearity of the polynomial but most importantly, because of the aforementioned degree blow-up. This is also the reason why more straightforward attempts to "set-multilinearize" [29] such as by adding two "copies" y_0 and y_1 for each of their auxiliary variables y (where intuitively y_0 and y_1 correspond to setting y as 0 or 1 respectively in their argument) do not work.

The following is a more precise and general version of Theorem 3 that is stated in Section 1. We also incorporate Remark 4 here and show our lower bound for any degree $d \leq n$. Theorem 3 follows from the special case $d = n$.

▶ **Theorem 10.** *Let $n = 2^k$, and suppose $d \leq n$ be an even integer that is large enough*[14]*, and $1 \leq \Delta \leq \log d / \log\log d$ be any positive integer. Let X_i, Y_i denote the sets of n variables $\{x_{i,j} : j \in [n]\}$ and $\{y_{i,j} : j \in [n]\}$ respectively and let X, Y be the tuples $(X_1, \ldots, X_d)$ and $(Y_1, \ldots, Y_d)$. Then,*

- *there is a $\text{poly}(n, d)$-size set-multilinear circuit computing $F_{n,d} = f_{1,d}(X, Y)$ as defined above,*
- *any set-multilinear formula of product-depth Δ computing $F_{n,d}$ must have size at least $d^{\Omega(d^{1/\Delta}/\Delta)}$, and*
- *further, any set-multilinear formula of arbitrary product-depth computing $F_{n,d}$ must have size at least $d^{\Omega(\log d)}$.*

3.2 Proof of Hardness

Note that the first item in Theorem 10 follows immediately from the recursive definition of $f_{1,d}$ (notice that there are only up to d^2 many distinct intervals of $[d]$). For proving the next two items, we invoke the *symmetric word* framework of [22]. The following couple of lemmas help establish that the relative rank measure with respect to symmetric words is (suitably) small for low-depth and general-depth set-multilinear formulas, respectively.

[14] We only need d to be larger than some absolute constant.

▶ **Lemma 11** ([22]). *Let C be a set-multilinear formula of product-depth $1 \leq \Delta \leq \log d / \log\log d$ of size at most s which computes a polynomial (over any fixed field) that is set-multilinear with respect to the partition $(X_1, \ldots, X_d)$ where each $|X_i| = n$. Let $w \in \{k, -k\}^d$ be chosen uniformly at random. Then, we have*

$$\text{relrk}_w(C) \leq s \cdot 2^{-\frac{k d^{1/\Delta}}{20}}$$

with probability at least $1 - s \cdot d^{-\frac{d^{1/\Delta}}{12\Delta}}$.

▶ **Lemma 12** ([22]). *Let F be a set-multilinear formula of size at most s which computes a polynomial (over any fixed field) that is set-multilinear with respect to the partition $(X_1, \ldots, X_d)$ where each $|X_i| = n$. Let $w \in \{k, -k\}^d$ be chosen uniformly at random. Then, we have*

$$\text{relrk}_w(F) \leq s \cdot 2^{-\frac{k \log d}{20}}$$

with probability at least $1 - s \cdot d^{-\frac{\log d}{60}}$.

Next, we shall show that the hard polynomial $F_{n,d}$ in Theorem 10 has high relative rank (in fact, the maximum possible value – 1) with respect to a symmetric word. For this, we consider an alternate view of these polynomials and require the following notion. For an even integer d, define Dyck(d) to be the collection of all strings (called *Dyck words*) of length d over symbols '(' and ')' that are well-matched in the natural way. More precisely, it is the collection of all strings u of length d such that all prefixes of u contain no more)'s than ('s and the total number of ('s in u equals the total number of). For example, "()()" and "(())" belong to Dyck(4) but not "(()(". Note that for any '(' appearing in a Dyck word, there is a *unique* ')' which "closes" it. Given a Dyck word $u \in$ Dyck(d), we call (i, j) a *matching parenthesis pair* of u if there is '(' in the i-th position of u that is closed by a ')' in the j-th position of u (clearly then, $j - i > 0$ must be odd).

Given a string $u \in$ Dyck(d) and the setup above for defining the polynomials $f_{i,j}$, we can associate to u a *product of inner products* polynomial $IP_u \in \mathbb{F}_{\text{sm}}[X_i, \ldots, X_j]$ in the natural way: define IP_u to be the product of all $X_i \cdot X_j$ where (i, j) is a matching parenthesis pair of u. For example, the strings "()()" and "(())" would correspond to the polynomials $(X_1 \cdot X_2) \cdot (X_3 \cdot X_4)$ and $(X_1 \cdot X_4) \cdot (X_2 \cdot X_3)$ respectively. We define y_u analogously: it is the product of all $y_{i,j} \cdot y_{j,i}$ where (i, j) is a matching parenthesis pair of u. So, if $u =$ "(())" $\in$ Dyck(4), then $y_u = y_{1,4} y_{4,1} y_{2,3} y_{3,2}$. The following observation then follows immediately from the recursive definition of $f_{i,j}$.

▶ **Observation 13.** *For every interval $[i, j] \subseteq [d]$ where $j - i$ is odd, there exist constants $c_u \in \mathbb{F}$ corresponding to every $u \in$ Dyck($j - i + 1$) such that $f_{i,j}(X, Y) = \sum_{u \in \text{Dyck}(j-i+1)} c_u y_u IP_u$[15]. Moreover, if $\mathbb{F}$ has characteristic zero, then every $c_u \neq 0$.*

▷ Claim 14. Let d be a positive even integer. For any $w \in \{-k, k\}^d$ with $w_{[d]} = 0$ (i.e., $|\mathcal{P}_w| = |\mathcal{N}_w|$), there exists a Dyck work $u \in$ Dyck(d) such that for every matching parenthesis pair (i, j) of u, either $i \in \mathcal{P}_w$ and $j \in \mathcal{N}_w$, or $i \in \mathcal{N}_w$ and $j \in \mathcal{P}_w$.

Proof. We prove this by induction on d. The base case $d = 2$ is trivial as there is only a single matching parenthesis pair $(1, 2)$ for which the given condition must indeed hold. Now, suppose $d > 2$ and $w \in \{-k, k\}^d$ is a given word with $w_{[d]} = 0$. Let us refer to $\mathcal{P}_w$ and $\mathcal{N}_w$ as the two "parts" of w and take cases on the membership of 1 and d in these sets.

[15] Strictly speaking, the indices within IP_u and y_u here need to be "translated" appropriately to suit the interval $[i, j]$ (which may not necessarily be $[1, j - i + 1]$).

Case 1: 1 and d are in different parts. By the induction hypothesis, there is a Dyck word $u' \in \mathrm{Dyck}(d-2)$ (corresponding to the subset $\{2,\ldots,d-1\}$ of indices) for which the desired condition holds. Hence, we can simply define u to be the string "(u')" and the claim follows.

Case 2: 1 and d are in the same part. Notice that there exists an index r such that both $w_{[1,r]}$ and $w_{[r+1,d]}$ are 0. Then, we can define u to be the concatenation of the two Dyck words that the induction hypothesis yields for the intervals $[1,r]$ and $[r+1,d]$ respectively and the claim follows. ◁

▶ **Lemma 15.** *Let $n = 2^k$ and $d \le n$ be an even integer. Over any field $\mathbb{F}$ of characteristic zero, the polynomial $F_{n,d} = f_{1,d} \in \mathbb{F}_{\mathrm{sm}}[X,Y]$ as defined above satisfies the following: For any $w \in \{-k,k\}^d$ with $w_{[d]} = 0$, $\mathcal{M}_w(F_{n,d})$ is full-rank when viewed as a matrix over the field $\mathbb{F}(Y)$, the field of rational functions over the Y variables.*

Proof. Fix a word $w \in \{-k,k\}^d$ with $w_{[d]} = 0$ and let $s \in \mathrm{Dyck}(d)$ be a Dyck word as given by Claim 14. By Observation 13, we know that $F = f_{1,d}$ has the form $\sum_{u\in\mathrm{Dyck}(d)} c_u y_u IP_u$. Consider the polynomial f obtained by plugging in $y_{i,j} = y_{j,i} = 0$ for every i,j such that (i,j) is *not* a matching parenthesis pair of s, and $y_{i,j} = y_{j,i} = 1$ for every i,j such that (i,j) *is* a matching parenthesis pair of s. Observe that the only surviving term from F in f is the one indexed by s. Therefore, to argue that $\mathcal{M}_w(F)$ is full-rank over $\mathbb{F}(Y)$[16], it suffices to show that $\mathcal{M}_w(c_s IP_s)$ is full-rank. As $c_s \neq 0$ by Observation 13, this follows from Observation 8 (the matrix of the inner product gadget has full rank), Claim 14 (w "splits" every matching parenthesis pair of s), and Claim 7 (the multiplicativity of relrk_w). ◀

Let us return to the proof of the last two items of Theorem 10. Let C be a set-multilinear formula of product depth Δ of size s computing $F_{n,d}(X)$ (now interpreted as a formula over the field $\mathbb{F}(Y)$). Suppose $s < d^{\frac{d^{1/\Delta}}{24\Delta}}$. Then, by Lemma 11, with probability at least $1 - d^{-\frac{d^{1/\Delta}}{24\Delta}}$,

$$\mathrm{relrk}_w(C) \le s \cdot 2^{-\frac{k d^{1/\Delta}}{20}}.$$

But now, we can condition on the event that $w_{[d]} = 0$ (which occurs with probability $\Theta(\frac{1}{\sqrt{d}})$) to establish the existence of a word $w \in \{-k,k\}^d$ with $w_{[d]} = 0$ such that w satisfies $\mathrm{relrk}_w(C) \le s \cdot 2^{-\frac{k d^{1/\Delta}}{20}}$. This is because of the asymptotic bound $\frac{1}{\sqrt{d}} \gg d^{-\frac{d^{1/\Delta}}{24\Delta}}$, which follows from the given constraints on the parameters d, Δ. Therefore, by Lemma 15,

$$s \ge 2^{\frac{k d^{1/\Delta}}{20}} \cdot \mathrm{relrk}_w(C) = n^{\frac{d^{1/\Delta}}{20}}$$

which contradicts the assumption that $s < d^{\frac{d^{1/\Delta}}{24\Delta}}$. Thus, we conclude that $s \ge d^{\frac{d^{1/\Delta}}{24\Delta}} = d^{\Omega(d^{1/\Delta}/\Delta)}$.

Similarly, to see the final item of Theorem 10, let F be a set-multilinear formula of size s computing $F_{n,d}$ (now interpreted as a formula over the field $\mathbb{F}(Y)$). Suppose $s < d^{\frac{\log d}{120}}$. Then, by Lemma 12, with probability at least $1 - d^{-\frac{\log d}{120}}$,

$$\mathrm{relrk}_w(F) \le s \cdot 2^{-\frac{k \log d}{20}}.$$

[16] We need to show that its determinant – a polynomial in $\mathbb{F}[Y]$ – is non-zero.

But now, we can condition on the event that $w_{[d]} = 0$ (which occurs with probability $\Theta(\frac{1}{\sqrt{d}})$) to establish the existence of a word $w \in \{-k, k\}^d$ with $w_{[d]} = 0$ such that w satisfies $\text{relrk}_w(F) \leq s \cdot 2^{-\frac{k \log d}{20}}$. This is because of the trivial asymptotic bound $\frac{1}{\sqrt{d}} \gg d^{-\frac{\log d}{120}}$. Therefore, again by Lemma 15,

$$s \geq 2^{\frac{k \log d}{20}} \cdot \text{relrk}_w(F) = n^{\frac{\log d}{20}}$$

which contradicts the assumption that $s < d^{\frac{\log d}{120}}$. Thus, we conclude that $s \geq d^{\frac{\log d}{120}} = d^{\Omega(\log d)}$.

4 A Hard Set-multilinear Polynomial in VBP

4.1 Arc-partition Measure Description

This subsection is adapted from Section 2 of [8]. Let $n = 2^k$, $d \leq n$ be an even integer, and let $X = (X_1, X_2, \ldots, X_d)$ be a collection of disjoint sets of n variables each. An *arc-partition* will be a special kind of *symmetric* word $w \in \{-k, k\}^d$ (i.e., a one-to-one map Π from X to $\{-k, k\}^d$). For the purpose of this subsection, the reader can even choose to think of the alphabet of w as $\{-1, 1\}$ (i.e., one "positive" and one "negative" value) – we use $k, -k$ only to remain consistent with Definition 5.

Identify X with the set $\{1, 2, \ldots, d\}$ in the natural way. Consider the d-cycle graph, i.e., the graph with nodes $\{1, 2, \ldots, d\}$ and edges between i and $i + 1$ modulo d. For two nodes $i \neq j$ in the d-cycle, denote by $[i, j]$ the arc between i, j, that is, the set of nodes on the path $\{i, i + 1, \ldots, j - 1, j\}$ from i to j in d-cycle. First, define a distribution $\mathcal{D}_P$ on a family of pairings (a list of disjoint pairs of nodes in the cycle) as follows. A random pairing is constructed in $d/2$ steps. At the end of step $t \in [d/2]$, we shall have a pairing $(P_1, \ldots, P_t)$ of the arc $[L_t, R_t]$. The size of $[L_t, R_t]$ is always $2t$. The first pairing contains only $P_1 = \{L_1, R_1\}$ with $L_1 = 1$ and $R_1 = 2$. Given $(P_1, \ldots, P_t)$ and $[L_t, R_t]$, define the random pair $P_t + 1$ (independently of previous choices) by

$$P_{t+1} = \begin{cases} \{L_t - 2, L_t - 1\} & \text{with probability } 1/3 \\ \{L_t - 1, R_t + 1\} & \text{with probability } 1/3 \\ \{R_t + 1, R_t + 2\} & \text{with probability } 1/3 \end{cases}$$

Define

$$[L_{t+1}, R_{t+1}] = [L_t, R_t] \cup P_{t+1}.$$

So, L_{t+1} is either $L_t - 2$, $L_t - 1$ or L_t, each value is obtained with probability $1/3$, and similarly (but not independently) for R_{t+1}.

The final pairing is $P = (P_1, P_2, \ldots, P_{d/2})$. Denote by $P \sim \mathcal{D}_P$ a pairing distributed according to $\mathcal{D}_P$.

Once a pairing P has been obtained, a word $w \in \{-k, k\}^d$ is obtained by simply randomly assigning $+k$ and $-k$ to the indices of any pair P_i. More formally, for every $t \in [n/2]$, if $P_t = \{i_t, j_t\}$, let with probability $1/2$, independently of all other choices,

$$w_{i_t} = +k \text{ and } w_{j_t} = -k,$$

and with probability $1/2$,

$$w_{i_t} = -k \text{ and } w_{j_t} = +k.$$

Denote by $w \sim \mathcal{D}$ a word in $\{-1, 1\}^n$ that is sampled using this procedure. We call such a word an *arc-partition*. For a pair $P_t = \{i_t, j_t\}$, we refer to i_t and j_t as *partners*.

▶ **Definition 16** (Arc-full-rank). *We say that a polynomial f that is set-multilinear over $X = (X_1, \ldots, X_d)$ is* ***arc-full-rank*** *if for every arc-partition $w \in \{-k, k\}^d$, $\text{relrk}_w(f) = 1$.*

4.2 Construction of an Arc-full-rank Polynomial

Below, we describe a simple construction of an ABP that computes an arc-full-rank set-multilinear polynomial. The high-level idea is to construct an ABP in which every path between start-node and end-node corresponds to a specific execution of the random process which samples arc-partitions. Each node in the ABP corresponds to an arc $[L, R]$, which sends an edge to each of the nodes $[L-2, R], [L-1, R+1]$ and $[L, R+2]$. The edges have specially chosen labels that help guarantee full rank with respect to every arc-partition. For simplicity of presentation, we allow the edges of the program to be labeled by degree three polynomials in three variables. This assumption can be easily removed by replacing each edge with a constant-size ABP computing the corresponding degree three polynomial.

Formally, the nodes of the program are even-size arcs in the d-cycle, d an even integer. The start-node of the program is the empty arc $\emptyset$ and the end-node is the whole cycle $[d]$ (both are "special" arcs). Let $X = (X_1, \ldots, X_d)$ be a collection of sets of variables where each $|X_i| = n$, and similarly, let $Y = (Y_1, \ldots, Y_d)$ be a distinct collection of sets of variables where each $|Y_i| = n$ (we shall refer to the Y-variables as *auxiliary* variables). For i and j in $\{1, \ldots, d\}$, let $X_i \cdot X_j$ denote the inner-product quadratic form $\sum_{k=1}^{n} x_{ik}x_{jk}$. Here, we shall assume that $X_i = \{x_{i,1}, \ldots, x_{i,n}\}$ and $Y_i = \{y_{i,1}, \ldots, y_{i,n}\}$.

Construct the branching program by connecting a node/arc of size $2t$ to three nodes/arcs of size $2t+2$. For $t = 1$, there is just one node $[1, 2]$, and the edge from start-node to it is labeled $y_{1,2}y_{2,1}(X_0 \cdot X_1)$. For $t > 1$, the node $[L, R] \supset [1, 2]$ of size $2t < d$ is connected to the three nodes: $[L-2, R], [L-1, R+1]$, and $[L, R+2]$. (It may be the case that the three nodes are the end-node.) The edge labeling is:

- The edge between $[L, R]$ and $[L-2, R]$ is labeled $y_{L-2,L-1}y_{L-1,L-2}(X_{L-2} \cdot X_{L-1})$.
- The edge between $[L, R]$ and $[L-1, R+1]$ is labeled $y_{L-1,R+1}y_{R+1,L-1}(X_{L-1} \cdot X_{R+1})$.
- The edge between $[L, R]$ and $[L, R+2]$ is labeled $y_{R+1,R+2}y_{R+2,R+1}(X_{R+1} \cdot X_{R+2})$.

Consider the ABP thus described, and the polynomial $G = G_{n,d}$ it computes. For every path γ from start-node to end-node in the ABP, the list of edges along γ yields a pairing P; every edge e in γ corresponds to a pair $P_e = \{i_e, j_e\}$ of nodes in d-cycle. Thus,

$$G = \sum_{\gamma} \prod_{e=\{i_e,j_e\}\in\gamma} y_{i_e,j_e}y_{j_e,i_e} \cdot (X_{i_e} \cdot X_{j_e}). \tag{1}$$

where the sum is over all paths γ from start-node to end-node.

▶ Remark 17. There is in fact a one-to-one correspondence between pairings P and such paths γ (this follows by induction on t). Note that this is true only because pairings are tuples i.e., they are *ordered* by definition. Otherwise, it is of course still possible to obtain the same *set* of pairs in a given pairing using multiple different orderings. The sum defining G can be thought of, therefore, as over pairings P.

The following statement summarizes the main useful property of G.

▶ **Lemma 18.** *Over any field $\mathbb{F}$ of characteristic zero, the polynomial $G = G_{n,d}$ defined above is arc-full-rank as a set-multilinear polynomial in the variables X over the field $\mathbb{F}(Y)$ of rational functions in Y.*

Proof. Let $w \sim \mathcal{D}$ be an arc-partition. We want to show that $\mathcal{M}_w(G)$ has full rank. The arc-partition w is defined from a pairing $P = (P_1, \ldots, P_{d/2})$ (though as discussed in Remark 17, there could be multiple such P). The pairing P corresponds to a path γ from start-node to end-node. Consider the polynomial f that is obtained by setting every $y_{i,j} = y_{j,i} = 0$ in F

such that $\{i,j\}$ is not a pair in P, and setting every $y_{i,j} = y_{j,i} = 1$ for every pair $\{i,j\}$ in P. Then, it is easy to see that the only terms that survive in (1) correspond to paths (and in turn, pairings) which have the same underlying *set* of pairs as P. As a consequence, f is simply some non-zero constant times a polynomial which is full-rank. $M_w(f)$ being full rank then implies that $M_w(G)$ is also full-rank[17]. ◀

4.3 Bounding relrk_w for Small Set-multilinear Formulas

As discussed in Section 1, the high-level strategy to prove Theorem 1 is to show that the relative rank (with respect to arc-partitions) of our hard polynomial is large (as already established in Lemma 18), while it is small for (small enough) set-multilinear formulas. The remainder of the section is devoted to establishing the latter. Before moving on to it, we shall first state the following more precise and general version of Theorem 1. We also incorporate Remark 2 here and show our lower bound for any degree $d \leq n$. Theorem 1 follows from the special case $d = n$.

▶ **Theorem 19.** *Let $n = 2^k$, and suppose $d \leq n$ be an even integer that is large enough[18], and $1 \leq \Delta \leq \log d / \log\log d$ be any integer. Let X_i, Y_i denote the sets of n variables $\{x_{i,j} : j \in [n]\}$ and $\{y_{i,j} : j \in [n]\}$ respectively and let X, Y be the tuples $(X_1, \ldots, X_d)$ and $(Y_1, \ldots, Y_d)$. Then,*

- *there is a $\mathrm{poly}(n,d)$-size branching program computing $G_{n,d}$ as defined above whose every internal node computes a set-multilinear polynomial,*
- *any set-multilinear formula of product-depth Δ computing $G_{n,d}$ must have size at least $d^{\Omega(d^{1/\Delta}/\Delta)}$.*
- *Further, any set-multilinear formula of arbitrary product-depth computing $G_{n,d}$ must have size at least $d^{\Omega(\log d)}$.*

The following couple of lemmas formalize the high-level idea mentioned before the statement of Theorem 19 – they correspond to the low-depth case and general depth case respectively. Most of the remainder of this section is devoted to the proof of Lemma 20; Lemma 22 has a similar (and in fact, easier) proof and for this reason, we only provide a sketch that is deferred to the appendix.

▶ **Lemma 20.** *Let $\mathbb{K}$ be any field and let $X_1, \ldots, X_D$ be sets of n distinct variables each. Let C be a set-multilinear formula over $\mathbb{K}$ of constant product-depth $\Delta \geq 1$ of size at most s which computes a polynomial over $\mathbb{K}$ that is set-multilinear with respect to the partition $(X_{i_1}, \ldots, X_{i_d})$ where $1 \leq i_1 < \cdots < i_d \leq D$ and each $|X_i| = n$. Let $w \sim \mathcal{D}$ be an arc-partition sampled from $\{-k,k\}^D$. Then, we have*

$$\mathrm{relrk}_w(C) \leq s \cdot 2^{-\frac{kd^{1/\Delta}}{2000}}$$

with probability at least $1 - s \cdot d^{-\frac{d^{1/\Delta}}{10^7 \Delta}}$.

▶ Remark 21. Note that in the statement above, we are abusing notation and overloading the relrk_w notation – assume that $\mathrm{relrk}_w(C)$ is defined in the obvious projective manner i.e., if $S = \{i_1, \ldots, i_d\}$, then $\mathrm{relrk}_w(C) := \mathrm{relrk}_{w|_S}(C)$ where $w|_S$ is as defined in Definition 6.

[17] This argument is the same as in the proof of Lemma 15.
[18] We only need d to be larger than some absolute constant.

▶ **Lemma 22.** *Let $\mathbb{K}$ be any field and let $X_1, \ldots, X_d$ be sets of n distinct variables each. Let F be a set-multilinear formula over $\mathbb{K}$ of size at most s which computes a polynomial over $\mathbb{K}$ that is set-multilinear with respect to the partition $(X_1, \ldots, X_d)$ where each $|X_i| = n$. Let $w \sim \mathcal{D}$ be an arc-partition sampled from $\{-k, k\}^d$. Then, we have*

$$\mathrm{relrk}_w(F) \leq s \cdot 2^{-\frac{k \log d}{2000}}$$

with probability at least $1 - s \cdot d^{-\frac{\log d}{10^7 \Delta}}$.

Before moving on to the technical core of this section (the proof of Lemma 20), let us finish the proof of Theorem 19.

Proof of Theorem 19 given Lemmas 20 and 22. Note that the first item follows immediately from the definition of $G_{n,d}$ (see (1)). Let us prove the last two items of Theorem 19. Let C be a set-multilinear formula of product depth Δ of size s computing $G_{n,d}(X)$ (now interpreted as a formula over the field $\mathbb{F}(Y)$). Suppose $s < d^{\frac{d^{1/\Delta}}{2 \times 10^7 \Delta}}$. Then, by Lemma 20, for an arc-partition $w \sim \mathcal{D}$ sampled from $\{-k, k\}^d$, it follows that with probability at least $1 - d^{-\frac{d^{1/\Delta}}{2 \times 10^7 \Delta}}$,

$$\mathrm{relrk}_w(C) \leq s \cdot 2^{-\frac{k d^{1/\Delta}}{2000}}.$$

Fix such an arc-partition w. By Lemma 18, we have

$$s \geq 2^{\frac{k d^{1/\Delta}}{2000}} \cdot \mathrm{relrk}_w(C) = n^{\frac{d^{1/\Delta}}{2000}}$$

which contradicts the assumption that $s < d^{\frac{d^{1/\Delta}}{2 \times 10^7 \Delta}}$. Thus, we conclude that $s \geq d^{\frac{d^{1/\Delta}}{2 \times 10^7 \Delta}} = d^{\Omega(d^{1/\Delta}/\Delta)}$.

Similarly, to see the final item of Theorem 19, let F be a set-multilinear formula of size s computing $G_{n,d}$ (now interpreted as a formula over the field $\mathbb{F}(Y)$). Suppose $s < d^{\frac{\log d}{2 \times 10^7}}$. Then, for an arc-partition $w \sim \mathcal{D}$ sampled from $\{-k, k\}^d$, by Lemma 22, with probability at least $1 - d^{-\frac{\log d}{2 \times 10^7}}$,

$$\mathrm{relrk}_w(F) \leq s \cdot 2^{-\frac{k \log d}{2000}}.$$

Fix such an arc-partition w. Therefore, again by Lemma 18,

$$s \geq 2^{\frac{k \log d}{2000}} \cdot \mathrm{relrk}_w(F) = n^{\frac{\log d}{2000}}$$

which contradicts the assumption that $s < d^{\frac{\log d}{2 \times 10^7}}$. Thus, we conclude that $s \geq d^{\frac{\log d}{2 \times 10^7}} = d^{\Omega(\log d)}$. ◀

The essential ingredient in the proof of Lemma 20 is a combinatorial proposition which we will call the "many violations lemma". As alluded to in Section 1.5, this is a modification of a corresponding statement in [8] (Lemma 4.1). However, because we are working in the low-depth setting (as opposed to [8]) and because we are seeking such strong and near-optimal lower bounds, we need to make significant changes – this includes introducing new conceptual arguments to tighten the analysis. To state this lemma, we shall reproduce some of the definitions made in Section 4 of [8].

Again, we identify the set of variables $X = (X_1, \ldots, X_D)$ with the D-cycle $\{1, \ldots, D\}$, where addition is modulo D. Let S be a collection of disjoint subsets of the cycle to K parts, namely, $S = (S_1, \ldots, S_K)$ where each $S_k \subset \{1, \ldots, D\}$ and $S_k \cap S_{k'} = \emptyset$ for all $k \neq k'$ in $[K]$.

We also think of $[K]$ as a set of colors, and of S as a (partial) K-coloring of some d nodes of the cycle, where $d = |S_1| + \cdots + |S_K|$. We shall refer to the nodes in the D-cycle outside of S as *uncolored*.

For a pairing P, define the number of k-violations by

$$V_k(P) = \{P_t \in P : |P_t \cap S_k| = 1\}.$$

In words, it is the set of pairs in which one color is k and the other color is different. Fix $\varepsilon = 1/1000$ and denote

$$G(P) = \{k \in [K] : |V_k(P)| \geq d^{\varepsilon}\}.$$

We do not include S as a subscript in these two notations since S will be known from the context (and will be fixed throughout most of the discussion). The next crucial lemma shows that for every fixed non-redundant K-coloring of the cycle, a random pairing has, with high probability, many colors with many violations.

▶ **Lemma 23** (Many Violations Lemma)**.** *For all large enough d and for all integers K in the range $[2d^{\frac{1}{\Delta+1}}/3, 2d^{\frac{1}{\Delta+1}}]$ the following holds: Let $S = (S_1, \ldots, S_K)$ be a collection of disjoint subsets of the D-cycle and suppose that $|S_k| \geq d^{\frac{\Delta}{\Delta+1}}/2$ for all $k \in [K]$. Then,*

$$\mathbb{P}[G(P) \leq K/1000] \leq d^{-K/500},$$

where $P \sim \mathcal{D}_P$.

▶ Remark 24. Other than the differences in parameter ranges, one key difference between the statement above from Lemma 4.1 in [8] is the loosening of the requirement that S be a *partition* of the D-cycle. Note that here, we only demand that S be a collection of disjoint subsets (i.e., some nodes are allowed to remain *uncolored*) – this requirement is indeed key for the inductive proof of Lemma 20 to go through.

Before proving Lemma 23, let us next see that the many violations lemma suffices to prove the relative rank upper bound on low-depth set-multilinear formulas.

Proof of Lemma 20 given Lemma 23. We prove the statement by induction on Δ. Identify the set $\{i_1, \ldots, i_d\}$ with $[d]$.

If $\Delta = 1$, then $C = C_1 + \cdots + C_t$ where each C_i is a product of linear forms. So, for all $i \in [t]$, by Claim 7,

$$\mathrm{relrk}_w(C_i) = \prod_{i=1}^{d} 2^{-\frac{1}{2}|w_j|} = 2^{-\frac{kd}{2}}$$

where in the last step, we used the observation that regardless of the choice of w, $|w_j| = k$ for all $j \in [n]$. Hence, by the sub-additivity of relrk_w, with probability 1, we have

$$\mathrm{relrk}_w(C) \leq s \cdot 2^{-\frac{kd}{2}} \leq s \cdot 2^{-\frac{kd}{2000}}.$$

Next, we assume the statement is true for all formulas of product-depth $\leq \Delta$. Let C be a formula of product-depth $\Delta + 1$. So, C is of the form $C = C_1 + \cdots + C_t$. Using a similar terminology to that in [23] and [22], we say that a sub-formula C_i of size s_i is of type 1 if one of its factors has degree at least $T_\Delta = d^{\frac{\Delta}{\Delta+1}}$, otherwise we say it is of type 2.

Suppose $C_i = C_{i,1} \cdot \cdots \cdot C_{i,t_i}$ is of type 1 with, say, $C_{i,1}$ having degree at least T_Δ. Let $w^{i,1}$ be the corresponding word i.e., $w^{i,1} = w|_{S_1}$ if $C_{i,1}$ is set-multilinear with respect to $S_1 \subsetneq [d]$. If it has size $s_{i,1}$, then since it has product-depth at most Δ, it follows by induction that

$$\mathrm{relrk}_w(C_i) \leq \mathrm{relrk}_{w^{i,1}}(C_{i,1}) \leq s_{i,1} \cdot 2^{-\frac{kT_\Delta^{1/\Delta}}{2000}} \leq s_i \cdot 2^{-\frac{kd^{1/(\Delta+1)}}{2000}}$$

with probability at least

$$1 - s_{i,1} \cdot T_\Delta^{-\frac{T_\Delta^{1/\Delta}}{10^7\Delta}} \geq 1 - s_i \cdot d^{-\frac{d^{1/(\Delta+1)}}{10^7\Delta} \cdot \frac{\Delta}{\Delta+1}} = 1 - s_i \cdot d^{-\frac{d^{1/(\Delta+1)}}{10^7(\Delta+1)}}.$$

Now suppose that $C_i = C_{i,1} \cdot \cdots \cdot C_{i,t_i}$ is of type 2 i.e., each factor $C_{i,j}$ has degree $< T_\Delta$. Note that this forces $t_i > d/T_\Delta = d^{\frac{1}{\Delta+1}}$. As the formula is set-multilinear, $(S_1, \ldots, S_{t_i})$ form a partition of $[d]$ where each $C_{i,j}$ is set-multilinear with respect to $(X_\ell)_{\ell \in S_j}$ and C_i is set-multilinear with respect to $(X_\ell)_{\ell \in S}$. Let $w^{i,1}, \ldots, w^{i,t_i}$ be the corresponding decomposition, whose respective sums are denoted simply by $w_{S_1}, \ldots, w_{S_{t_i}}$.

From the properties of relrk_w (Claim 7), we have

$$\mathrm{relrk}_w(C_i) = \prod_{j=1}^{t_i} \mathrm{relrk}_{w^{i,j}}(C_{i,j}) \leq \prod_{j=1}^{t_i} 2^{-\frac{1}{2}|w_{S_j}|} = 2^{-\frac{1}{2}\sum_{j=1}^{t_i}|w_{S_j}|},$$

from which we observe that the task of upper bounding $\mathrm{relrk}_w(C)$ can be reduced to the task of lower bounding the sum $\sum_{j=1}^{t_i} |w_{S_j}|$, which is established in the following claim. For the sake of convenience, the choice of the alphabet for w below is scaled down to $\{-1, 1\}$.

▷ Claim 25. For large enough d, suppose $(S_1, \ldots, S_K)$ is a partition of $[d]$ such that each $|S_j| < T_\Delta = d^{\frac{\Delta}{\Delta+1}}$. Then, we have

$$\mathop{\mathbb{P}}_{w \sim \mathcal{D}} \left[\sum_{j=1}^{K} |w_{S_j}| < \frac{d^{1/(\Delta+1)}}{2000} \right] \leq d^{-\frac{d^{1/(\Delta+1)}}{10^7}}.$$

Here, $\mathcal{D}$ refers to the original distribution i.e., an arc-partition over the D-cycle.

Proof. We first show that without loss of generality, we may assume that each S_j has size "roughly" T_Δ. To see this, we apply the following *clubbing* procedure to the sets in the partition $(S_1, \ldots, S_K)$:

- Start with the given partition $(S_1, \ldots, S_K)$. At each step in the procedure, we shall "club" two of the sets in the partition according to the following rule.
- If there are two distinct sets S' and S'' in the current partition each of size $< T_\Delta/2$, we remove both of them and add their union $S' \cup S''$ to the partition.
- If the rule above is no longer applicable, then we have at most one set in the current partition of size $< T_\Delta/2$. If there is none, then we halt the procedure here. Otherwise, we union this set with any one of the other sets and then halt.

After the clubbing procedure, we are left with a partition $(S'_1, \ldots, S'_{K'})$ of $[d]$ such that $\frac{T_\Delta}{2} \leq |S'_j| \leq \frac{3T_\Delta}{2}$ for each $j \in [K']$, also implying that $\frac{2d^{1/(\Delta+1)}}{3} \leq K' \leq 2d^{1/(\Delta+1)}$. Through a repeated use of the triangle inequality, we see that $\sum_{j=1}^{K'} |w_{S'_j}| \leq \sum_{j=1}^{K} |w_{S_j}|$. Therefore, upper bounding the latter sum is a "smaller" event than upper bounding the former sum. Hence, it suffices to prove the statement of the claim with the assumption that $\frac{T_\Delta}{2} \leq |S_j| \leq \frac{3T_\Delta}{2}$ for each $j \in [K]$ (we henceforth drop the primed notation).

Applying Lemma 23 to the tuple $(S_1, \ldots, S_K)$, we obtain that

$$\mathbb{P}[G(P) \leq K/1000] \leq d^{-K/500}.$$

The idea is to condition on the high probability event that $G(P) > K/1000$. Fix a pairing P with this property. Consider an ordering σ of the colors in $G(P)$. A color k is said to be *bright* with respect to an ordering if there are at least $d^{\varepsilon}/2$ nodes x of color k such that either the partner of x is uncolored or its partner is colored using a color that appears *after* k in the ordering σ. Call an ordering σ of the nodes in $G(P)$ *good* if there are at least $|G(P)|/2$ bright colors with respect to σ. The observation is that for any ordering σ of the colors, either σ itself is good, or its reverse is good. We conclude that given any pairing P, there exists a good ordering of $G(P)$. Fix any such good ordering and let $H(P)$ be the collection of bright colors with respect to this ordering.

Next, notice that if the sum $\sum_{j=1}^{K} |w_{S_j}|$ is at most $\frac{d^{1/(\Delta+1)}}{2000}$, then so is the sum $\sum_{k \in H(P)} |w_{S_k}|$. Let $K' = |H(P)|$ (which is at least $K/2000$ if $G(P) > K/1000$). View the sampling of Π from P as happening in a specific order, according to the order of $k_1, k_2, \ldots, k_{K'}$: First define Π on pairs with at least one point with color k_1, then define Π on remaining pairs with at least one point with color k_2, and so forth. When finished with $k_1, \ldots, k_{K'}$, continue to define Π on all other pairs.

Conditioned on the event that $G(P) > K/1000$, this implies that $|w_{S_j}| \leq 1$ for each $j \in H(P)$. For every $j \in H(P)$, define E_j to be the event that $|w_{S_{k_j}}| \leq 1$. By choice, conditioned on $E_1, \ldots, E_{j-1}$, there are at least $d^{\varepsilon}/2$ pairs P_t so that $|P_t \cap S_{k_j}| = 1$ that are not yet assigned a "positive" or "negative" sign. For every such P_t, the element in $P_t \cap S_{k_j}$ is assigned a positive sign with probability $1/2$, and is independent of any other $P_{t'}$. The probability that a binomial random variable B over a universe of size $U \geq d^{\varepsilon}/2$ and marginals $1/2$ obtains any specific value is at most $O(U^{-1/2}) = O(d^{-\varepsilon/2})$. Hence, for all $j \in H(P)$, by the union bound,

$$\mathbb{P}[E_j | E_1, \ldots, E_{j-1}, P] \leq \mathbb{P}_B[U/2 - 1 \leq B \leq U/2 + 1] \leq O(3 \cdot d^{-\varepsilon/2}) \leq d^{-\varepsilon/4}.$$

Therefore,

$$\mathbb{P}[|w_{S_{k_j}}| \leq 1 \text{ for all } j \in H(P)] \leq \mathbb{E}[d^{-\varepsilon |H(P)|/4} | G(P) > K/1000] + d^{-K/500} \leq d^{-K/10^7}.$$

Finally, we note that

$$\mathbb{P}_{w \sim \mathcal{D}}\left[\sum_{j=1}^{K} |w_{S_j}| < \frac{d^{1/(\Delta+1)}}{2000}\right] \leq \mathbb{P}[|w_{S_{k_j}}| \leq 1 \text{ for all } j \in H(P)]. \qquad \lhd$$

The claim above and the preceding calculation immediately implies that for a sub-formula C_i of type 2,

$$\text{relrk}_w(C_i) \leq s_i \cdot 2^{-\frac{k d^{1/(\Delta+1)}}{2000}}$$

with probability at least $1 - d^{-\frac{d^{1/(\Delta+1)}}{10^7}} \geq 1 - s_i \cdot d^{-\frac{d^{1/(\Delta+1)}}{10^7 (\Delta+1)}}$.

Next, by a union bound over $i \in [t]$ and the sub-additivity property of relrk_w, it follows that

$$\text{relrk}_w(C) \leq \text{relrk}_w(C_1) + \cdots + \text{relrk}_w(C_t) \leq s_1 \cdot 2^{-\frac{k d^{1/(\Delta+1)}}{2000}} + \cdots + s_t \cdot 2^{-\frac{k d^{1/(\Delta+1)}}{2000}} = s \cdot 2^{-\frac{k d^{1/(\Delta+1)}}{2000}}$$

with probability at least $1 - s \cdot d^{-\frac{d^{1/(\Delta+1)}}{10^7 (\Delta+1)}}$, which concludes the proof of the lemma. $\blacktriangleleft$

4.4 Proof of the Many Violations Lemma

Fix some collection of disjoint subsets (or a "partial" coloring) $S = (S_1, \ldots, S_K)$ of the D-cycle satisfying the conditions of the lemma. Think of S as a partial function from the D-cycle to the set $[K]$, either assigning a node its color in $[K]$ or leaving it uncolored; $S(i)$ is the color of i. Use the following definition to partition the proof into cases. For a color k, count the number of jumps in it (with respect to the partition S) to be

$$J_k = \{j \in S_k : k = S(j) \neq S(j+1)\},$$

the set of elements j of color k so that $j+1$ is either uncolored or has a color different from k. As mentioned previously, this subsection is adapted from the proof of Lemma 4.1 in [8]. In what follows, we include remarks where we require a more refined analysis than [8] or a different argument to suit the parameter demands of Lemma 23. Overall, we have attempted to provide a more comprehensive and complete exposition to the proof of the many violations lemma.

Case 1: Many colors with many jumps

The high-level idea is that each color with many jumps has many violations because pairs of the form $(j, j+1)$ yield violations as soon as they are constructed.

Assume that for at least $K/2$ colors k, $|J_k| > d^{2\varepsilon}$. Denote by $B \subseteq [K]$ the set of k's that satisfy this inequality. Then, for every k in B, there exists a subset $Q_k \subset J_k$ of size $N = \lceil d^{2\varepsilon} \rceil$. Let

$$Q \coloneqq \bigcup_{k \in B} Q_k.$$

We think of the construction of the (random) pairing P as happening in *epochs*, depending on Q, as follows.

For $t > 0$, define the random variable

$$Q(t) = Q \setminus [L_t - 4, R_t + 4],$$

the set Q after removing a four-neighborhood of $[L_t, R_t]$. For a certain sequence of time steps t, we will define special nodes q_t which lie in this small "cloud" around the arc $[L_t, R_t]$ (i.e., within a distance of 4 on either side of the arc) - it is for these special nodes q_t that the set of pairs (q_t, q_{t+1}) will provide many violations. We now formalize this intuition.

Let $\tau_1 \geq \tau_0 \coloneqq 1$ be the first time t after τ_0 so that the distance between $[L_t, R_t]$ and $Q(\tau_0)$ is at most two. The distance between $[L_{\tau_0}, R_{\tau_0}]$ and $Q(\tau_0)$ is at least five. The size of the arc $[L_t, R_t]$ increases by two at each time step. So, $\tau_1 \geq \tau_0 + 2$. Let q_1 be an element of $Q(\tau_0)$ that is of distance at most two from $[L_{\tau_1}, R_{\tau_1}]$; if there is more than one such q_1, choose arbitrarily. The minimality of τ_1 implies that q_1 is not in $[L_{\tau_1}, R_{\tau_1}]$.

Let $\tau_2 \geq \tau_1$ be the first time t after τ_1 so that the distance between $[L_t, R_t]$ and $Q(\tau_1)$ is at most two. Let q_2 be an element of $Q(\tau_1)$ that is of distance at most two from $[L_{\tau_2}, R_{\tau_2}]$. Define τ_j, q_j for $j > 2$ similarly, until $Q(\tau_j)$ is empty. As long as $|Q(\tau_j)| \geq 8$, we have $|Q(\tau_j + 1)| \geq |Q(\tau_j)| - 8$. This process, therefore, has at least $KN/16$ steps. For $1 \leq j \leq KN/16$, denote by E_j the event that during the time between τ_j and τ_{j+1} the pair $\{q_j, q_j + 1\}$ is added to P. The pair $\{q_j, q_j + 1\}$ is violating color $S(q_j)$. At time τ_j, even conditioned on all the past $P_1, \ldots, P_{\tau_j}$, in at most two steps (and before τ_{j+1}) we can add the pair $\{q_j, q_j + 1\}$ to P. For every j, therefore,

$$\mathbb{P}[E_j | P_1, \ldots, P_{\tau_j}] \geq (1/3)(1/3) = 1/9.$$

Next, let $N' = \lceil KN/960 \rceil$. We want to show that with high probability, for at least N' many j, the event E_j occurs. There are $\binom{\lfloor KN/16 \rfloor}{\lceil KN/960 \rceil}$ many ways of choosing a set of indices j of size $N - N'$. Subsequently,

$$\begin{aligned}\mathbb{P}[\text{there is } j_1, \ldots, j_{N'} \text{ so that } E_{j_1} \cap \cdots \cap E_{j_{N'}}] &\geq 1 - \binom{\lfloor KN/16 \rfloor}{\lceil KN/960 \rceil} \cdot \left(\frac{8}{9}\right)^{N-N'} \\ &\geq 1 - \left(\frac{960e}{16}\right)^{N'} \cdot \left(\frac{8}{9}\right)^{60N'} \\ &\geq 1 - c^{N'}\end{aligned}$$

where $0 < c < 1$ is a universal constant. Finally, we argue that if there do exist $j_1, \ldots, j_{N'}$ for which the events $E_{j_1}, \ldots, E_{j_{N'}}$ occur, then $G(P) \geq K/1000$. To see this, note that the size of every Q_k is N. So, every color k in B can contribute at most N elements to $j_1, \ldots, j_{N'}$. If $G(P) < K/1000$, then at most these many colors can contribute larger than d^{ε} (and up to N elements) - combined, at most $KN/1000$ elements. However, there are at least $K/2 - K/1000$ colors which can contribute only up to d^{ε} elements. Again combined, this is not sufficient to cover the N' elements overall (for large enough d), which is a contradiction. Hence,

$$\mathbb{P}[G(P) \geq K/1000] \geq \mathbb{P}[\text{there is } j_1, \ldots, j_{N'} \text{ so that } E_{j_1} \cap \cdots \cap E_{j'_N}].$$

and the proof follows in this case as $c^{N'} \ll d^{-\Omega(K)}$.

Case 2: Many colors with few jumps

The intuition is that many violations will come from pairs of the form $\{L_t - 1, R_t + 1\}$ in the construction of the pairing. Assume that for at least $K/2$ colors k, $|J_k| \leq d^{2\varepsilon}$. Denote again by $B \subseteq [K]$ the set of k's that satisfy the above inequality. We say that a color k is noticeable in the arc A if

$$d^{\frac{\Delta}{\Delta+1}-4\varepsilon} \leq |S_k \cap A| \leq |A| - d^{\frac{\Delta}{\Delta+1}-4\varepsilon}.$$

▷ Claim 26. There are $K' \geq K/2 - 1$ disjoint arcs $A_1, \ldots, A_{K'}$ so that for every $j \in [K']$,
1. $|A_j| = m = \lfloor d^{\frac{\Delta}{\Delta+1}-3\varepsilon} \rfloor$ and,
2. there is a color k_j in B that is noticeable in A_j.
Moreover, the colors $k_1, \ldots, k_{K'}$ can be chosen to be pairwise distinct.

Proof. For each color k in B, there are at least $d^{\frac{\Delta}{\Delta+1}}/2$ vertices of color k in the D-cycle and at most $d^{2\varepsilon}$ jumps in the color k. Therefore, there is at least one k-monochromatic arc of size at least $d^{\frac{\Delta}{\Delta+1}-2\varepsilon}$. Hence, on the D-cycle, there are such monochromatic arcs $I_{k_1}, \ldots, I_{k_{|B|}}$ for the colors $k_1, \ldots, k_{|B|}$ in B, in this order ($1 < 2 < \cdots < D$).

Consider an arc A of size m included in I_{k_1}. Thus $|S_{k_1} \cap A| = m$. If we "slide" the arc A until it is included in I_{k_2}, then $|S_{k_1} \cap A| = 0$. By continuity, there is an intermediate position for the arc A such that $d^{\frac{\Delta}{\Delta+1}-4\varepsilon} \leq |S_{k_1} \cap A| \leq m - d^{\frac{\Delta}{\Delta+1}-4\varepsilon}$. This provides the first arc A_1 of the claim.

Sliding an arc inside I_{k_2} to inside I_{k_3} shows that there exists an arc A_2 such that $d^{\frac{\Delta}{\Delta+1}-4\varepsilon} \leq |S_{k_2} \cap A_2| \leq m - d^{\frac{\Delta}{\Delta+1}-4\varepsilon}$. The arcs A_1 and A_2 are disjoint: The distance of the largest element of A_1 and the smallest element of I_{k_2} is at most m. The distance of the smallest element of A_2 and the largest element of I_{k_2} is at most m. The size of I_{k_2} is larger than $2m$. Proceed in this way to define $A_3, \ldots, A_{|B|-1}$. ◁

Use Claim 26 to divide the construction of the (random) pairing into *epochs*. Denote by $A^{(0)}$ the family of arcs given by the claim. Let τ_1 be the first time t that the arc $[L_t, R_t]$ hits one of the arcs in $A^{(0)}$. Denote by A_1 that arc that is hit at time τ_1 (break ties arbitrarily). Denote by k_1 the color that is noticeable in A_1. Let σ_1 be the first time t so that A_1 is contained in $[L_t, R_t]$. Let $A^{(1)}$ be the subset of $A^{(0)}$ of arcs that have an empty intersection with $[L_{\sigma_1}, R_{\sigma_1}]$. Similarly, let τ_2 be the first time t after σ_1 that the arc $[L_t, R_t]$ hits one of the arcs in $A^{(1)}$. If there are no arc in $A^{(1)}$, define $\tau_2 = \infty$. Denote by A_2 that arc that is hit at time τ_2. Denote by k_2 the color that is noticeable in A_2. Let σ_2 be the first time t so that A_2 is contained in $[L_t, R_t]$. Let $A^{(2)}$ be the subset of $A^{(1)}$ of arcs that have an empty intersection with $[L\sigma_2, R\sigma_2]$. Define $\tau_j, \sigma_j, A_j, k_j, A^{(j)}$ for $j > 2$ analogously. For every $j \geq 1$, denote by E_j the event that during the time between τ_j and τ_{j+1} the number of pairs added that violate color k_j's at most d^{ε}. (If E_j does not hold, then $|V_{k_j}(P)| \geq d^{\varepsilon}$ and $k_j \in G(P)$. The main part of the proof is summarized in the following proposition, whose proof is deferred to Section 4.5.

▶ **Lemma 27** (Chessboard Lemma). *Let $\delta = 0.10$. For every $j \geq 1$, and any choice of pairs $P_1, \ldots, P_{\tau_j}$,*

$$\mathbb{P}[E_j | P_1, \ldots, P_{\tau_j}, |A^{(j-1)}| \geq 3] \leq d^{-\delta}$$

Given this lemma, let us finish the proof of Lemma 23. Define $K'' = \lfloor K'/10 \rfloor$ and let T denote the event that the number of j's for which $|A^{(j)}| \geq 3$ is at least K''. First, we argue that T occurs with high probability.

For any $j \geq 1$, consider the evolution of the arc $[L_t, R_t]$ between the time steps τ_j (when it first hits arc A_j) and σ_j (when it completely engulfs it). During this epoch, let us call the evolution of $[L_t, R_t]$ *in* the "direction" of A_j as *good* (labelled "G") and *away* from the direction of A_j as *bad* ("B"). To this end, for any time step in this epoch, we can *code* the three possible choices for the evolution of $[L_t, R_t]$ as GG (when the arc is grown *in* the direction of A_j), GB (when it is grown equally on either side), or BB (when it is grown *away* from the direction of A_j). Consequently, the evolution of $[L_t, R_t]$ during this epoch can be realized as a sequence consisting of the symbols G and B.

Consider the sequence s of G's and B's obtained by concatenating the sequences corresponding to all the epochs (ignoring the choices made at time steps that do *not* lie in such epochs, i.e., between τ_j and σ_j for some j - as there is no corresponding notion of a "good" direction outside such epochs). The intuition is that if $|A^{(K'')}| < 3$ (i.e., if T does not occur), then there must be an extremely large number of B's compared to G's (i.e., the arc $[L_t, R_t]$ evolves disproportionately in the *bad* direction) in the concatenated string s, which should occur only with a vanishingly small probability.

Consider the sub-string s' of s that corresponds to the choices made only for the nodes in $A^{(0)} \setminus A^{(K'')}$. Note that there are precisely mK'' many G's in s'. Suppose $|A^{(K'')}| = 2$ for concreteness (the cases $|A^{(K'')}| = 1$ and $|A^{(K'')}| = 0$ are similar). This implies that there are $m(K' - 2 - K'')$ many B's in s'. Since only up to mK'' many of these B's may appear as a result of the evolution making a choice of the form GB, it follows that the evolution of $[L_t, R_t]$ must make a choice of the form BB at least $m(K' - 2 - 2K'')/2$ times out of a possible $m(K'-2)/2$, in order to cover the elements of $A^{(0)} \setminus A^{(K'')}$. Denote $K_1 := (K'-2)/2$. By the union bound, this probability is at most

$$\mathbb{P}[|A^{(K'')}| = 2] \leq \binom{mK_1}{mK''} \cdot \left(\frac{1}{3}\right)^{m(K_1 - K'')} < c_2^{mK''}$$

for some universal constant $0 < c_2 < 1$. Similarly, we have bounds for both $\mathbb{P}[|A^{(K'')}| = 1]$ and $\mathbb{P}[|A^{(K'')}| = 0]$ and it follows that $\mathbb{P}[T] \geq 1 - c^{mK''}$ for some universal constant $0 < c < 1$.

▶ Remark 28. The argument above for showing that T occurs with high probability differs considerably from [8], where the corresponding event is sketched to occur with probability only at least $1 - dc^{m^{1/3}}$, which is not strong enough for our purposes.

Next, note that

$$\mathbb{P}[G(P) < K/1000] \leq \mathbb{P}[G(P) < K/1000 \cap T] + \mathbb{P}[\neg T] \leq \mathbb{P}[G(P) < K/1000|T] + \mathbb{P}[\neg T].$$

If $G(P) < K/1000$, then at least $K/2 - K/1000$ colors in B have at most d^ε many violations. Since $K'' = \lfloor K'/10 \rfloor < K/2 - K/1000$, in particular, there must exist at least $K''/2$ colors within the *first* K'' colors (here we are using the ordering of colors as provided by Claim 26) for which there are at most d^ε many violations. We then obtain the following by conditioning on T, using the union bound.

$$\mathbb{P}[G(P) < K/1000 \cap T] \leq 2^{K''} \max_{H=\{j_1<\cdots<j_{K''/2}\}\subset[K'']} \mathbb{P}[E_{j_1}, \ldots, E_{j_{K''/2}} || A^{(K'')}| \geq 3]$$

For a fixed choice of H, by the chain rule and Lemma 27, we have

$$\begin{aligned}
&\mathbb{P}[E_{j_1} \cap \cdots \cap E_{j_{K''/2}} || A^{(K'')}| \geq 3] \\
&\quad = \mathbb{P}[E_{j_1}|T] \cdot \mathbb{P}[E_{j_2}|E_{j_1} \cap T] \cdot \cdots \cdot \mathbb{P}[E_{j_{K''/2}}|E_{j_{K''/2-1}} \cap \cdots \cap E_{j_1} \cap T] \\
&\quad \leq d^{-\delta K''/2} \leq d^{-0.1K'/20} \leq d^{-K/400}.
\end{aligned}$$

Overall, we conclude that

$$\mathbb{P}[G(P) < K/1000] \leq d^{-K/500}.$$

4.5 Proof of the Chessboard Lemma

To prove Lemma 27, we use a different point of view of the random process. We begin by describing this different view, and later describe its formal connection to the distribution on pairings. This subsection is adapted from Section 5 of [8] and closely follows their argument, though with numerous parameter changes to suit our demands.

The view uses two definitions. One is a standard definition of a two-dimensional random walk, and the other is a definition of a "chessboard" configuration in the plane. The proof of the proposition will follow by analyzing the behavior of the random walk on the "chessboard". Let d be as above and m be as defined in Lemma 26. The random walk W on $\mathbb{N}^2$ is defined as follows. It starts at the origin, $W_0 = (0, 0)$. At every step it move to one of three nodes, independently of previous choices,

$$W_{t+1} = \begin{cases} W_t + (0, 2) & \text{with probability } 1/3 \\ W_t + (1, 1) & \text{with probability } 1/3 \\ W_t + (2, 0) & \text{with probability } 1/3 \end{cases}$$

At time t, the L_1-distance of Wt from the origin is thus $2t$.

The "chessboard" is defined as follows. Let $\alpha_1 : [m] \to \{0, 1\}$ and $\alpha_2 : [2m] \to \{0, 1\}$ be two Boolean functions. The functions α_1, α_2 induce a "chessboard" structure on the board $[m] \times [2m]$. A position in the board $\xi = (\xi_1, \xi_2)$ is colored either white or black. It is colored black if $\alpha_1(\xi_1) \neq \alpha_2(\xi_2)$ and white if $\alpha_1(\xi_1) = \alpha_2(\xi_2)$. We say that the "chessboard" is well-behaved if

1. α_1 is far from constant:

$$d^{\frac{\Delta}{\Delta+1}-4\varepsilon} \leq |\{\xi_1 \in [m] : \alpha_1(\xi_1) = 1\}| \leq m - d^{\frac{\Delta}{\Delta+1}-4\varepsilon}.$$

2. α_1 does not contain many jumps:

$$|\{\xi_1 \in [m-1] : \alpha_1(\xi_1) \neq \alpha_1(\xi_1+1)\}| \leq d^{2\varepsilon}$$

3. α_2 does not contain many jumps:

$$|\{\xi_2 \in [2m-1] : \alpha_2(\xi_2) \neq \alpha_2(\xi_2+1)\}| \leq d^{2\varepsilon}$$

Consider a random walk W on top of the "chessboard" and stop it when reaching the boundary of the board (i.e., when it tries to make a step outside the board $[m] \times [2m]$). We define a good step to be a step of the form $(1,1)$ that lands in a black block. We will later relate good steps to violating edges. Our goal is, therefore, to show that a typical W makes many good steps.

▶ **Lemma 29.** *Let $\delta = 0.10$ and assume the chessboard is well-behaved. The probability that W makes less than $d^{2\varepsilon}$ good steps is at most $d^{-\delta}$.*

We use this lemma to show Lemma 27.

Proof of Lemma 27 given Lemma 29. Recall that A_j is an arc of size $|A_j| = m = \lfloor d^{\frac{\Delta}{\Delta+1}-3\varepsilon} \rfloor$ so that there is a color k_j satisfying

$$d^{\frac{\Delta}{\Delta+1}-4\varepsilon} \leq |S_k \cap A| \leq |A| - d^{\frac{\Delta}{\Delta+1}-4\varepsilon}. \tag{2}$$

Furthermore, condition on $P_1, \ldots, P_{\tau_j}$, $|A^{(j-1)}| \geq 3$. Assume without loss of generality that R_{τ_j} is in A_j (when L_{τ_j} is in A_j, the analysis is similar). The distance of R_{τ_j} from the smallest element of A_j is at most one (the length of "one step to the right" is between zero and two). We now grow the random interval until σ_j, i.e., as long as R_t stays in A_j. At the same time, L_t performs a movement to the left. Since $|A^{(j-1)}| \geq 3$, there are at least $2m$ steps for L_t to take to the left before hitting A_j. There is a one-to-one correspondence between pairings P and random walks W using the correspondence

$$P_{t+1} = \{L_t - 2, L_t - 1\} \longleftrightarrow W_{t+1} = W_t + (0,2),$$

$$P_{t+1} = \{L_t - 1, R_t + 1\} \longleftrightarrow W_{t+1} = W_t + (1,1),$$

$$P_{t+1} = \{R_t + 1, R_t + 2\} \longleftrightarrow W_{t+1} = W_t + (2,0).$$

Define the function α_1 to be 1 at positions of A_j with color k_j, and 0 at the other positions. Set the function α_2 as to describe the color k_j from L_{τ_j} leftward. The "chessboard" is well-behaved by (2) and since k_j is in the set B defined in case 2 of the proof of Lemma 23 (so there are not many jumps for the color k_j). Finally, if W makes a good step, then the corresponding pair added to P violated color k_j . So, if E_j holds for P, then the corresponding W makes less than $d^{2\varepsilon}$ good steps. Formally, by Lemma 23,

$$\mathbb{P}[E_j | P_1, \ldots, P_{\tau_j}, |A^{(j-1)}| \geq 3] \leq \mathbb{P}[W \text{ makes less than } d^{2\varepsilon} \text{ good steps}] \leq d^{-\delta}. \quad \blacktriangleleft$$

Proof of Lemma 29. Define three events E_R, E_C, E_D, all of which happen with small probability, so that every W that is not in their union makes many good steps.

Call a subset of the board of the form $I \times [2m]$ or $[m] \times I$, where I is a sub-interval, a *region*. The width of a region is the size of I. Let R be the set of regions of width at least $d^{4\varepsilon}$. The size of R is at most $2m^2$. For a region r in R, denote by E_r the event that the number of steps of the form $(1,1)$ that W makes in r is less than $d^{2\varepsilon}$ *given* that it makes at least $d^{3\varepsilon}$ steps in r. Denote

$$E_R = \bigcup_{r \in R} E_r$$

To estimate the probability of E_r, note that we can simply apply the Chernoff bound to a sum of $d^{3\varepsilon}$ Bernoulli random variables with $p = 1/3$. By the union bound, we conclude that there is a universal constant $0 < c < 1$ such that

$$\mathbb{P}[E_R] \leq c^{d^{3\varepsilon}}.$$

Denote by H the set of all points in the board with L_1-norm at least $m^{5/8}$. At time T the random walk W is distributed along all points in $\mathbb{N}^2$ of L_1-norm exactly T. The distribution of W on this set is the same as that of a random walk on $\mathbb{Z}$ that is started at 0, and moves at every step to the right with probability $1/3$, stays in place with probability $1/3$ and moves to the left with probability $1/3$. The probability that such a random walk on $\mathbb{Z}$ is at a specific point in $\mathbb{Z}$ at time T is at most $O(T^{-1/2})$. Hence, for every point h in H,

$$\mathbb{P}[W \text{ hits } h] \leq O(m^{-5/16}) \leq m^{-1/4}.$$

Call a point $c = (\xi_1, \xi_2)$ in the board a corner if both (ξ_1, ξ_2) and $(\xi_1 + 1, \xi_2 + 1)$ are of the same color $\kappa \in \{\text{black}, \text{white}\}$, but $(\xi_1 + 1, \xi_2)$ and $(\xi_1, \xi_2 + 1)$ are not of color κ. For a corner c, denote by $\Delta(c)$ the $d^{4\varepsilon}$-neighborhood of c in L_1-metric. Denote by Δ the union over all $\Delta(c)$, for corners c in H. Denote by E_C the event that W hits any point in Δ. Since the board is well-behaved, the number of jumps in each of α_1, α_2 is at most $d^{2\varepsilon}$. Therefore, the number of corners is at most $d^{4\varepsilon}$. By the union bound,

$$\mathbb{P}[E_C] \leq O(d^{4\varepsilon} d^{8\varepsilon} m^{-1/4}) \leq d^{-0.112},$$

where in the last step, we plugged in $\varepsilon = 1/1000$ and used $m \geq d^{1/2-3\varepsilon}$. Next, let $m' = \lceil m^{5/8} \rceil$. Define three (vertical) lines: D_1 is the line $\{m'\} \times [2m]$, D_2 is the line $\{2m'\} \times [2m]$ and D_3 is the line $\{m - m'\} \times [2m]$. Denote by E_D the event that W does not cross the line D_3 before stopped (i.e., hitting the boundary of the board). Chernoff's bound implies that there is a universal constant $0 < c < 1$ for which

$$P[E_D] \leq c^m.$$

To conclude the proof by the union bound, it suffices to show that for every W not in $E_R \cup E_C \cup E_D$, the walk W makes at least $d^{2\varepsilon}$ good steps. Fix such a walk W. Since $W \notin E_D$, we know that W crosses the line D_2.

We consider several cases. Define a *block* to be a maximal monochromatic rectangle in the board. The board is thus partitioned into black blocks and white blocks - which is what led [8] to calling it a "chessboard." We now think of the board $[m] \times [2m]$ as drawn in the plane with $(1,1)$ at the bottom-left corner and $(m, 2m)$ at the upper-right corner.

Case 1: The walk W does not hit any white block after crossing D_1 and before crossing D_2.

In this case, all steps taken in the *region* whose left border is D_1 and right border is D_2 are in a black area. The number of such steps is at least $m^{5/8}/2 \gg d^{3\varepsilon}$. Since $W \notin E_R$, the claim holds.

Case 2: The walk W hits a white block after crossing D_1 and before crossing D_2. Let us label the blocks as follows: we associate every block with a pair $\langle \eta_1, \eta_2 \rangle$ where η_1 is between 1 and the number of jumps in α_1 and η_2 is between 1 and the number of jumps in α_2. So, the label of the "bottom-left" is $\langle 1, 1 \rangle$, the label of the block "above" it is $\langle 1, 2 \rangle$ and the label of the block "to its right" is $\langle 2, 1 \rangle$, etc. There are two sub-cases to consider:

Sub-case 1: At some point after crossing D_1 and before crossing D_3, there are two white blocks of the form $\langle \eta_1, \eta_2 \rangle$, $\langle \eta_1 + 1, \eta_2 + 1 \rangle$ so that W intersects both blocks. Let c be the corner between these two blocks (which must exist by definition). Since $W \notin E_C$, we know that W does not visit $\Delta(c)$. Therefore, W must walk in a black area around $\Delta(c)$. Every path surrounding $\Delta(c)$ has length at least $d^{4\varepsilon}$. Since $W \notin E_R$, the claim holds.

Sub-case 2: At all times after crossing D_1 and before crossing D_3, the walk never moves from a white block $\langle \eta_1, \eta_2 \rangle$ to one of the two white blocks $\langle \eta_1 + 1, \eta_2 + 1 \rangle$, $\langle \eta_1 - 1, \eta_2 - 1 \rangle$. Since $W \notin E_D$, this is indeed the last case. The width of a combinatorial rectangle in the board is the size of its "bottom side" (i.e., the corresponding subset of [m]). Let η be the first white block W hits after crossing D_1. Let Σ be the family of black blocks that are to the right but on the same height as η. Define γ as the maximal width of a rectangle of the form $\sigma \cap [0, m - m_0 - 1] \times [2m]$ over all $\sigma \in \Sigma$. Since the board is well-behaved, it follows (from the first condition) that the total width of the black area on the same height as η is at least $d^{\frac{\Delta}{\Delta+1} - 4\varepsilon}$. Also, since we are in case 2, the left border of η is to the left of D_2. Therefore, the total width of the black area to the right of the left border of η and to the left of D_3, on the same height as η is at least $d^{\frac{\Delta}{\Delta+1} - 4\varepsilon} - 3m'$. Therefore, since the number of jumps is at most $d^{2\varepsilon}$,

$$\gamma \geq (d^{\frac{\Delta}{\Delta+1} - 4\varepsilon} - 3m')/d^{2\varepsilon} \gg d^{4\varepsilon}.$$

Since we are in this sub-case, the walk W must "go through" every black block it hits: it can go from bottom side to upper side or from left side to right side (but not from left side to upper side or from bottom side to right side). Consider the behaviour of W after it hits η: starting from a white block, because $W \notin E_D$, it is guaranteed to cross D_3. Therefore, the color of the block that W "exits" from from each *column* must keep alternating between white and black. For each black block in Σ, therefore, there exists a black block in the same column that W crosses horizontally. Focusing on one such black block of width γ, since $W \notin E_R$, the claim holds. ◀

5 Discussion and Open Problems

We conclude by mentioning some interesting directions for future work.

- The most interesting and natural question is to make the hard polynomial in our main result $\mathrm{IMM}_{n,n}$. This would imply super-polynomial algebraic formula lower bounds. As far as we know, it is conceivable that even the complexity measure of [22] as described in Section 3 could be used to prove the lower bound for the $\mathrm{IMM}_{n,n}$ polynomial. While the relative rank of $\mathrm{IMM}_{n,n}$ itself is low, there might be a suitable "restriction" of it such that for a randomly chosen $w \in \{-k, k\}^n$, with reasonably high probability the restriction has large rank. This could then be used to prove the lower bound for $\mathrm{IMM}_{n,n}$ (using Lemma 11 and Lemma 12). Secondly, we point out that perhaps it is more viable to find an *ordered* set-multilinear branching program (as described in Section 1.4) which can be shown to be *arc-full-rank*. This would also lead to general formula lower bounds.

- The discussion in Section 1.4 raises the question of the relative computational power of the ordered vs general set-multilinear branching program models. Clearly, if it is shown that these classes coincide, then it leads to formula lower bounds via Theorem 1. We would like to note here that in fact, exponential lower bounds are known for the *ordered* model (see [4] for a discussion[19]).

References

1. Manindra Agrawal, Rohit Gurjar, Arpita Korwar, and Nitin Saxena. Hitting-sets for ROABP and sum of set-multilinear circuits. *SIAM J. Comput.*, 44(3):669–697, 2015. doi:10.1137/140975103.
2. Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 67–75. IEEE Computer Society, 2008. doi:10.1109/FOCS.2008.32.
3. Miklós Ajtai. $\sum^1_1$-formulae on finite structures. *Ann. Pure Appl. Log.*, 24(1):1–48, 1983. doi:10.1016/0168-0072(83)90038-6.
4. Vikraman Arvind and S. Raja. Some lower bound results for set-multilinear arithmetic computations. *Chic. J. Theor. Comput. Sci.*, 2016, 2016. URL: http://cjtcs.cs.uchicago.edu/articles/2016/6/contents.html.
5. Peter Bürgisser. Cook's versus valiant's hypothesis. *Theor. Comput. Sci.*, 235(1):71–88, 2000. doi:10.1016/S0304-3975(99)00183-8.
6. Suryajith Chillara, Christian Engels, Nutan Limaye, and Srikanth Srinivasan. A near-optimal depth-hierarchy theorem for small-depth multilinear circuits. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 934–945. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00092.
7. Suryajith Chillara, Nutan Limaye, and Srikanth Srinivasan. Small-depth multilinear formula lower bounds for iterated matrix multiplication with applications. *SIAM J. Comput.*, 48(1):70–92, 2019. doi:10.1137/18M1191567.
8. Zeev Dvir, Guillaume Malod, Sylvain Perifel, and Amir Yehudayoff. Separating multilinear branching programs and formulas. In Howard J. Karloff and Toniann Pitassi, editors, *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19–22, 2012*, pages 615–624. ACM, 2012. doi:10.1145/2213977.2214034.
9. Michael A. Forbes and Amir Shpilka. Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 243–252. IEEE Computer Society, 2013. doi:10.1109/FOCS.2013.34.
10. Hervé Fournier, Nutan Limaye, Guillaume Malod, and Srikanth Srinivasan. Lower bounds for depth-4 formulas computing iterated matrix multiplication. *SIAM J. Comput.*, 44(5):1173–1201, 2015. doi:10.1137/140990280.
11. Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Math. Syst. Theory*, 17(1):13–27, 1984. doi:10.1007/BF01744431.
12. Zeyu Guo and Rohit Gurjar. Improved explicit hitting-sets for roabps. In Jaroslaw Byrka and Raghu Meka, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2020, August 17-19, 2020, Virtual Conference*, volume 176 of *LIPIcs*, pages 4:1–4:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.APPROX/RANDOM.2020.4.

[19] What they term as a "type-width 1" set-multilinear ABP is an *ordered* set-multilinear ABP for us.

13 Johan Håstad. Almost optimal lower bounds for small depth circuits. In Juris Hartmanis, editor, *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20. ACM, 1986. doi:10.1145/12130.12132.

14 Pavel Hrubes and Amir Yehudayoff. Homogeneous formulas and symmetric polynomials. *Comput. Complex.*, 20(3):559–578, 2011. doi:10.1007/s00037-011-0007-3.

15 Neeraj Kayal, Nutan Limaye, Chandan Saha, and Srikanth Srinivasan. An exponential lower bound for homogeneous depth four arithmetic formulas. *SIAM J. Comput.*, 46(1):307–335, 2017. doi:10.1137/151002423.

16 Neeraj Kayal, Vineet Nair, and Chandan Saha. Separation between read-once oblivious algebraic branching programs (roabps) and multilinear depth-three circuits. *ACM Trans. Comput. Theory*, 12(1):2:1–2:27, 2020. doi:10.1145/3369928.

17 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 146–153. ACM, 2014. doi:10.1145/2591796.2591847.

18 Neeraj Kayal, Chandan Saha, and Sébastien Tavenas. An almost cubic lower bound for depth three arithmetic circuits. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 33:1–33:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.33.

19 Neeraj Kayal, Chandan Saha, and Sébastien Tavenas. On the size of homogeneous and of depth-four formulas with low individual degree. *Theory Comput.*, 14(1):1–46, 2018. doi:10.4086/toc.2018.v014a016.

20 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theor. Comput. Sci.*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.

21 Mrinal Kumar and Shubhangi Saraf. On the power of homogeneous depth 4 arithmetic circuits. *SIAM J. Comput.*, 46(1):336–387, 2017. doi:10.1137/140999335.

22 Deepanshu Kush and Shubhangi Saraf. Improved low-depth set-multilinear circuit lower bounds. In Shachar Lovett, editor, *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 38:1–38:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPIcs.CCC.2022.38.

23 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 804–814. IEEE, 2021. doi:10.1109/FOCS52979.2021.00083.

24 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

25 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Comput. Complex.*, 6(3):217–234, 1997. doi:10.1007/BF01294256.

26 Ran Raz. Separation of multilinear circuit and formula size. *Theory Comput.*, 2(6):121–135, 2006. doi:10.4086/toc.2006.v002a006.

27 Ran Raz. Multi-linear formulas for permanent and determinant are of super-polynomial size. *J. ACM*, 56(2):8:1–8:17, 2009. doi:10.1145/1502793.1502797.

28 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *J. ACM*, 60(6):40:1–40:15, 2013. doi:10.1145/2535928.

29 Ran Raz and Amir Yehudayoff. Balancing syntactically multilinear arithmetic circuits. *Comput. Complex.*, 17(4):515–535, 2008. doi:10.1007/s00037-008-0254-0.

30 Ran Raz and Amir Yehudayoff. Lower bounds and separations for constant depth multilinear circuits. *Comput. Complex.*, 18(2):171–207, 2009. doi:10.1007/s00037-009-0270-8.

31 Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41:333–338, 1987.

32 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. *Github Survey*, 2015. URL: `https://github.com/dasarpmar/lowerbounds-survey`.

33 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Found. Trends Theor. Comput. Sci.*, 5(3-4):207–388, 2010. `doi:10.1561/0400000039`.

34 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In Alfred V. Aho, editor, *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, pages 77–82. ACM, 1987. `doi:10.1145/28395.28404`.

35 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Inf. Comput.*, 240:2–11, 2015. `doi:10.1016/j.ic.2014.09.004`.

36 Sébastien Tavenas, Nutan Limaye, and Srikanth Srinivasan. Set-multilinear and non-commutative formula lower bounds for iterated matrix multiplication. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20–24, 2022*, pages 416–425. ACM, 2022. `doi:10.1145/3519935.3520044`.

37 Leslie G. Valiant, Sven Skyum, Stuart J. Berkowitz, and Charles Rackoff. Fast parallel computation of polynomials using few processors. *SIAM J. Comput.*, 12:641–644, 1983.

38 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *26th Annual Symposium on Foundations of Computer Science, Portland, Oregon, USA, 21-23 October 1985*, pages 1–10. IEEE Computer Society, 1985. `doi:10.1109/SFCS.1985.49`.

A Proof Sketch of Lemma 22

In this section, we describe the proof of Lemma 22. As mentioned in Section 4, the proof structure is very similar to that of Lemma 20. The setup is similar as well, but we describe it here again for the convenience of the reader.

Again, we identify the set of variables $X = (X_1, \ldots, X_d)$ with the d-cycle $\{1, 2, \ldots, d\}$, where addition is modulo d. Let S be a partition of the cycle to K parts, namely, $S = (S_1, \ldots, S_K)$. We also think of $[K]$ as a set of colors, and of S as a (now "full") coloring of the cycle.

For a pairing P, define the number of k-violations by

$$V_k(P) = \{P_t \in P : |P_t \cap S_k| = 1\}$$

in words, the set of pairs in which one color is k and the other color is different. Fix $\varepsilon = 1/1000$ and denote

$$G(P) = \{k \in [K] : |V_k(P)| \geq d^{\varepsilon}\}$$

We do not include S as a subscript in these two notations since S will be known from the context (and will be fixed throughout most of the discussion). We begin by stating the analogue to Lemma 23, which shows that for every fixed K-coloring of the cycle, a random pairing has, with high probability, many colors with many violations.

▶ **Lemma 30.** *There is a constant $C > 0$ such that for all integers K in the range $[C, d^{1/1000}]$ the following holds: Let $S = (S_1, \ldots, S_K)$ be a partition of the d-cycle and suppose that $|S_k| \geq d^{7/8}$ for all $k \in [K]$. Then,*

$$\mathbb{P}[G(P) \leq K/1000] \leq d^{-K/500},$$

where $P \sim \mathcal{D}_P$.

Let us prove Lemma 22 given this lemma.

Proof of Lemma 22 given Lemma 30. We first need the following structural result, whose proof can be extrapolated from [33] , where it is shown for multilinear formulas.

▶ **Lemma 31** (Product Lemma; see [8, 33]). *Assume that F is a formula with at most s leaves, and is set-multilinear with respect to the set partition $(X_1, \ldots, X_d)$. Then, we can write*

$$F = \sum_{i=1}^{s} \prod_{j=1}^{\ell} F_{i,j}$$

where $\ell \geq \log d/100$ and for each $i \in [s]$, the product $F_i = \prod_{j=1}^{\ell} F_{i,j}$ is also set-multilinear. Furthermore, the degree of each $F_{i,j}$ is at least $d^{7/8}$.

Continuing with the proof, let F be a formula as in the statement of Lemma 22. We start by writing $F = \sum_{i=1}^{s} F_i$ in the form given by Lemma 31, so that each $F_i = \prod_{j=1}^{\ell} F_{i,j}$. As each F_i is set-multilinear, $(S_{i,1}, \ldots, S_{i,\ell})$ form a partition of $[d]$ where each $F_{i,j}$ is set-multilinear with respect to $(X_p)_{p \in S_{i,j}}$. Let $w^{i,1}, \ldots, w^{i,\ell}$ be the corresponding decomposition, whose respective sums are denoted simply by $w_{S_{i,1}}, \ldots, w_{S_{i,\ell}}$.

From the properties of relrk_w (Claim 7), we have

$$\mathrm{relrk}_w(F_i) = \prod_{j=1}^{\ell} \mathrm{relrk}_{w^{i,j}}(F_{i,j}) \leq \prod_{j=1}^{\ell} 2^{-\frac{1}{2}|w_{S_j}|} = 2^{-\frac{1}{2}\sum_{j=1}^{\ell} |w_{S_j}|},$$

from which we observe that the task of upper bounding $\mathrm{relrk}_w(F)$ can be reduced to the task of lower bounding the sum $\sum_{j=1}^{\ell} |w_{S_j}|$, which is established in the following claim. For the sake of convenience, the choice of the alphabet for w below is scaled down to $\{-1, 1\}$.

▷ Claim 32. For large enough d, suppose $(S_1, \ldots, S_K)$ is a partition of $[d]$ such that each $|S_j| \geq d^{7/8}$. Then, we have

$$\mathbb{P}_{w \sim \mathcal{D}} \left[\sum_{j=1}^{K} |w_{S_j}| < \frac{\log d}{2000} \right] \leq d^{-\frac{\log d}{10^7}}.$$

Here, $\mathcal{D}$ refers to the original distribution i.e., an arc-partition over the d-cycle.

Proof. The proof is going to be similar to that of Claim 25. Applying Lemma 30 to the tuple $(S_1, \ldots, S_K)$, we obtain that

$$\mathbb{P}[G(P) \leq K/1000] \leq d^{-K/500}.$$

The idea is to condition on the high probability event that $G(P) > K/1000$. Fix a pairing P with this property. Consider an ordering σ of the colors in $G(P)$. A color k is said to be *bright* with respect to an ordering if there are at least $d^{\varepsilon}/2$ nodes x of color k such that either the partner of x is uncolored or its partner is colored using a color that appears *after* k in the ordering σ. Call an ordering σ of the nodes in $G(P)$ *good* if there are at least $|G(P)|/2$ bright colors with respect to σ. The observation is that for any ordering σ of the colors, either σ itself is good, or its reverse is good. We conclude that given any pairing P, there exists a good ordering of $G(P)$. Fix any such good ordering and let $H(P)$ be the collection of bright colors with respect to this ordering.

Next, notice that if the sum $\sum_{j=1}^{K} |w_{S_j}|$ is at most $\frac{\log d}{2000}$, then so is the sum $\sum_{k \in H(P)} |w_{S_k}|$. Let $K' = |H(P)|$ (which is at least $K/2000$ if $G(P) > K/1000$). View the sampling of Π from P as happening in a specific order, according to the order of $k_1, k_2, \ldots, k_{K'}$: First define Π on pairs with at least one point with color k_1, then define Π on remaining pairs with at least one point with color k_2, and so forth. When finished with $k_1, \ldots, k_{K'}$, continue to define Π on all other pairs.

Conditioned on the event that $G(P) > K/1000$, this implies that $|w_{S_j}| \leq 1$ for each $j \in H(P)$. For every $j \in H(P)$, define E_j to be the event that $|w_{S_{k_j}}| \leq 1$. By choice, conditioned on $E_1, \ldots, E_{j-1}$, there are at least $d^{\varepsilon}/2$ pairs P_t so that $|P_t \cap S_{k_j}| = 1$ that are not yet assigned a "positive" or "negative" sign. For every such P_t, the element in $P_t \cap S_{k_j}$ is assigned a positive sign with probability $1/2$, and is independent of any other $P_{t'}$. The probability that a binomial random variable B over a universe of size $U \geq d^{\varepsilon}/2$ and marginals $1/2$ obtains any specific value is at most $O(U^{-1/2}) = O(d^{-\varepsilon/2})$. Hence, for all $j \in H(P)$, by the union bound,

$$\mathbb{P}[E_j | E_1, \ldots, E_{j-1}, P] \leq \mathbb{P}_B[U/2 - 1 \leq B \leq U/2 + 1] \leq O(3 \cdot d^{-\varepsilon/2}) \leq d^{-\varepsilon/4}.$$

Therefore,

$$\mathbb{P}[|w_{S_{k_j}}| \leq 1 \text{ for all } j \in H(P)] \leq \mathbb{E}[d^{-\varepsilon |H(P)|/4} | G(P) > K/1000] + d^{-K/500} \leq d^{-K/10^7}.$$

Finally, we note that

$$\mathbb{P}_{w \sim \mathcal{D}}\left[\sum_{j=1}^{K} |w_{S_j}| < \frac{\log d}{2000}\right] \leq \mathbb{P}[|w_{S_{k_j}}| \leq 1 \text{ for all } j \in H(P)]. \qquad \lhd$$

The claim above and the preceding calculation immediately implies that for every sub-formula F_i of size s_i,

$$\mathrm{relrk}_w(F_i) \leq s_i \cdot 2^{-\frac{k \log d}{2000}}$$

with probability at least $1 - d^{-\frac{\log d}{10^7}} \geq 1 - s_i \cdot d^{-\frac{\log d}{10^7}}$.

Next, by a union bound over $i \in [s]$ and the sub-additivity property of relrk_w, it follows that

$$\mathrm{relrk}_w(F) \leq s \cdot 2^{-\frac{k \log d}{2000}}$$

with probability at least $1 - s \cdot d^{-\frac{\log d}{10^7}}$, which concludes the proof of the lemma. $\blacktriangleleft$

We shall omit the proof of Lemma 30 here as it is, in fact, a significantly easier adaptation of Lemma 4.1 from [8] than the proof of Lemma 23 – this is because we no longer need to conduct the tighter analysis that was necessary for the low-depth case.

Matrix Multiplication and Number on the Forehead Communication

Josh Alman
Columbia University, New York, NY, USA

Jarosław Błasiok
Columbia University, New York, NY, USA

Abstract

Three-player Number On the Forehead communication may be thought of as a three-player Number In the Hand promise model, in which each player is given the inputs that are supposedly on the other two players' heads, and promised that they are consistent with the inputs of the other players. The set of all allowed inputs under this promise may be thought of as an order-3 tensor. We surprisingly observe that this tensor is exactly the matrix multiplication tensor, which is widely studied in the design of fast matrix multiplication algorithms.

Using this connection, we prove a number of results about both Number On the Forehead communication and matrix multiplication, each by using known results or techniques about the other. For example, we show how the Laser method, a key technique used to design the best matrix multiplication algorithms, can also be used to design communication protocols for a variety of problems. We also show how known lower bounds for Number On the Forehead communication can be used to bound properties of the matrix multiplication tensor such as its zeroing out subrank. Finally, we substantially generalize known methods based on slice-rank for studying communication, and show how they directly relate to the matrix multiplication exponent ω.

2012 ACM Subject Classification Theory of computation → Communication complexity

Keywords and phrases Number on the forehead, communication complexity, matrix multiplication

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.16

Related Version *Full Version*: `https://arxiv.org/abs/2302.11476`

Funding *Josh Alman*: Supported in part by NSF Grant CCF-2238221 and a grant from the Simons Foundation (Grant Number 825870 JA).
Jarosław Błasiok: Supported by Junior Fellowship from the Simons Society of Fellows.

Acknowledgements The authors thank Madhu Sudan and Toniann Pitassi for helpful discussions.

1 Introduction

Number on the forehead (NOF) communication complexity was introduced by Chandra, Furst and Lipton [12] as a variant of Yao's model of communication complexity. Here, we consider k players, each of them having one of k inputs, but instead of providing the players with their inputs, we imagine them having their inputs written on their foreheads – in particular each player can observe all inputs *except* for their own. Similar to the standard communication complexity, the players wish to exchange as little communication as possible, in order to jointly compute some function of their inputs.

As it turns out this innocuous alteration has deep mathematical consequences. For example, being able to prove lower bounds for specific NOF problems with $k = \Theta(\log^c n)$ players would imply explicit circuit complexity lower bounds [24, 7].

Moreover, even if we consider only three players, the theory of NOF communication complexity is simultaneously extremely deep, and frustratingly lacking. Its depth comes in part from a number of fascinating and perhaps surprising connections between natural

questions in NOF communication complexity and well-studied problems in Ramsey Theory, extremal combinatorics and additive combinatorics. For example, already in the seminal paper [12], a surprisingly efficient deterministic protocol was provided for the NOF problem of checking whether $x + y + z = 0$ in $\mathbb{Z}_N$, where $x, y, z \in \mathbb{Z}_N$ are the inputs on the players' foreheads (we will call this problem $\textsc{Eval}_{\mathbb{Z}_N}$ from now on). The protocol uses just $\mathcal{O}(\sqrt{\log N})$ communication, and makes use of Behrand's construction of a relatively dense set without three-terms arithmetic progression [6]. On the other hand, in the same paper they showed an $\omega(1)$ lower bound for the $\textsc{Eval}_{\mathbb{Z}_N}$ problem using the Hales–Jewett theorem – a deep theorem in Ramsey theory. Since then, connections with the Ruzsa–Szemerédi problem, Corner-Free Sets and other objects from extremal combinatorics have been discovered, frequently leading to state-of-the-art constructions [26, 4, 3, 27, 14].

At the same time, perhaps because of these connections, many seemingly simple questions about NOF communication complexity remain unresolved, despite the fact that their number-in-hand variants are very easy to answer. For example, completely resolving the communication complexity of $\textsc{Eval}_{\mathbb{Z}_N}$ requires a significantly better understanding of the correct quantitative dependency in the Roth's theorem (that large density subsets of $[N]$ contain a 3-terms arithmetic progression), and seems to be far out of reach of current techniques.

Similarly, it is still open how to find an explicit 3-party NOF problem which can be solved with $\mathcal{O}(1)$ randomized communication complexity (assuming shared randomness), but which requires $\Omega(n)$ deterministic communication, even though this is a relatively simple property of Equality for two-party communication, and even though, with a non-constructive argument, it is possible to show that such NOF problems exist [21]. A natural candidate for such a problem is checking whether $x + y + z = 0$ in $\mathbb{F}_3^n$ (or, more generally, if $xyz = 1$ in a large power G^n of some fixed group G, not necessarily abelian – we will call those problems $\textsc{Eval}_{G^n}$), but proving a NOF communication complexity lower bound has been so far elusive.

Problems like $\textsc{Eval}_{G^n}$, in which for any pair of inputs to two of the players, there is exactly one input to the remaining player for which they should accept, will be called *permutation problems*. These problems always have $\mathcal{O}(1)$ randomized communication complexity, so they are natural candidates to try to prove $\Omega(n)$ deterministic communication lower bounds. Indeed, a number of recent works have studied permutation problems toward this goal [5, 14].

The construction [12] of an efficient protocol for $\textsc{Eval}_{\mathbb{Z}_N}$ can be interpreted as a statement that existence of a large set $S \subset G$ without three terms arithmetic progressions in an abelian group G can be used to provide an efficient protocol for $\textsc{Eval}_G$; by contrapositive, a lower bound for the communication complexity of $\textsc{Eval}_G$ would imply an upper bound for the size of any such set S. This latter statement for the group $\mathbb{F}_3^n$ used to be known as a *cap-set conjecture* for few decades, and has famously been positively resolved in 2016 [22, 20, 32] – the largest set $S \subset \mathbb{F}_3^n$ without three terms arithmetic progressions has size $\mathcal{O}(c^n)$ for some $c < 3$. This leads to a natural question: can the techniques used to prove the cap-set theorem be generalized to prove a linear lower bound for the communication complexity of $\textsc{Eval}_{\mathbb{F}_3^n}$?

Christandl et al. [14] recently identified a barrier against a direct applications of such techniques – they used a method called *combinatorial degenration* to prove that a *subrank* of a specific tensor associated with the communication problem $\textsc{Eval}_{G^n}$ for an abelian group G is near-maximal – whereas an upper bound on this subrank would imply a desired lower-bound for communication complexity.

1.1 Our contributions

In this paper, we study three-player communication through the lens of *promise number-in-hand problems*. For an alphabet Σ, a promise problem (I, P) consists of two subsets $I \subset P \subset \Sigma \times \Sigma \times \Sigma$. In this problem, the three players are given as input $a, b, c \in \Sigma$,

respectively, with the promise that $(a, b, c) \in P$, and their goal is to determine whether $(a, b, c) \in I$. For instance, usual number-in-hand communication corresponds to this model where $P = \Sigma \times \Sigma \times \Sigma$.

NOF problems can also be viewed as promise number-in-hand problems. Each player is given as an input a pair $(a_i, b_i) \in \Sigma^2$ for $i \in \{1, 2, 3\}$, and their inputs are guaranteed to satisfy that $a_2 = b_1, a_3 = b_2$ and $a_1 = b_3$. (For instance, one thinks of a_2 and b_1 as the input on the head of player 3, so we are promised that players 1 and 2 see the same value there.) Write $P_{\text{NOF}} \subset \Sigma^2 \times \Sigma^2 \times \Sigma^2$ to denote the inputs that satisfy that promise.

A key idea we pursue is to interpret I and P, not just as subsets of inputs, but also as order-3 (3-dimensional) $\{0, 1\}$-valued tensors. For example, letting $N = |\Sigma|$, we can represent P_{NOF} as a $\{0, 1\}$-valued tensor $P_{\text{NOF}} \in \mathbb{F}^{N^2 \times N^2 \times N^2}$, with the tensor entry being one whenever the corresponding tuple is a legal input.

As it turns out, P_{NOF} is *exactly* the matrix multiplication tensor, the same tensor which is extensively studied in the theory of fast matrix multiplication algorithms! We leverage this extremely surprising, if only formal connection, to transfer ideas which have been developed in studying fast matrix multiplication to give new insights into NOF communication complexity, and conversely, to use results proved originally about NOF communication complexity and obtain new alternative proofs of consequential properties of the matrix multiplication tensor.

We use this connection to prove the following results.

Communication Protocols from the Laser Method

We use the Laser method, the technique introduced by Strassen and used to design the best-known matrix multiplication algorithms [31, 19], as a tool to design NOF communication protocols. For any three NOF problems I, J, K, let $I \otimes J \otimes K$ ("their product") denote the NOF problem of solving the three simultaneously, i.e., given one input to each, determining whether all three are accepting inputs. Using the Laser method, we show that for any three permutation NOF problems I, J, K with n-bit inputs, the product $I \otimes J \otimes K$ (which has $3n$-bit inputs) can be solved using only n bits of communication deterministically.

For example, for any permutation NOF problem I with n-bit inputs, one can consider the problem of, given N simultaneous inputs to I, determining if they are all accepting inputs. A consequence of our protocol is that this can be solved with $\frac{1}{3}nN + \mathcal{O}(n)$ bits of communication deterministically. This implies constructions of surprisingly large corner-free sets for powers of arbitrary abelian groups G.

By contrast, we show using a counting argument that a random NOF permutation problem I with n-bit inputs has requires deterministic communication complexity $\geq n/3 - \mathcal{O}(1)$. It was not previously clear to what extent this is tight; our protocol shows that this bound can at least be achieved for any permutation problem which is itself the product of three permutation problems.

Our protocol should also be contrasted with the proof by probabilistic method in [14], which showed that every NOF permutation problem with n-bit inputs has a protocol with communication complexity $n/2 + \mathcal{O}(1)$, improving in turn upon the trivial n upper bound.

The Zeroing Out Subrank of Matrix Multiplication

A quantity which appears frequently in the theory of matrix multiplication algorithms is the *zeroing out subrank* of matrix multiplication, denoted $Q_{zo}(\langle n, n, n\rangle)$. This is the size of the largest identity tensor (3-dimensional analogue of the identity matrix) which can be achieved as a zeroing out of the tensor $\langle n, n, n\rangle$ for multiplying two $n \times n$ matrices. Strassen [31]

famously proved that $Q_{zo}(\langle n,n,n\rangle) \geq n^{2-o(1)}$, which is roughly as large as possible (since $\langle n,n,n\rangle$ is an $n^2 \times n^2 \times n^2$ tensor), and used it as part of the Laser method to design a faster algorithm for matrix multiplication[1]. Recent *barrier* results have also used this bound to rule out certain approaches to designing faster matrix multiplication algorithms [2, 1, 15].

In the bound $Q_{zo}(\langle n,n,n\rangle) \geq n^{2-o(1)}$, it is natural to ask what is hidden by the $o(1)$. For instance, it was recently shown that the "border subrank" of $\langle n,n,n\rangle$ is $\lceil \frac{3}{4}n^2 \rceil$ [25]; is it possible that we also have $Q_{zo}(\langle n,n,n\rangle) \geq \Omega(n^2)$? This could improve low-order terms in some applications.

In fact, we prove that this is not possible, and that $Q_{zo}(\langle n,n,n\rangle) < n^2/\omega(1)$. To prove this, we build off of the work of [26], who showed a NOF communication lower bound, that any NOF permutation problem requires $\omega(1)$ deterministic communication. Using our connection between NOF communication and matrix multiplication, we are able to modify their proof to get our upper bound on $Q_{zo}(\langle n,n,n\rangle)$.

Interestingly, the prior work [26] proved their communication lower bound using a connection with the Ruzsa–Szemerédi problem from extremal combinatorics. Not only does our new proof show a connection between $Q_{zo}(\langle n,n,n\rangle)$ and the Ruzsa–Szemerédi problem, but we are in fact able to show that the two are *equivalent*! More precisely, it is impossible to determine the precise asymptotics of $Q_{zo}(\langle n,n,n\rangle)$ without also resolving the Ruzsa–Szemerédi problem. This is interesting in contrast with the "border subrank" which has been precisely determined [25].

Finally, using the connection with NOF communication, we also provide an alternative proof of Strassen's lower bound $Q_{zo}(\langle n,n,n\rangle) \geq n^{2-o(1)}$, deducing it as a corollary of the existence of the efficient deterministic NOF protocol for $\textsc{Eval}_{\mathbb{Z}_N}$.

Slice-Rank Methods

Since their introduction [12], NOF problems I have been associated with a corresponding 3-hypergraph H whose chromatic number exactly characterizes the deterministic communication complexity of I. One recent powerful approach for bounding the chromatic number of such a hypergraph is the "slice rank" method, which was introduced to resolve the cap-set conjecture [20, 22]. Roughly speaking, if one can show that the slice rank of the adjacency tensor of H is not too large, then this implies the chromatic number of H is not too small, and hence a communication lower bound.

Since then, a few successful techniques have been introduced to bound the slice ranks of many different tensors [32, 10]. Recent work [13] asked whether these techniques could give linear deterministic NOF communication lower bounds. However, they showed that, unfortunately, it is impossible to use such techniques to prove communication lower bounds for any Eval group problem. They proved this by defining the corresponding adjacency tensor and calculating that it has maximal "asymptotic subrank" and hence large slice rank.

We generalize their result and show that slice-rank methods cannot be used to prove a linear deterministic communication lower bound for *any* NOF permutation problem (not just an Eval group problem). To show this, we observe that for any NOF permutation problem, the corresponding adjacency tensor is actually (a permutation of) the matrix multiplication

[1] See, e.g., [9, Section 8]. Bounds used in algorithms are sometimes described as bounds on a "monomial degeneration subrank" or "combinatorial degeneration subrank", but these are in turn ultimately used to bound the zeroing out subrank via [8]. Note that bounds on more general notions like "border subrank" or "subrank" would not suffice in these algorithms, since the subrank decomposition is applied only to the "outer structure" of a tensor in the Laser method [31].

tensor P_{NOF}. Since the matrix multiplication tensor is known to have maximal asymptotic subrank [31], it follows that the adjacency tensor never has small enough slice rank. In particular, this implies that in hindsight, the tensor whose asymptotic subrank was computed in the prior work [13] is exactly the matrix multiplication tensor.

By contrast, we use slice-rank methods to prove linear lower bounds for other promise problems. The key is that this method works whenever the *promise tensor* has low slice-rank. For example, consider the promise problem $\text{EQ}_{\mathbb{F}_3^N}$, where parties receive inputs $x, y, z \in \mathbb{F}_3^N$ with a promise that $x + y + z = 0$, and they wish to decide whether $x = y$. We prove that this problem requires $\Omega(N)$ communication. This may seem like it should be straightforward to prove ("how could the third player"s input $z = -(x + y)$ possibly help the first two players to test for equality?"), but we prove that the problem is actually *equivalent* to the cap-set problem (which was a long-open conjecture).

General Rank Methods and the Asymptotic Spectrum of Tensors

Slice-rank is one way to measure the "complexity" of a tensor, but there are many others, including other notions of tensor rank, and more generally a spectrum of different measures which is well-studied in algebraic complexity and tensor combinatorics called the "asymptotic spectrum of tensors" [31, 34]. Let r be any of these measures (the reader may want to focus on the case when r is the tensor rank). We generalize the slice-rank method and prove that for any promise problem (I, P), its deterministic communication complexity is at least $\log_2(r(I)/r(P))$. Hence, finding a measure r which is larger for the problem I than the promise P leads to a communication lower bound. (This is why, for slice rank, the fact that the promise P_{NOF} has maximal slice rank means that a lower bound cannot be proved in this way.)

This already has intriguing consequences when r is tensor rank R.

For number-in-hand communication, we know that the promise tensor P_{NIH} is the all-1s tensor which has rank $R(P_{NIH}) = 1$. Hence, for any number-in-hand problem I, we get a communication lower bound of $\log_2(R(I))$. This is analogous to the standard upper bound in two-party deterministic communication compleixty by the log of the rank of the communication matrix. Moreover, any $2^n \times 2^n \times 2^n$ tensor I (corresponding to an n-bit input problem) which is not "degenerate" in some way has rank at least 2^n, which explains why nearly all number-in-hand communication problems require linear communication.

For NOF communication for n-bit inputs, as we've discussed, the promise tensor P_{NOF} is the matrix multiplication tensor. Determining the rank of matrix multiplication is the *central question* when designing matrix multiplication algorithms; the matrix multiplication exponent ω is defined exactly such that the rank of P_{NOF} is $R(P_{NOF}) = (2^n)^{\omega+o(1)}$. Hence, for any problem I with rank $R(I) \geq (2^n)^c$, we get a communication lower bound of $(c - \omega - o(1)) \cdot n$. The fact that ω appears negated in this lower bound is intriguing: this means that designing faster matrix multiplication algorithms, and hence lowering the known upper bound on ω, leads to an improved communication lower bound for such problems I.

Matrix Multiplication as a Communication Problem

Finally, we consider how other promise problems can be used to shed further light on NOF communication lower bounds. Let I be any n-bit NOF problem, let $N = 2^n$, and let $T_{(\mathbb{Z}/N\mathbb{Z})^2}$ denote the structure tensor of the group $(\mathbb{Z}/N\mathbb{Z})^2$, i.e., the tensor

$$T_{(\mathbb{Z}/N\mathbb{Z})^2} = \sum_{a,b,c,d \in [N]} x_{(a,b)} y_{(c,d)} z_{(a+c \pmod N, b+d \pmod N)}.$$

We observe that $P_{NOF} \subset T_{(\mathbb{Z}/N\mathbb{Z})^2}$ (after appropriately permuting variables), and so it is well-defined to consider three different promise problems using these three tensors: (I, P_{NOF}), $(P_{NOF}, T_{(\mathbb{Z}/N\mathbb{Z})^2})$, and $(I, T_{(\mathbb{Z}/N\mathbb{Z})^2})$. Moreover, a simple triangle inequality-style argument shows that their communication complexities are related via:

$$\mathcal{CC}(I, P_{NOF}) \geq \mathcal{CC}(I, T_{(\mathbb{Z}/N\mathbb{Z})^2}) - \mathcal{CC}(P_{NOF}, T_{(\mathbb{Z}/N\mathbb{Z})^2}).$$

On the left-hand side is exactly the deterministic NOF complexity of the problem I that we would like to prove a lower bound for. On the right-hand side, $(I, T_{(\mathbb{Z}/N\mathbb{Z})^2})$ is a similar problem to the original NOF problem of I, but with a weaker promise, and so it should be easier to prove communication lower bounds for. What about $\mathcal{CC}(P_{NOF}, T_{(\mathbb{Z}/N\mathbb{Z})^2})$? (Recall that in this problem, we're given as input a term from the tensor $T_{(\mathbb{Z}/N\mathbb{Z})^2}$, and our goal is to determine whether or not it is a valid NOF input from P_{NOF}.)

For this problem, we give a protocol with communication $\leq n$ (the trivial bound would be $2n$). Moreover, we prove a communication lower bound of $\geq (\omega - 2)n$ (this follows from the rank-based approach discussed above). So, further improvements to the communication protocol actually *require* fast matrix multiplication (since they would imply an upper bound on ω via this inequality). If it were the case that $\omega = 2$ (which is popularly conjectured), then it is plausible that this problem actually has sublinear communication complexity. In that case, in order to prove a linear lower bound for the NOF problem (I, P_{NOF}), it would suffice to prove a linear lower bound for the seemingly-harder problem $(I, T_{(\mathbb{Z}/N\mathbb{Z})^2})$. In summary, if we could design a faster matrix multiplication algorithm achieving $\omega = 2$, then this may be a promising approach to proving NOF communication lower bounds.

2 Tensor Preliminaries

In this paper we will relate communication problems to related tensors and their properties. We will focus on order-3 tensors (also known as 3-dimensional tensors). For a field $\mathbb{F}$ and three positive integers A, B, C, we define $\mathbb{F}^{A \times B \times C}$ to be the set of tensors

$$T = \sum_{a \in A} \sum_{b \in B} \sum_{c \in C} T_{a,b,c} x_a y_b z_c$$

for coefficients $T_{a,b,c} \in \mathbb{F}$ (where x_a, y_b, z_c are formal variables). When the specific sets are not important, we will often write $\mathbb{F}^{|A| \times |B| \times |C|}$ instead of $\mathbb{F}^{A \times B \times C}$.

T is $\{0,1\}$-valued if, for all a, b, c, we have $T_{a,b,c} \in \{0,1\}$. Such tensors are in bijection with subsets of $A \times B \times C$. All tensors considered in this paper are assumed to be $\{0,1\}$-valued unless stated otherwise.

The field $\mathbb{F}$ that one works over will be primarily important to us for the following definition. The (not necessarily $\{0,1\}$-valued) tensor T has rank 1 if it can be written as

$$T = \left(\sum_{a \in A} \alpha_a x_a \right) \left(\sum_{b \in B} \beta_b y_b \right) \left(\sum_{c \in C} \gamma_c z_c \right)$$

for coefficients $\alpha_a, \beta_b, \gamma_c \in \mathbb{F}$. More generally, the rank of the (not necessarily $\{0,1\}$-valued) tensor T, denoted $R(T)$, is the minimum number of tensors of rank 1 which sum to T.

Rank is critical to the definition of the matrix multiplication exponent ω. For $n, m, p \in \mathbb{N}$, we write $\langle n, m, p \rangle \in \mathbb{F}^{n^2 \times n^2 \times n^2}$ to denote the matrix multiplication tensor

$$\langle n, m, p \rangle = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{p} x_{i,j} y_{j,k} z_{k,i}.$$

The exponent ω is defined as

$$\omega := \inf\{t \mid R(\langle n, n, n\rangle) \leq n^t \text{ for some } n \in \mathbb{N}\}.$$

Indeed, it is known that for any $\varepsilon > 0$, matrix multiplication of $n \times n$ matrices can be performed with an arithmetic circuit of size $\mathcal{O}(n^{\omega+\varepsilon})$, and conversely, that any arithmetic circuit for matrix multiplication can be converted into a tensor rank upper bound which yields the same operation count in this way [30].

We say T is an identity tensor if, for all $a \in A$, there is at most one pair $(b, c) \in B \times C$ such that $T_{a,b,c} \neq 0$, and similarly for all $b \in B$ and for all $c \in C$. Often, when $A = B = C$, one specifically thinks of the identity tensor of size k as a tensor $\sum_{a \in S} x_a y_a z_a$ for some subsets $S \subset A$ of size $|S| = k$. To be clear that this is sufficient but not necessary (as one could in general permute the indices), we will refer to permutations of identity tensors.

If $T, T' \in \mathbb{F}^{A \times B \times C}$, we say T' is a zeroing out of T if there are subsets $A' \subset A, B' \subset B, C' \subset C$ such that $T'_{a,b,c}$ is equal to $T_{a,b,c}$ whenever $a \in A', b \in B', c \in C'$, and $T'_{a,b,c} = 0$ otherwise. In other words, we are setting coefficients outside of A', B', C' to zero. We will sometimes write $T' = T|_{A',B',C'}$.

The zeroing out subrank of T, denoted $Q_{zo}(T)$, is the maximum k such that T has a zeroing out to a permutation of an identity tensor of size k. This more combinatorial variant on the notion of "subrank" (which uses "restrictions" rather than zeroing outs) will be useful a number of times.

3 Promise problems and colored tensors

▶ **Definition 1.** *A promise problem over alphabet Σ is a pair of subsets $I \subset P \subset \Sigma \times \Sigma \times \Sigma$, where P is a set of allowed inputs (the* promise*), and I is the subset of accepting inputs (the* problem*).*

To study those objects, we will introduce a notion of *colored tensors*. A colored tensor is a pair (I, T_P), where $T_P \in \mathbb{F}^{\Sigma \times \Sigma \times \Sigma}$ is a tensor and $I \subset \mathrm{supp}(T_P)$ is a subset of its non-zero terms. (We imagine those terms to have a special color, and in what follows we will call them *green* terms). 3-party promise number in hand problems over the alphabet Σ are therefore in direct correspondence with $\{0, 1\}$-valued colored tensors over $\mathbb{F}^{\Sigma \times \Sigma \times \Sigma}$, where the non-zero terms of the tensor T corresponds to the set of allowed inputs, and the subset I of green terms corresponds to accepting inputs.

▶ **Definition 2.** *For any field $\mathbb{F}$ we can define the* communication tensor *of a promise problem (I, P) as a colored tensor (I, T_P), with $T_P \in \mathbb{F}^{\Sigma \times \Sigma \times \Sigma}$ given by*

$$T_P := \sum_{(i,j,k) \in P} x_i y_j z_k.$$

We will say that T_P is the promise tensor *of the problem (I, P).*

We will sometimes abuse notation, and use the same symbol to denote a $\{0, 1\}$-valued tensor $P \in \mathbb{F}^{A \times B \times C}$ and its support $\mathrm{supp}(P) \subset A \times B \times C$ whenever it is clear from context. All tensors we discuss in this paper are $\{0, 1\}$-valued and for the vast majority of the discussion the underlying field is irrelevant.

Communication Complexity Model

We use $\mathcal{CC}(I, P)$ to denote the deterministic communication complexity of the promise problem (I, P). To make this concrete, we consider a shared blackboard model: here a protocol is given by a triple of functions, one for each player, encoding what each player should append to the blackboard in each round of communication, given the current state of the blackboard, and their private input. Players write their communication on the blackboard in cyclic order; in each stage each player can either append something to the blackboard, accept the current instance, or reject the current instance. A specific instance is accepted if *all players* accept it (based on their own input, and the final state of the blackboard). A protocol solves a promise problem (I, P) if it accepts all the inputs from I and rejects all other inputs from $P \setminus I$. (Since P is the promise, the protocol need not have any particular behavior on inputs outside of P.)

Asymptotic communication complexity

For a promise problem (I, P) we use $(I, P)^{\otimes n}$ to denote a promise problem in which parties are given n valid instances of the original promise problem (I, P), and wish to determine whether *all* of those instances simultaneously are accepting.

More formally, $(I, P)^{\otimes n}$ is the promise problem with the promise $P^{\otimes n} \subset (\Sigma^n)\times(\Sigma^n)\times(\Sigma^n)$, i.e., for $A, B, C \in \Sigma^n$, the triple of sequences (A, B, C) is in $P^{\otimes n}$ if and only if we have $(A_i, B_i, C_i) \in P$ for all $i \in [n]$. The set of accepting instances $I^{\otimes n}$ is defined analogously. This operation corresponds to the Kronecker product of the associated communication tensors.

▶ **Definition 3.** *For tensors $I^{(1)} \subset T^{(1)} \subset \mathbb{F}^{A_1\times B_1\times C_1}$ and $I^{(2)} \subset T^{(2)} \subset \mathbb{F}^{A_2\times B_2\times C_2}$ We define the Kronecker product of colored tensors $(I^{(1)}, T^{(1)}) \otimes (I^{(2)}, T^{(2)})$ as a colored tensor (I, T) in $\mathbb{F}^{A_1A_2\times B_1B_2\times C_1C_1}$, where $T = T_1 \otimes T_2$ satisfies*

$$T_{(a_1,a_2),(b_1,b_2),(c_1,c_2)} = T^{(1)}_{a_1,b_1,c_1} \cdot T^{(2)}_{a_2,b_2,c_2},$$

and $I = \{(a_1, a_2), (b_1, b_2), (c_1, c_2) : (a_i, b_i, c_i) \in I_i$ for $i \in \{1, 2\}\}$.

We remark that all tensors considered in this work are order-3 tensors (also known as 3-dimensional tensors), and we will use symbol $\otimes$ exclusively to denote the Kronecker product of tensors (as opposed to tensor product).

We will define the asymptotic communication complexity of the problem P as

$$\underline{\mathcal{CC}}(I, P) := \lim_{n\to\infty} \frac{\mathcal{CC}((I, P)^{\otimes n})}{n}.$$

Permutation problems

We pay special attention to a particular class of communication problems, which we will call *injection problems* – as we will see later on, the deterministic communication complexity of injection problems have very simple combinatorial interpretation, and they all have protocols with constant randomized communication (under public randomness).

▶ **Definition 4** (Injection problem)**.** *Let $\pi_1, \pi_2, \pi_3 : \Sigma^3 \to \Sigma$ be projections onto first, second, and third coordinate respectively.*

We say that (I, P) (where $I \subset P \subset \Sigma^3$) is an injection problem*, if the tensor $T_I := \sum_{(i,j,k)\in I} x_i y_j z_k$ restricted to the coordinates appearing in I (i.e. $(T_I)|_{\pi_1(I),\pi_2(I),\pi_3(I)}$) is a permutation of an identity tensor.*

That is: for every valid input of the $i \in \Sigma$ of the first player, there is at most one pair j, k, such that $(i, j, k) \in I$, and analogously for the second and the third player.

▶ **Definition 5** (Permutation problem). *We say that (I, P) is a permutation problem if (I, P) is an injection problem and moreover $|I| = \Sigma$.*

Note that if (I, P) is an injection (resp. permutation) problem, then $(I, P)^{\otimes n}$ also is an injection (permutation) problem.

3.1 Number on the forehead problems as promise-number-in-hand problems

We can treat a NOF problem $I \subset \Sigma \times \Sigma \times \Sigma$ as a promise problem over the alphabet $\Gamma := \Sigma \times \Sigma$, where the promise $P_{\text{NOF}} \subset \Gamma^3$ is given by the following promise-tensor

$$P_{\text{NOF}} := \sum_{i,j,k \in \Sigma} x_{i,j} y_{j,k} z_{k,i}.$$

A key observation that we will explore in this paper is that P_{NOF} is exactly the matrix multiplication tensor $P = \langle N, N, N \rangle$ where $N = |\Sigma|$.

The set of accepting instances $I \subset \Gamma \times \Gamma \times \Gamma$ can be bijectively mapped into a subset $\tilde{I} \subset \langle N, N, N \rangle$ of non-zero terms of the promise tensor, by $\tilde{I} := \{((x, y), (y, z), (z, x)) : (x, y, z) \in I\}$.

Having interpreted number-on-the-forehead problems as number-in-hand problems with promise $\langle N, N, N \rangle$, we observe that our more general definition of the permutation problem agrees with the definition of the permutation problem in [26] for NOF problems – which are in turn are equivalent to Latin squares of size $N \times N$.

An important family of examples of NOF permutation problems are Eval problems induced by an abelian group G. Specifically, for any abelian group G, let Eval_G be the problem defined as follows. Players have $x, y, z \in G$ on their foreheads, and they want to decide whether $x + y + z = 0$. We can represent the problem by its set of accepting instances $I \subset G^3$ defined by $I = \{(x, y, z) : x + y + z = 0\}$.

The observation that all NOF permutation problems admit an efficient randomized protocol, generalizes to injection problems regardless of the promise, using a constant communication randomized protocol for two-players Equality problem.

▶ **Fact 6.** *Every injection problem has a $O(1)$ randomized protocol with shared randomness, regardless of the promise.*

Proof. First player, upon receiving his input x can find unique y_1, z_1, such that $(x, y_1, z_1) \in I$. He can then proceed by using the randomized two-player protocol for Equality to check whether $y_1 = y$, and $z_1 = z$, where y, z are inputs of the second and third player respectively. ◀

What can we say about the deterministic communication complexity of promise injection problems? It turns out that we can express it equivalently as a question about the smallest proper coloring of the communication tensor (I, P).

▶ **Definition 7.** *Consider a colored tensor $T = (I, P)$, We say that a subset of its green terms $S \subset I$ is an independent set if S is a permutation of an identity tensor and the restriction $T|_{\pi_1(S), \pi_2(S), \pi_3(S)}$ is equal to S. We use $\alpha(T)$ to denote the size of largest independent set in the colored tensor T.*

If T is an ordinary tensor, we use $\alpha(T)$ to denote the size of the largest independent set in the colored tensor $(\text{supp}(T), T)$ (i.e., we treat all its terms as green). This coincides with the usual notion of zero-out subrank of a tensor.

Note that if $T = (I, P)$ is an injection problem, $S \subset I$ is an independent set if and only if all non-zero terms in the restriction $T|_{\pi_1(S),\pi_2(S),\pi_3(S)}$ are in I – that is, if we denote by $S_i := \pi_i(S)$ then set S is independent if and only if

$$\mathrm{supp}(T) \cap (S_1 \times S_2 \times S_3) = S.$$

With a definition of an independent set, we can define a proper coloring and the chromatic number of a colored tensor in a natural way.

▶ **Definition 8.** *For a colored tensor $T := (I, P)$ we say that $\tau : I \to [k]$ is a proper coloring if for all colors $c \in [k]$, the set $\tau^{-1}(c)$ is independent.*

We denote by $\chi(T)$ the chromatic number *of tensor T – the smallest k, such that there is a proper coloring $\tau : I \to [k]$. Again, if T is an ordinary tensor, we use a shorthand $\chi(T) := \chi(\mathrm{supp}(T), T)$.*

The proof of the following characterization follows exactly the original proof in [12], just phrased in a slightly more general language of promise problems instead of specifically Number on the Forehead problems.

▶ **Theorem 9.** *The deterministic communication complexity of any promise injection problem (I, P) is equal to $\lceil \log \chi(I, P) \rceil$.*

Proof. Given a coloring $\tau : I \to [k]$, we will describe a communication protocol with complexity $\lceil \log k \rceil$. The first player, with an input x, looks at the only $y^{(1)}, z^{(1)}$, s.t. $(x, y^{(1)}, z^{(1)}) \in I$, and writes down the color $\tau(x, y^{(1)}, z^{(1)})$. Similarly, the second player, with an input y, compares the written color with the color of $(x^{(2)}, y, z^{(2)}) \in I$, and analogously the third player. Clearly, if (x, y, z) was an accepting instance, all three of the triples in the consideration would be the same ($x^{(2)} = x^{(3)} = x, y^{(1)} = y^{(3)} = y, z^{(1)} = z^{(2)} = z$), and therefore the colors under consideration would agree – the protocol will correctly accept such an instance.

On the other hand, if all $(x, y^{(1)}, z^{(1)}), (x^{(2)}, y, z^{(2)}), (x^{(3)}, y^{(3)}, z)$ have the same color c, let us take the independent set $S := \tau^{-1}(c) \subset I$, then $x \in \pi_1(S), y \in \pi_2(S), z \in \pi_3(S)$, and since $(x, y, z) \in P$ and S is an independent set, we must have $(x, y, z) \in I$ – i.e. the instance was accepting. This completes the proof that a coloring $\tau : I \to [k]$ implies existence of an efficient protocol.

The converse is similar: given a protocol for a problem (I, P) with communication at most k, we can use the transcript of the communication on each input $(x, y, z) \in I$ as a color for the term. This yields a mapping $\tau : I \to [2^k]$, and all we need to check is that it is a proper coloring. Indeed, let us chose an arbitrary color c, and take a set $S = \tau^{-1}(c)$. If there was $(x, y, z) \in P - I$, such that $(x, y, z) \in \pi_1(S) \times \pi_2(S) \times \pi_3(S)$, the communication transcript on the (allowed) input (x, y, z) would have been the same as on accepting inputs $(x, y^{(1)}, z^{(1)}), (x^{(2)}, y, z^{(2)}), (x^{(3)}, y^{(3)}, z)$, and therefore each player would have accepted it – contradicting correctness of the protocol, since $(x, y, z) \notin I$. ◀

3.2 Comparison with standard bounds for the NOF communication complexity

What we discussed so far is a slightly different perspective on the standard way of characterizing NOF communication complexity (already present in [12]). Classically, with NOF permutation problem $I \subset \Gamma \times \Gamma \times \Gamma$, one associated a directed 3-hypergraph H, and the logarithm of the chromatic number of this hypergraph is equal to the deterministic communication complexity of the associated problem.

We will show that in fact the adjacency tensor of the aforementioned hypergraph H is just a permutation of the promise tensor P_{NOF} (i.e., the matrix multiplication tensor), and the permutation is uniquely determined by the problem I – it corresponds to permuting variables y_i and z_i in a way such that I is exactly on the diagonal of the tensor.

This provides a generalization, and another perspective on the barrier against using a slice-rank or any other bounds on the subrank of the adjacency tensor of H as a way to show a lower bound for communication complexity of H.

Specifically, note that any independent set $S \subset V(H)$ induces a zeroing-out of the adjacency tensor of the hypergraph H to a large identity subtensor (given by restriction to indices in $S \times S \times S$ – note that in the construction of the hypergraph H and its adjacency tensor, we include self-loops (t,t,t) for each $t \in V(H)$). One could then hope to show a lower bound on the chromatic number of the hypergraph H by upper bounding the size of the largest independent set, which in turn can be upper bounded by subrank of adjacency sub-tensor. Finally, it is known that subrank of any tensor is always bounded by its slice-rank, and a few successful techniques for controlling slice-ranks of particular tensors of interest have been developed [32, 10].

This, apparently promising avenue for showing lower bounds for specific NOF permutation problems unfortunately has to fail. In [14] it was shown that similar techniques as those used by Strassen to prove lower bound on the asymptotic subrank of the matrix multiplication tensor, can be used to lower bound the asymptotic subrank of the adjacency tensor for any Eval group problem – it is maximally large. We observe that this is not a coincidence: in fact all adjacency tensors constructed this way are just permutations of the matrix-multiplication tensor, so the lower bound for asymptotic subrank of the matrix multiplication tensor directly applies – not only for problems arising as Eval_G, but more generally for all number-on-the-forehead permutation problems.

▶ **Definition 10** (Communication Hypergraph [12]). *For a permutation NOF problem $I \subset \Gamma \times \Gamma \times \Gamma$, we define its communication hypergraph $H(I)$ to be an order-3 hypergraph, with vertices $V(H(I)) = I$ – set of all accepting instances, and hyperedges constructed in the following way: for all $(x,y,z) \in \Gamma \times \Gamma \times \Gamma$, let $T_1 := (x,y,z') \in I$ be the only triple in I with agreeing with (x,y,z) in the first two coordinates, and similarly $T_2 := (x',y,z) \in I, T_3 := (x,y',z) \in I$.*

Then $E(H(I)) = \{(T_1,T_2,T_3) : (x,y,z) \in \Gamma^3\}$.

▶ **Fact 11.** *The adjacency tensor of the hypergraph $H(I)$ is a permutation of the matrix multiplication tensor $\langle|\Gamma|,|\Gamma|,|\Gamma|\rangle$.*

Moreover if we consider a set of green terms $\tilde{I} \subset \langle|\Gamma|,|\Gamma|,|\Gamma|\rangle$ given by $\tilde{I} = \{((x,y),(y,z),(z,x)) : (x,y,z) \in I\}$ in the colored communication tensor for the problem I, this set corresponds under the aforementioned permutation exactly to the set of diagonal terms $\{(t,t,t) : t \in I\}$ of the adjacency tensor of $H(I)$.

Proof. Consider a map $\pi_{12} : I \to \Gamma \times \Gamma$, given by $\pi_{12}(x,y,z) = (x,y)$, and similarly π_{23}, π_{13}. Note that those three bijections between I and $\Gamma \times \Gamma$. Applying those bijections to three axes of the adjacency tensor A of the hypergraph $H(I)$, we see that an image of any hyperedge (T_1,T_2,T_3) is in the Definition 10 is $(\pi_{12}(T_1),\pi_{23}(T_2),\pi_{13}(T_3)) = ((x,y),(y,z),(x,z))$ – a non-zero term of the matrix multiplication tensor $\langle|\Gamma|,|\Gamma|,|\Gamma|\rangle$. All the non-zero terms of the matrix multiplication tensor can be obtained this way: since, by construction of the communication hypergraph, any term $((x,y),(y,z),(x,z))$ can be lifted to an hyperedge $(T_1,T_2,T_3) \in H(I)$.

Finally, the statement about the image of the diagonal follows directly from the definition of permutations π_{12}, π_{23} and π_{13} and construction of the hypergraph H. ◀

▶ **Corollary 12.** *For any NOF permutation problem, the adjacency tensor of its communication hypergraph has maximal asymptotic subrank.*

3.3 Lower bounding the communication complexity of random NOF permutation problem

In this section we will show that a random NOF permutation problem over the alphabet $[N] \times [N] \times [N]$, where $N = 2^n$ requires $\geq n/3 - \mathcal{O}(1)$ deterministic communication. This is a simple counting argument, similar in spirit to the one used in [21]. Since the NOF permutation problems over $[N] \times [N] \times [N]$ are bijective with Latin squares of size $N \times N$, known upper and lower bounds for the number of Latin squares can be used in our proof.

The following is a consequence of the fact that any Latin square can be constructed by repeatedly taking a perfect matching in a regular bipartite graph, and using standard bounds on the number of perfect matchings in a k-regular bipartite graph.

▶ **Theorem 13** ([33]). *Total number $\#P$ of NOF permutation problems over $[N] \times [N] \times [N]$ satisfies*

$$N^{N^2} e^{-N^2} \leq \#P \leq N^{N^2}.$$

With this in hand, we can prove

▶ **Theorem 14.** *For most of NOF permutation problems (I, P), over the alphabet $[N]$ where $N = 2^n$, we have $\mathcal{CC}(I, P) \geq \frac{n}{3} - \mathcal{O}(1)$.*

Proof. Let (I, P) be a NOF permutation problem over $[N] \times [N] \times [N]$ with communication complexity d. By Theorem 9 communication complexity is equivalent to the existence of proper coloring $\kappa : I \to \{0,1\}^d$. Since I is a permutation problem, for any $x \in [N] \times [N]$, there is a unique $\pi_1^{-1}(x) \in I \subset ([N] \times [N])^3$ (where $\pi_1 : ([N] \times [N])^3 \to [N] \times [N]$ is the projection onto first coordinate). We can define $\kappa_1 : [N] \times [N] \to \{0,1\}^d$ as $\kappa_1(x) := \kappa(\pi_1^{-1}(x))$. Analogously, we define κ_2 and κ_3.

We observe that the triple of functions $(\kappa_1, \kappa_2, \kappa_3)$ uniquely determines I. Indeed, for $(a, b, c) \in P$ to check if $(a, b, c) \in I \iff \kappa_1(a) = \kappa_2(b) = \kappa_3(c)$. Therefore, the number of bits to specify a problem with communication complexity d (i.e. the logarithm of the number of such problems) is at most $3dN^2$ – to specify a function $\kappa : [N] \times [N] \to \{0,1\}^d$ we need dN^2 bits.

On the other hand, using the lower bound of Theorem 13, we need $N^2 \log N - \mathcal{O}(N^2)$ bits to write down arbitrary NOF permutation problem. Therefore if $d \leq (\log N)/3 - \mathcal{O}(1)$, most of the problems do not have a protocol with communication complexity d. ◀

3.4 Background on independence number and coloring of matrix multiplication tensor

The following statement asserting that matrix-multiplication tensor has almost maximal zero-out subrank is an important insight in the study of matrix multiplication. It has been originally proven by Strassen in [29], and has been used as a technical step in the design of some of the fast matrix multiplication algorithms. Here we will use it as a tool in the proof of Theorem 20. On the other hand, in Section 3.6 we will discuss how, using our connection between number on the foreheads protocol and structural properties of the matrix multiplication tensor, the following statement can be easily deduced from the known results in the NOF communication complexity on efficient protocols for some permutation problems.

▶ **Theorem 15** ([29]). *There is $I \subset \mathrm{supp}\langle N,N,N\rangle$, such that $|I| \geq N^{2-o(1)}$, and the restriction $\langle N,N,N\rangle|_{\pi_1(I),\pi_2(I),\pi_3(I)}$ is a permutation of the identity tensor, i.e. $Q_{zo}(\langle N,N,N\rangle) \geq N^{2-o(1)}$.*

Equivalently, in the notation introduced in this work,

$$\alpha(\langle N,N,N\rangle) \geq N^{2-o(1)}.$$

The following is a standard technique of lifting the large independent set to a small coloring in symmetric combinatorial objects. We include the proof for completeness.

▶ **Definition 16** (Automorphism group). *For a colored tensor $T = (I,P)$, we define $\mathrm{Aut}(T)$ to be the set of all triples of permutations $(\sigma_1,\sigma_2,\sigma_3)$ such that $\sigma(I) = I$, and moreover for all (x,y,z) we have $T_{x,y,z} = T_{\sigma_1(x),\sigma_2(y),\sigma_3(z)}$.*

▶ **Lemma 17.** *For a colored tensor $T = (I,P)$ if for any two green elements $a,b \in I$, there is a $\sigma \in \mathrm{Aut}(T)$, s.t. $\sigma(a) = b$, then*

$$\chi(T) \leq \frac{|I|}{\alpha(T)} \log |I|.$$

Proof. Let $S \subset I$ be an independent set, and $\sigma \sim \mathrm{Aut}(T)$ a uniformly random element of the automorphism group. Then clearly $\sigma(S)$ is also an independent set. We claim that for any $x \in I$, we have $\Pr(x \in \sigma(S)) = \frac{|S|}{|I|}$. Indeed, let $S = \{s_1, s_2, \dots s_m\}$. The subset $T \subset \mathrm{Aut}(T)$ of permutations σ such that $\sigma(s_1) = x$ is exactly a coset of a subgroup H in $\mathrm{Aut}(T)$ where H is a set of permutations fixing x, i.e. $H = \{\sigma : \sigma \in \mathrm{Aut}(T), \sigma(x) = x\}$. Number of such cosets is exactly $|I|$, since we assumed that x can be mapped to any other $s \in I$ by some permutation in $\mathrm{Aut}(T)$ (and no other element, by the property of automorphism group that each $\sigma \in \mathrm{Aut}(T)$ preserves green terms: $\sigma(I) = I$). Since all cosets of a given subgroup are disjoint and have the same size, we get $\Pr(\sigma(s_1) = x) = 1/|I|$). Therefore

$$\Pr(x \in \sigma(S)) = \sum_{i \leq |S|} \Pr(x = \sigma(s_i)) = |S|/|I|.$$

Finally, if we consider $K = C(|I| \log |S|)/|S|$ independent random permutations $\sigma_1, \sigma_2, \dots, \sigma_m \in \mathrm{Aut}(T)$, we wish to say that with positive probability for every element $x \in I$ it is covered by at least one set $\sigma_i(S)$. Indeed, in expectations, each of those is covered by $C \log |S|$ such sets, and applying a standard Chernoff and union bound argument, we reach the conclusion.

Having a covering of I by K independents sets, we can easily find a coloring $\tau : I \to [K]$ mapping an elemnet $x \in I$ to the smallest $i \in [K]$ such that $x \in \sigma_i(S)$. ◀

▶ **Corollary 18.** *The chromatic number of the matrix multiplication tensor satisfies*

$$\chi(\langle N,N,N\rangle) \leq N^{1+o(1)}.$$

We shall also need the following standard fact on the Kronecker products of matrix multiplication tensors.

▶ **Fact 19** (see, e.g., [9, Page 24]). *The Krnoecker product of matrix multiplication tensors satisfies*

$$\langle N_1, M_1, P_1\rangle \otimes \langle N_2, M_2, P_2\rangle = \langle N_1N_2, M_1M_2, P_1P_1\rangle$$

3.5 Laser method gives a non-trivial asymptotic protocol for all NOF permutation problems

In this section we will use the celebrated Laser method, introduced to design fast matrix multiplication algorithms [31, 19], to prove the following surprising upper bound on the asymptotic NOF communication complexity of the third player of arbitrary permutation problem.

▶ **Theorem 20.** *For every triple of NOF permutation problems I, J, K, each over $\{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n$, we have*

$$\mathcal{CC}(I \otimes J \otimes K) \leq (1 + o(1))n.$$

(Note that the trivial bound is $\mathcal{CC}(I \otimes J \otimes K) \leq 3n$.)

In particular for any NOF permutation problem P over $\{0,1\}^n \times \{0,1\}^n \times \{0,1\}^n$, we have

$$\underline{\mathcal{CC}}(P) \leq (1 + o(1))\frac{n}{3}.$$

We define the *block-colored tensor* to be a colored tensor (I, T) over $A \times B \times C$ together with a collection of partitions $A = A_1 \cup \ldots A_p$, $B = B_1 \cup \ldots \cup B_q$, and $C = C_1 \cup \ldots C_q$.

The *outer structure* of a block tensor T, is a tensor $\mathrm{Out}(T)$ over $[p] \times [q] \times [r]$ with $\mathrm{Out}(T)_{i,j,k} = 1$ if and only if the restriction $T|_{A_i,B_j,C_k} \neq 0$ (and $\mathrm{Out}(T)_{i,j,k} = 0$ otherwise). The *inner structure* $\mathcal{I}(T)$ of a block tensor T is a collection of (colored) tensors $T|_{A_i,B_j,C_k}$. The notions of outer and inner structure feature prominently in the Laser method; see e.g., [9, Section 8].

▶ **Lemma 21.** *For a block-colored tensor T, we have*

$$\chi(T) \leq \chi(\mathrm{Out}(T)) \cdot \max_{T' \in \mathcal{I}(T)} \chi(T').$$

Proof. This is immediate – to find a proper coloring for the set of green terms I of a block tensor T, we can map an element $x \in I$ to (c_1, c_2), where c_1 is a color assigned to the block in which x lies in the coloring of $\mathrm{Out}(T)$, and c_2 is a color assigned to x in the coloring of the block itself.

We need to show that each color class is an independent set. With an induced partition of A, B, C, this amounts to observing that for any block tensor for which $\mathrm{Out}(T)$ is is a permutation of the identity tensor, as well as all tensors in $\mathcal{I}(T)$, the entire tensor also is a permutation of the identity tensor. ◀

▶ **Fact 22.** *Let T^1, T^2 be a pair of block-colored tensor. Then $T := T^1 \otimes T^2$ is a block-colored tensor, with the outer structure $\mathrm{Out}(T) = \mathrm{Out}(T^1) \otimes \mathrm{Out}(T^2)$, and the inner structure $\mathcal{I}(T) \subset \mathcal{I}(T^1) \otimes \mathcal{I}(T^2)$ (Here, for two collections of tensors $\mathcal{A}$ and $\mathcal{B}$ we use $\mathcal{A} \otimes \mathcal{B} := \{a \otimes b : a \in \mathcal{A}, b \in \mathcal{B}\}$.)*

Proof. Directly follows from the definition. ◀

▶ **Fact 23.** *For any pair of colored tensors T_1, T_2, we have $\chi(T_1 \otimes T_2) \leq \chi(T_1)\chi(T_2)$.*

Proof. Given two colorings $\kappa_1 : I_1 \to [k_1]$ and $\kappa_2 : I_2 \to [k_2]$ we can construct a coloring of the Kronecker product $T_1 \otimes T_2$ using pairs $[k_1] \times [k_2]$, by mapping a term $((i_1, i_2), (j_1, j_2), (k_1, k_2)) \in I_1 \otimes I_2$ to a pair $(\kappa_1(i_1, j_1, k_1), \kappa_2(i_2, j_2, k_2))$. The fact that it is a proper coloring follows directly from the fact that Kronecker product of two identity tensors is again an identity tensor. ◀

▶ **Lemma 24.** *There is a triple of partitions $\mathcal{A},\mathcal{B},\mathcal{C}$ such that for any NOF permutation-problem (I, P_{NOF}), the block-colored tensor $T := (I, P_{\text{NOF}}, \mathcal{A}, \mathcal{B}, \mathcal{C})$ induced by those partitions satisfies the following conditions*

- *The outer structure* $\text{Out}(T) = \langle N, 1, 1\rangle$.
- *Every colored tensor in the inner structure* $(Y, T') \in \mathcal{I}(T)$ *has* $T' = \langle 1, N, N\rangle$ *and* $\chi(Y, T') = 1$.

Proof. Consider partitions $\mathcal{A} := (A_1, \ldots, A_n)$ where $A_t := \{(i, t) : i \in [N]\}$, similarly $\mathcal{B} := (B_1, \ldots B_N)$ where $B_t = \{(t, i) : i \in [N]\}$, and finally $\mathcal{C} := (C) := ([N] \times [N])$ is a trivial partition with a single part.

First note that with this partition of the matrix multiplication tensor P, the block $A_{t_1} \times B_{t_2} \times C$ is non-empty if and only if $t_1 = t_2$,

$$\text{Out}(T) = \sum_t x_{1,t} y_{t,1} z_{1,1},$$

is the matrix multiplication tensor $\langle N, 1, 1\rangle$.

Moreover, let us consider an arbitrary non-empty block $A_t \times B_t \times C$. Tensor T restricted to this block is

$$T|_{A_t \times B_t \times C} = \sum_{j,k} x_{j,t} y_{t,k} z_{k,j},$$

which is exactly a $\langle 1, N, N\rangle$ matrix multiplication tensor.

Note now that the projection $\pi_3 : A_t \times B_t \times C \to C$ induces a bijection between C and $Y := \text{supp}(T|_{A_t \times B_t \times C})$. If we look at $S := I \cap Y$, we wish to argue that S is an independent set in $T|_{A_t \times B_t \times C}$. Indeed, denoting $S_i := \pi_i(S)$, since (I, P_{NOF}) is a permutation problem, all we need to argue is that

$$(S_1 \times S_2 \times S_3) \cap Y = S.$$

But since π_3 is bijection, we already have $(A_t \times B_t \times \pi_3^{-1}(S_3)) \cap Y = S$. ◀

We are now ready to prove Theorem 20.

Proof of Theorem 20. For permutation NOF problems I, J, K, according to Lemma 24 we can chose a block-colored tensor structure T_I, T_J, T_K, such that $\text{Out}(T_I) = \langle N, 1, 1\rangle$, $\text{Out}(T_J) = \langle 1, N, 1\rangle$ and $\text{Out}(T_K) = \langle 1, 1, N\rangle$.

By Fact 22 and Fact 19, the block-colored tensor $T = T_I \otimes T_J \otimes T_K$ satisfies $\text{Out}(T) = \langle N, N, N\rangle$, and by Fact 23 every colored tensor $Z \in \mathcal{I}(T)$ has $\chi(Z) = 1$ (since it is a product of three colored tensors with chromatic number 1).

By Corollary 18, $\chi(\langle N, N, N\rangle) \le N^{1+o(1)}$, and finally using Lemma 21,

$$\chi(T) \le \chi(\text{Out}(T)) \max_{T' \in \mathcal{I}(T)} \chi(T') \le N^{1+o(1)}.$$

The communication complexity bound now follows from the characterization in Theorem 9. ◀

3.6 Zeroing out subrank of the matrix multiplication tensor

We will use the connection between number on the forehead communication complexity, and structural properties of matrix multiplication tensor to find a non-trivial *upper bound* for the zero-out subrank of matrix multiplication tensor – i.e., an upper bound on the size of the largest diagonal subtensor to which matrix multiplication tensor can be zeroed-out.

Authors of the work [26] observed a connection between the NOF communication complexity and the Ruzsa–Szemerédi problem. In particular, they proved a super-constant lower bound for NOF communication complexity of *any* permutation problem, as a corollary of the Ruzsa–Szemerédi theorem (which in turns is proved via the triangle-removal lemma).

Here, we observe that with our connection between NOF communication and the matrix-multiplication tensor, we can use their proof to show a somewhat stronger statement – the upper bound on the zero-out subrank of matrix multiplication.

▶ **Theorem 25.** *The zero-out subrank of matrix multiplication satisfies*

$$Q_{zo}(\langle N, N, N\rangle)) \leq N^2/2^{c\log^* N}.$$

Note that this theorem (together with Proposition 27) implies in fact the $\Omega(\log^* N)$ lower bound for all NOF permutation problems present in [26], and the proof is almost identical.

Before we continue, we shall state a convenient well-known equivalent formulation of the Ruzsa–Szemerédi theorem. The sub-quadratic upper bound of the quantity below has been famously proven by Ruzsa and Szemerédi in 1978 [28], using the notorious triangle removel lemma (in fact they considered a slightly different formulation, the $(6, 3)$-problem, but it can be easily seen to be equivalent to the following statement [16]). Substituting a quantitative version of the triangle removal lemma by Fox [23] in their proof leads to the following quantitative bound.

▶ **Theorem 26** ([28, 16, 23])**.** *For any graph $G(V, E)$, let $\mathcal{C}$ be a collection of all triangles in the graph G. If all those triangles are edge-disjoint, then $|\mathcal{C}| \leq N^2/2^{c\log^* N}$ where $N = |V|$.*

With this theorem in hand, it is easy to show a sub-quadratic upper bound on the largest independent set in the matrix multiplication tensor. In fact the quantitative question about the density of the largest independent set in the matrix multiplication tensor is equivalent to the Ruzsa–Szemerédi problem.

Proof of Theorem 25. Consider an independent set $\tilde{S} \subset \langle N, N, N\rangle$ – we can treat it as a subset $S \subset [N] \times [N] \times [N]$. Let us now consider the following tri-partite graph on $[N] \times [3]$: for each $(a, b, c) \in S$ we put a triangle on vertices $(a, 0), (b, 1)$ and $(c, 2)$. This construction yields a graph G on $3N$ vertices, together with a collection $\mathcal{C}$ of its $|S|$ triangles. To appeal to the Theorem 26, we need to show that triangles in the collection $\mathcal{C}$ are edge-disjoint, and that those are all the triangles in a graph G.

First of all, those are edge disjoint – if we had a pair of distinct triangles in $\mathcal{C}$ sharing an edge, it would correspond to two elements $(a, b, c_1) \in S$ and $(a, b, c_2) \in S$ – a contradiction with an assumption that S is an independent set in matrix multiplication tensor.

Similarly, since the graph G is tri-partite, the only possible triangle are of form $(a, 0), (b, 1), (c, 2)$ for some a, b, c. We wish to show that if S was an independent set, this is possible only for $(a', b', c') \in S$.

Indeed, if the edge $(a, 0), (b, 1)$ was included in the graph, there must have been a tuple $(a, b, c') \in S$, and analogously $(a, b', c) \in S, (a', b, c) \in S$ for some a', b', c'. Since S was an independent set, this readily implies that $a' = a, b' = b, c' = c$ and indeed $(a, b, c) \in S$. ◀

Lower bound for matrix multiplication subrank from NOF protocols

In the study of the matrix multiplication tensor, lower bounds on its zero-out subrank are much more consequential than upper bounds. Strassen proved that $\alpha(\langle N, N, N\rangle) \geq N^{2-o(1)}$, which is almost maximal (since the tensor $\langle N, N, N\rangle$ has size $N^2 \times N^2 \times N^2$, the largest diagonal tensor one could potentially hope to find there is of size N^2). He used this

together with the Laser method to design a fast matrix multiplication algorithm in [31], and all subsequent record-holding matrix multiplication algorithms have also used this. At the same time, recent barrier results [2, 1, 15] used the lower bounds on the subrank of matrix multiplication to prove barrier against certain approaches for proving that the matrix multiplication constant $\omega = 2$. (Roughly speaking, using our notation here, they show that since $\alpha(\langle N, N, N\rangle)$ is so large, one would need to use an intermediate tensor T with $\alpha(T)$ also large in order to design a sufficiently fast algorithm.)

One of the crucial ingredients in the proof using the Laser method that $\alpha(\langle N, N, N\rangle) \geq N^{2-o(1)}$ is leveraging the existence of a subset $S \subset [N]$ which does not have three-term arithmetic progressions, and has relatively high density. Using Behrend's [6] construction of such a set with density $2^{-\mathcal{O}(\sqrt{\log N})}$, it follows that $\alpha(\langle N, N, N\rangle) \geq N^2/2^{\mathcal{O}(\sqrt{\log N})}$.

Using the same construction of Behrend's set, it was proved in [12] that the NOF problem $\textsc{Eval}(\mathbb{Z}_N)$ has a deterministic protocol with communication complexity $\mathcal{O}(\sqrt{\log N})$ – significantly improving upon the naive $\mathcal{O}(\log N)$. We observe that their construction of an efficient NOF protocol not only uses the same technical ingredients as the construction of large independent set in the matrix multiplication tensor, but in fact it is much more intimately related – known results in communication complexity together with our connection directly imply the Strassen result on the subrank of matrix multiplication tensor.

▶ **Proposition 27.** *If there is any permutation problem $I \subset [N] \times [N] \times [N]$ with NOF communication complexity $\mathcal{CC}(\tilde{I}, P_{\text{NOF}}) \leq k$, then $\alpha(\langle N, N, N\rangle) \geq N^2 2^{-k}$.*

Proof. By Theorem 9, if the promise problem $(\tilde{I}, P_{\text{NOF}})$ has NOF communication complexity k, then $\chi(I, P_{\text{NOF}}) \leq 2^k$, and in particular taking the largest color we get $\alpha(I, P_{\text{NOF}}) \geq n^2 2^{-k}$. Since the promise tensor P_{NOF} for the number on the forehead communication problems is exactly the matrix multiplication tensor $\langle n, n, n\rangle$, we can disregard which terms are green, and deduce $\alpha(\langle n, n, n\rangle) \geq \alpha(I, P_{\text{NOF}}) \geq n^2 2^{-k}$. ◀

3.7 Other promise problems

As we understand now, the slice rank-based approach for proving lower bounds for asymptotic communication complexity of NOF permutation problems failed, because the technique depends only on the promise tensor, and is insensitive to the specific problem – and the promise tensor corresponding to the number-on-the-forehead problems is a matrix multiplication tensor, with maximal subrank.

With this in mind, it is worth looking at promise problems with other promise-tensors, hopefully ones for which we know the asymptotic subrank is not maximal – this statement alone shall imply that *every* permutation problem with such a promise should have positive asymptotic communication complexity.

▶ **Fact 28.** *If the promise tensor P over $\Sigma \times \Sigma \times \Sigma$ has non-maximal asymptotic subrank*

$$\underline{\alpha}(P) := \limsup \alpha(P^{\otimes N})^{1/N} < |\Sigma|,$$

then for every permutation problem $I \subset P$ we have

$$\mathcal{CC}((I, P)^{\otimes N}) \geq \Omega(N).$$

The proof of this fact is straightforward with what we have discussed so far. We later provide a complete proof of a strictly stronger theorem (Theorem 33), so we leave a direct proof of this special case to the reader.

The solution of the cap-set problem [20, 22, 32] and its subsequent generalizations [10], can be interpreted as saying that for any abelian group G, the structure tensor

$$T_G := \sum_{g,h \in G} x_g y_h z_{-g-h},$$

has $\underline{\alpha}(T_G) < |G|$. In particular, all permutation problems with a promise tensor being a structure tensor of such a group, have positive asymptotic communication complexity. For example, the following is true

▶ **Corollary 29.** *Consider the promise problem $EQ_{\mathbb{F}_3^N}$, where parties receive inputs $x, y, z \in \mathbb{F}_3^N$ with a promise that $x + y + z = 0$, and they wish to decide whether $x = y$. This problem requires $\Omega(N)$ communication.*

Without the preceding discussion it would not be immediately clear whether we should expect this statement to be true – the relevant part of the problem is between Alice and Bob, and they just wish to decide the equality predicate for their inputs – a task which, with ordinary deterministic two-party protocols, requires $\Omega(N)$ communication. But should we expect Charlie, who knows $x + y$ to be of much help in solving the task?

On the other hand, one might think that the statement above – if true – should be relatively straightforward to prove directly, with a proof similar to the communication lower bound for deterministic two-party equality problem. We observe that in fact this statement is easily *equivalent* to the cap-set theorem, hence a direct proof would be an interesting new proof of what used to be a cap-set *conjecture* for almost four decades.

▶ **Fact 30.** *For any abelian group G, if there is a $S \subset G$ of size $|G|^{1-\delta}$ without three-terms arithmetic progressions, then $\mathcal{CC}(EQ_G) \leq \delta \log |G| + \mathcal{O}(\log \log |G|)$*

Proof. The corresponding communication tensor of a problem EQ_G is (I, T_G) where T_G is the structure tensor of the group G, and $I = \{(x, x, -2x) : x \in G\} \subset T_G$.

We will first show that $\alpha(I, T_G) \geq |S|$. We wish to show that if S does not have three terms arithmetic progressions, then $S' := \{(x, x, -2x) : x \in S\} \subset I$ is an independent set. Indeed, for the sake of contradiction, let us assume that $(x, y, -x - y)$ for some $x \neq y$ has $x \in \pi_1(S'), y \in \pi_2(S'), -x - y \in \pi_3(S')$. This means $x, y \in S$, and $-x - y = -2z$ for some $z \in S$ – i.e. x, z, y is a three terms arithmetic progression.

To lift a large independent set to a small coloring, we will refer to Lemma 17 – it is enough to show that there is an automorphism of (I, T_G) that maps arbitrary $(x, x, -2x)$ to $(y, y, -2y)$. This is given by a triple of permutations $\sigma_1 = \sigma_2 : w \mapsto w - x + y$, and $\sigma_3 : w \mapsto w + 2y - 2x$. Finally, by Theorem 9 a coloring of the tensor (I, T_G) into $|G|^\delta \log |G|$ colors yields a protocol with complexity $\delta \log |G| + \log \log |G|$. ◀

3.8 Lower Bounds from the Asymptotic Spectrum of Tensors

The results in this section concern promise problems which are not necessarily permutation problems. We will first need a simple combinatorial description of the communication complexity of such a problem.

For any transcript of a protocol π, the set of inputs $S_\pi \subset \Sigma \times \Sigma \times \Sigma$ which result in such a transcript is a combinatorial cube $A_\pi \times B_\pi \times C_\pi$ for some $A_\pi, B_\pi, C_\pi \subset \Sigma$. If the protocol is correct, in every such cube either all terms from P are contained in I, or they all are not in I. As such, a correct protocol of complexity k induces a partition of the tensor P into 2^k monochromatic combinatorial cubes (where a cube is monochromatic in the preceding sense).

In particular, for any promise problems (I, P) all terms from I can be covered by $2^{\mathcal{CC}(I,P)}$ combinatorial cubes, each of which does not contain any term in $P - I$.

▶ **Theorem 31.** *Let $r : \mathbb{F}^{\Sigma\times\Sigma\times\Sigma} \to \mathbb{R}_{\geq 0}$ be any function which is:*

- sub-additive*, meaning, if $A, B \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ then $r(A+B) \leq r(A) + r(B)$, and*
- monotone under zeroing outs*, meaning, if $A, B \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ and B is a zeroing out of A, then $r(B) \leq r(A)$.*

Suppose P is any promise tensor over $\Sigma \times \Sigma \times \Sigma$, and $I \subset P$ is any problem (not necessarily a permutation problem). Then,

$$\mathcal{CC}(I, P) \geq \log_2 \left(\frac{r(I)}{r(P)} \right).$$

Proof. Let $k = 2^{\mathcal{CC}(I,P)}$. By the preceding discussion, there are k tensors $T_1, \ldots, T_k \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ which are each zeroing outs of P, such that $T_1 + \cdots + T_k = I$. We thus have

$$r(I) \leq \sum_{i=1}^{k} r(T_i) \leq k \cdot r(P),$$

as desired. ◀

Although it is relatively simple, Theorem 31 is quite powerful since there are many well-studied examples of functions r in algebraic complexity and tensor combinatorics which are sub-additive and monotone under zeroing outs. Tensor rank (R) is the most prominent example, but there are also other well-studied rank variants such as "slice rank" [32] and "geometric rank" [25]. (Note that subrank (Q) is super-additive, not sub-additive.)

Number In Hand

For one example, consider the all-1s tensor $P_{NIH} \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$, which is the promise tensor for the Number In Hand model. This very simple tensor has rank $R(P_{NIH}) = 1$. Thus, by Theorem 31, any tensor I with $R(I) \geq |\Sigma|^{\Omega(1)}$ has essentially maximal Number In Hand communication complexity $\Omega(\log |\Sigma|)$. Note that all tensors $I \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ have $R(I) \geq |\Sigma|$ except for tensors which are somehow "degenerate"; for instance, all "concise" tensors have this property [18].

This instantiation of Theorem 31, where the promise is the all-ones tensor and r is the rank of the problem tensor I, is akin to the standard lower bound in two-party deterministic communication complexity by the log rank of the communication matrix. The celebrated log-rank conjecture stipulates that in the standard setting this statement can be weakly inverted, and in fact any two-party communication problem (with trivial promise) has a protocol with complexity $(\log \operatorname{rank}(I))^c$ for some constant c.

Number On Forehead

For another example, consider the tensor $P_{NOF} \in \mathbb{F}^{\Sigma^2\times\Sigma^2\times\Sigma^2}$, which is the promise tensor for the Number On Forehead model. As we've seen, $P_{NOF} = \langle |\Sigma|, |\Sigma|, |\Sigma| \rangle$, and so we know that $R(P_{NOF}) = |\Sigma|^{\omega+o(1)}$. It follows that if $I \subset P_{NOF}$ is a problem with $R(I) \geq |\Sigma|^c$ for some constant $c > \omega$, then I has Number On Forehead communication complexity $\geq (c - \omega + o(1)) \log |\Sigma|$.

This relates the classic open problem in algebraic complexity of finding high-rank tensors to the task of proving deterministic NOF lower bounds. Furthermore, the fact that ω, the exponent of matrix multiplication, is *subtracted* in the communication lower bound is intriguing. This means that designing faster matrix multiplication algorithms, and hence decreasing our best-known upper bound on ω, actually makes it *easier* to prove NOF *lower bounds* via this approach.

Generalization to the Asymptotic Spectrum

Finally, in this section, we note that while subrank is not sub-additive, the zeroing out variant (Q_{ZO}) we have been studying is additive (and hence sub-additive) for direct sums, and so a similar version of Theorem 31 still holds for it. Moreover, this holds for any r in the "asymptotic spectrum of tensors' [31, 34], a well-studied class which roughly captures the different ways to generalize the notion of the "rank" of a matrix to tensors.

▶ **Definition 32.** *Tensor B is an* identification *of tensor A if you can get to B from A by first doing a zeroing out then setting some variables equal to each other. For example, $B = x_0y_0z_0 + x_0y_1z_1$ is an identification of $A = x_0y_0z_0 + x_1y_1z_1 + x_0y_1z_2$ by zeroing out z_2 and setting $x_1 \leftarrow x_0$. Identifications generalize zeroing outs, but are a special case of tensor restrictions.*

▶ **Theorem 33.** *Let $r : \mathbb{F}^{\Sigma\times\Sigma\times\Sigma} \to \mathbb{R}_{\geq 0}$ be any function which is:*

- sub-additive for direct sums*, meaning, if $A \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ and $B \in \mathbb{F}^{\Sigma'\times\Sigma'\times\Sigma'}$ are tensors for disjoint sets Σ, Σ', then $r(A + B) \geq r(A) + r(B)$, where $A + B$ is a tensor over $\mathbb{F}^{\Sigma\cup\Sigma'\times\Sigma\cup\Sigma'\times\Sigma\cup\Sigma'}$ and*
- monotone under identifications*, meaning, if $A, B \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ and B is an identification of A, then $r(B) \leq r(A)$.*

Suppose P is any promise tensor over $\Sigma \times \Sigma \times \Sigma$, and $I \subset P$ is any problem (not necessarily a permutation problem). Then,

$$\mathcal{CC}(I, P) \geq \log_2 \left(\frac{r(I)}{r(P)} \right).$$

Proof. Let $k = 2^{\mathcal{CC}(I,P)}$. By definition, there are k tensors $T_1, \ldots, T_k \in \mathbb{F}^{\Sigma\times\Sigma\times\Sigma}$ which are each zeroing outs of P, such that $T_1 + \cdots + T_k = I$. In particular, the direct sum of k copies of P has a zeroing out to the direct sum of all the T_i tensors, which in turn has an identification to I. We thus have

$$r(I) \leq \sum_{i=1}^{k} r(T_i) \leq k \cdot r(P),$$

as desired. ◀

▶ **Fact 34.** *The zeroing out subrank $r = Q_{ZO}$ satisfies the premises of Theorem 33.*

Proof. Q_{ZO} is sub-additive for direct sums since any zeroing out of a direct sum to a diagonal tensor must separately zero out each part into a diagonal tensor. It's monotone under identifications since, when transforming a tensor A to a diagonal tensor via identification, we may assume that we first perform a zeroing out and then set variables equal to each other, but setting variables equal to each other cannot increase the size of a diagonal tensor. ◀

3.9 Intermediate Group Promise Problems

Suppose $T_1 \subset T_2 \subset T_3$ are $\{0, 1\}$-valued tensors. These give rise to three promise problems, depending on which one picks as the promise and which one picks as the problem: (T_1, T_2), (T_1, T_3), and (T_2, T_3). We observe a simple triangle inequality-type relationship between their communication complexities:

▶ **Fact 35.** $\mathcal{CC}(T_1, T_3) \leq \mathcal{CC}(T_1, T_2) + \mathcal{CC}(T_2, T_3)$

Proof. In order to solve (T_1, T_3), first use a protocol for (T_2, T_3) to determine whether the input is in T_2, then use a protocol for (T_1, T_2) to determine whether the input is in T_1. ◀

We will now consider a particular instantiation of Fact 35 which yields a potential avenue toward proving deterministic NOF lower bounds. T_1, T_2, T_3 will be tensors from $\mathbb{F}^{\Sigma^2 \times \Sigma^2 \times \Sigma^2}$ for some finite set Σ. Let $n := |\Sigma|$.

We will pick T_3 to be the structure tensor of the group $(\mathbb{Z}/n\mathbb{Z})^2$. This tensor is typically written

$$T_3 = \sum_{a,b,c,d \in [n]} x_{(a,b)} y_{(c,d)} z_{(a+c,b+d)},$$

where additions in the subscripts (here and in the remainder of this subsection) are done mod n. However, to simplify the next step, we can permute the variable names (replacing $x_{(a,b)}$ with $x_{(a+b,b)}$ for all $a, b \in [n]$, replacing $y_{(c,d)}$ with $y_{(c,c+d)}$ for all $c, d \in [n]$, and replacing $z_{(a+c,b+d)}$ with $z_{(b+d,a+c)}$ for all $a, b, c, d \in [n]$) to equivalently write it as

$$T_3 = \sum_{a,b,c,d \in [n]} x_{(a+b,b)} y_{(c,d+c)} z_{(b+d,a+c)}.$$

Next, we will pick $T_2 = P_{\text{NOF}} \in \mathbb{F}^{n^2 \times n^2 \times n^2}$ to be the Number On Forehead promise tensor, i.e., the $\langle n, n, n \rangle$ matrix multiplication tensor,

$$T_2 = \sum_{i,j,k \in [n]} x_{(i,j)} y_{(j,k)} z_{(k,i)}.$$

▶ **Fact 36.** $T_2 \subset T_3$.

Proof. For each $i, j, k \in [n]$, we need to prove that there is a choice of $a, b, c, d \in [n]$ such that $x_{(i,j)} y_{(j,k)} z_{(k,i)} = x_{(a+b,b)} y_{(c,d+c)} z_{(b+d,a+c)}$. The choice which achieves this is $b = c = j$, $a = i - j \pmod n$, and $d = k - j \pmod n$. ◀

Finally, we may pick T_1 to be any Number On Forehead problem $T_1 \subset T_2$. Fact 35 now relates the Number On Forehead complexity of the problem T_1 to two different promise problems with the promise T_3:

$$\mathcal{CC}(T_1, P_{NOF}) \geq \mathcal{CC}(T_1, T_3) - \mathcal{CC}(P_{NOF}, T_3).$$

Using the tools we've developed thusfar, we can bound the communication complexity $\mathcal{CC}(P_{NOF}, T_3)$. Note that these are tensors over $\mathbb{F}^{n^2 \times n^2 \times n^2}$, so the trivial communication upper bound would be $2 \log n$.

▶ **Lemma 37.**

$$(\omega - 2) \log n \leq \mathcal{CC}(P_{NOF}, T_3) \leq \log(n).$$

Proof. Since P_{NOF} is the $\langle n, n, n \rangle$ matrix multiplication tensor, its rank is $R(P_{NOF}) = n^{\omega + o(1)}$. Meanwhile, since T_3 is the structure tensor of an abelian group, its rank is $R(T_3) = n^2$ (see, e.g., [17, Theorem 2.3 and Theorem 4.1]). Thus, Theorem 31 yields the lower bound $(\omega - 2) \log n \leq \mathcal{CC}(P_{NOF}, T_3)$.

For the upper bound, we directly give a communication protocol. In this problem, player A is given $(a + b, b)$, player B is given $(c, d + c)$, and player C is given $(b + d, a + c)$, for some values $a, b, c, d \in \mathbb{Z}_n$, and their goal is to determine whether or not the following three equalities hold: $b = c$, $d + c = b + d$, and $a + c = a + b$. Note that this is equivalent to testing whether $b = c$, since the other two equalities follow from this. They can do this by having player A send b, which uses $\log n$ bits, and then having player B confirm that it is equal to c. ◀

Interestingly, in light of the lower bound in Lemma 37, we know that improving the upper bound $\mathcal{CC}(P_{NOF}, T_3) \leq \log(n)$ *requires* using fast matrix multiplication. Indeed, any bound $\mathcal{CC}(P_{NOF}, T_3) \leq (1-\delta)\log(n)$ would imply that $\omega < 3 - \delta$. More specifically, one can observe that an upper bound on ω obtained in this way would fall under the "group-theoretic approach" to matrix multiplication [17], using the group $(\mathbb{Z}/n\mathbb{Z})^2$. There are known barriers to this approach for any fixed n [10], but it's consistent with the best-known barriers that this approach could prove $\omega = 2$ by using increasingly larger n. (On the other hand, most recent work on the group-theoretic approach has focused instead on non-abelian groups [11].)

Combining Lemma 37 together with Fact 35 shows that

▶ **Fact 38.**

$$\mathcal{CC}(T_1, P_{NOF}) \geq \mathcal{CC}(T_1, T_3) - \log(n).$$

For any T_1, the straightforward algorithm for the problem (T_1, T_3) has communication complexity $2\log(n)$. Fact 38 shows that any T_1 with $\mathcal{CC}(T_1, T_3) \geq (1+\delta)\log(n)$ for any fixed $\delta > 0$ would give a linear Number On Forehead lower bound $\mathcal{CC}(T_1, P_{NOF}) \geq \Omega(\log n)$.

In light of this, improving the upper bound of Lemma 37 could be quite powerful. In particular, if $\omega = 2$, and if this can be used to give an algorithm achieving $\mathcal{CC}(P_{NOF}, T_3) \leq o(\log(n))$, then any T_1 with $\mathcal{CC}(T_1, T_3) \geq \Omega(\log(n))$ (without any restriction on the leading constant) would give a linear Number On Forehead lower bound $\mathcal{CC}(T_1, P_{NOF}) \geq \Omega(\log n)$ as well.

References

1. Josh Alman. Limits on the universal method for matrix multiplication. *Theory Of Computing*, 17(1):1–30, 2021.
2. Josh Alman and Virginia Vassilevska Williams. Limits on all known (and some unknown) approaches to matrix multiplication. *SIAM Journal on Computing*, FOCS:18–285, 2021.
3. Noga Alon, Ankur Moitra, and Benny Sudakov. Nearly complete graphs decomposable into large induced matchings and their applications. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 1079–1090, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2213977.2214074.
4. Noga Alon and Adi Shraibman. Algorithmic number on the forehead protocols yielding dense ruzsa-szemerédi graphs and hypergraphs, 2020. doi:10.48550/arXiv.2001.00387.
5. Paul Beame, Matei David, Toniann Pitassi, and Philipp Woelfel. Separating deterministic from randomized multiparty communication complexity. *Theory of Computing*, 6(1):201–225, 2010.
6. Felix A Behrend. On sets of integers which contain no three terms in arithmetical progression. *Proceedings of the National Academy of Sciences*, 32(12):331–332, 1946.
7. Richard Beigel and Jun Tarui. On acc. *Computational Complexity*, 4:350–366, 1994.
8. Dario Bini. Border rank of a $p \times q \times 2$ tensor and the optimal approximation of a pair of bilinear forms. In *Automata, Languages and Programming: Seventh Colloquium Noordwijkerhout, the Netherlands July 14–18, 1980*, pages 98–108. Springer, 2005.
9. Markus Bläser. Fast matrix multiplication. *Theory of Computing*, pages 1–60, 2013.
10. Jonah Blasiak, Thomas Church, Henry Cohn, Joshua A Grochow, Eric Naslund, William F Sawin, and Chris Umans. On cap sets and the group-theoretic approach to matrix multiplication. *Discrete Analysis*, 3, 2017.
11. Jonah Blasiak, Henry Cohn, Joshua A Grochow, Kevin Pratt, and Chris Umans. Matrix multiplication via matrix groups. *arXiv preprint*, 2022. arXiv:2204.03826.
12. Ashok K Chandra, Merrick L Furst, and Richard J Lipton. Multi-party protocols. In *Proceedings of the fifteenth annual ACM symposium on Theory of computing*, pages 94–99, 1983.

13 Matthias Christandl, Omar Fawzi, Hoang Ta, and Jeroen Zuiddam. Symmetric subrank of tensors and applications, 2021. doi:10.48550/arXiv.2104.01130.

14 Matthias Christandl, Omar Fawzi, Hoang Ta, and Jeroen Zuiddam. Larger corner-free sets from combinatorial degenerations. In *13th Innovations in Theoretical Computer Science Conference, ITCS 2022, January 31 – February 3, 2022, Berkeley, CA, USA*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.ITCS.2022.48.

15 Matthias Christandl, Péter Vrana, and Jeroen Zuiddam. Barriers for fast matrix multiplication from irreversibility. *THEORY OF COMPUTING*, 17(2):1–32, 2021.

16 Lane H Clark, Roger C Entringer, Joseph E McCanna, and László A Székely. Extremal problems for local properties of graphs. *Australas. J Comb.*, 4:25–32, 1991.

17 Henry Cohn and Christopher Umans. A group-theoretic approach to fast matrix multiplication. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 438–449. IEEE, 2003.

18 Austin Conner, Fulvio Gesmundo, Joseph M Landsberg, Emanuele Ventura, and Yao Wang. Towards a geometric approach to strassen's asymptotic rank conjecture. *Collectanea mathematica*, 72:63–86, 2021.

19 Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing*, pages 1–6, 1987.

20 Ernie Croot, Vsevolod F Lev, and Péter Pál Pach. Progression-free sets in are exponentially small. *Annals of Mathematics*, pages 331–337, 2017.

21 Matei David and Toniann Pitassi. Separating nof communication complexity classes rp and np, 2008. doi:10.48550/arXiv.0802.3860.

22 Jordan S. Ellenberg and Dion Gijswijt. On large subsets of f_q^n with no three-term arithmetic progression, 2016. doi:10.48550/arXiv.1605.09223.

23 Jacob Fox. A new proof of the graph removal lemma. *Annals of Mathematics*, pages 561–579, 2011.

24 J. Hastad and M. Goldmann. On the power of small-depth threshold circuits. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*, pages 610–618 vol.2, 1990. doi:10.1109/FSCS.1990.89582.

25 Swastik Kopparty, Guy Moshkovitz, and Jeroen Zuiddam. Geometric rank of tensors and subrank of matrix multiplication. *arXiv preprint*, 2020. arXiv:2002.09472.

26 Nati Linial, Toniann Pitassi, and Adi Shraibman. On the communication complexity of high-dimensional permutations, 2019.

27 Nati Linial and Adi Shraibman. Larger corner-free sets from better nof exactly-n protocols, 2021. doi:10.48550/arXiv.2102.00421.

28 Imre Z Ruzsa and Endre Szemerédi. Triple systems with no six points carrying three triangles. *Combinatorics (Keszthely, 1976), Coll. Math. Soc. J. Bolyai*, 18(939-945):2, 1978.

29 V Strassen. Relative bilinear complexity and matrix multiplication. *Journal für die reine und angewandte Mathematik*, 375:406–443, 1987.

30 Volker Strassen. Vermeidung von divisionen. *Journal für die reine und angewandte Mathematik*, 264:184–202, 1973.

31 Volker Strassen. The asymptotic spectrum of tensors and the exponent of matrix multiplication. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 49–54. IEEE, 1986.

32 Terence Tao. A symmetric formulation of the Croot-Lev-Pach-Ellenberg-Gijswijt capset bound. *blog post*, 2016.

33 Jacobus Hendricus Van Lint and Richard Michael Wilson. *A course in combinatorics*. Cambridge university press, 2001.

34 Avi Wigderson and Jeroen Zuiddam. Asymptotic spectra: Theory, applications and extensions, 2022.

Instance-Wise Hardness Versus Randomness Tradeoffs for Arthur-Merlin Protocols

Dieter van Melkebeek
University of Wisconsin-Madison, WI, USA

Nicollas Mocelin Sdroievski
University of Wisconsin-Madison, WI, USA

Abstract

A fundamental question in computational complexity asks whether probabilistic polynomial-time algorithms can be simulated deterministically with a small overhead in time (the BPP vs. P problem). A corresponding question in the realm of interactive proofs asks whether Arthur-Merlin protocols can be simulated nondeterministically with a small overhead in time (the AM vs. NP problem). Both questions are intricately tied to lower bounds. Prominently, in both settings *blackbox* derandomization, i.e., derandomization through pseudo-random generators, has been shown equivalent to lower bounds for decision problems against circuits.

Recently, Chen and Tell (FOCS'21) established near-equivalences in the BPP setting between *whitebox* derandomization and lower bounds for multi-bit functions against algorithms on almost-all inputs. The key ingredient is a technique to translate hardness into targeted hitting sets in an instance-wise fashion based on a layered arithmetization of the evaluation of a uniform circuit computing the hard function f on the given instance.

In this paper we develop a corresponding technique for Arthur-Merlin protocols and establish similar near-equivalences in the AM setting. As an example of our results in the hardness to derandomization direction, consider a length-preserving function f computable by a nondeterministic algorithm that runs in time n^a. We show that if every Arthur-Merlin protocol that runs in time n^c for $c = O(\log^2 a)$ can only compute f correctly on finitely many inputs, then AM is in NP. Our main technical contribution is the construction of suitable targeted hitting-set generators based on probabilistically checkable proofs for nondeterministic computations.

As a byproduct of our constructions, we obtain the first result indicating that whitebox derandomization of AM may be equivalent to the existence of targeted hitting-set generators for AM, an issue raised by Goldreich (LNCS, 2011). Byproducts in the average-case setting include the first uniform hardness vs. randomness tradeoffs for AM, as well as an unconditional mild derandomization result for AM.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Hardness versus randomness tradeoff, Arthur-Merlin protocol, targeted hitting set generator

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.17

Funding Partial support for this research was provided by the University of Wisconsin-Madison Office of the Vice Chancellor for Research and Graduate Education with funding from the Wisconsin Alumni Research Foundation, and by the National Science Foundation under Grant No. 2312540.

Acknowledgements We thank Ronen Shaltiel and Chris Umans for answering questions about their work, Oded Goldreich for helpful feedback on the write-up, and Lijie Chen for suggesting the potential use of PCPs during a presentation of our preliminary results.

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 17; pp. 17:1–17:36

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

1 Introduction

The power of randomness constitutes a central theme in the theory of computing. In some computational settings, randomness is indispensable for any algorithmic solution. In others, it is provably needed for attaining efficiency. In yet others, the use of randomness leads to algorithms that run faster than all known deterministic ones, but the question remains open: Does an efficient deterministic algorithm exist?

In the context of decision problems, the key question is whether probabilistic polynomial-time algorithms with bounded error (the class BPP) can be simulated deterministically with a small overhead in time. In the realm of interactive verification protocols, the corresponding question asks whether Arthur-Merlin protocols (the class AM) can be simulated nondeterministically with a small overhead in time. In both settings, polynomial overhead is conjectured to suffice but even subexponential overhead remains open. Both settings have intricate connections to the quest for lower bounds, referred to as hardness vs. randomness tradeoffs. In some cases equivalences are known. We first describe the situation for BPP and then the one for AM, the focal point of this paper.

BPP setting. The first hardness vs. randomness tradeoffs were developed for *blackbox* derandomization, where a pseudo-random generator (PRG) produces, in an input-oblivious way, a small set of strings that "look random" to the process under consideration on every input of a given length. A long line of research established tight equivalences between *blackbox* derandomization of prBPP (the promise version of the class BPP) and *nonuniform* lower bounds for exponential-time classes. At the low end of the derandomization spectrum, subexponential-time blackbox derandomizations of prBPP are equivalent to super-polynomial circuit lower bounds for $\mathsf{EXP} \doteq \mathsf{DTIME}[2^{\mathsf{poly}(n)}]$ [2]. At the high end, polynomial-time blackbox derandomizations of prBPP are equivalent to linear-exponential circuit lower bounds for $\mathsf{E} \doteq \mathsf{DTIME}[2^{O(n)}]$ [15]. A smooth interpolation between the two extremes exists and yields tight equivalences over the entire derandomization spectrum [27]. The results are also robust in the sense that if the circuit lower bound holds at infinitely many input lengths (equivalent to the separation $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$ at the low end), then the derandomization works at infinitely many input lengths, and if the circuit lower bound holds at almost-all input lengths, then the derandomization works at almost-all input lengths.

A uniformization of the underlying arguments led to equivalences between derandomizations that work on *most* inputs of a given length, and *uniform* lower bounds, i.e., lower bounds against algorithms. This derandomization setting is often referred to as the *average-case* setting.[1] At the low end, there exist subexponential-time simulations of BPP that work on all but a negligible fraction of the inputs of infinitely many lengths if and only if $\mathsf{EXP} \not\subseteq \mathsf{BPP}$ [16]. Unfortunately, the known construction does not scale well (see [26, 7, 6] for progress toward an equivalence at the high end) and is not robust (a version for almost-all input lengths remains open). On the other hand, the result holds for blackbox derandomization as well as for general, "whitebox" derandomization, and implies an equivalence between blackbox and whitebox derandomization in this setting: If derandomization is possible at all, it can be done through pseudo-random generators.

This left open the setting of *whitebox* derandomizations that work for *almost all* inputs. For prBPP, such derandomizations are equivalent to the construction of *targeted* pseudo-random generators, which take an input x for the underlying randomized process, and produce a

[1] The underlying distribution may be the uniform one or any other polynomial-time sampleable distribution.

small set of strings that "look random" on that specific input x [9]. Recently, Chen and Tell [8] raised the question of an equivalent lower bound condition, and proposed a candidate: *uniform* lower bounds for *multi-bit* functions (rather than usual decision problems) that hold on *almost-all inputs* in the following sense.

▶ **Definition 1** (Hardness on almost-all inputs). *A computational problem f is hard on almost-all inputs against a class of algorithms if for every algorithm A in the class there is at most a finite number of inputs on which A computes f correctly.*

Chen and Tell started from the following observation about derandomization to hardness at the high end of the spectrum.

▶ **Proposition 2** (Chen and Tell [8]). *If* $\mathsf{prBPP} \subseteq \mathsf{P}$*, then for every constant c there exists a length-preserving function f that is computable in deterministic polynomial time and is hard on almost-all inputs against* $\mathsf{prBPTIME}[n^c]$.

Remarkably, they also established a converse, albeit with an additional uniform-circuit depth restriction on the hard function f. Their approach naturally yields a targeted *hitting-set generator* (HSG), the counterpart of a pseudo-random generator for randomized decision processes with one-sided error (the class RP and its promise version prRP).

▶ **Theorem 3** (Chen and Tell [8]). *Let f be a length-preserving function computable by logspace-uniform circuits of polynomial size and depth n^b for some constant b. If f is hard on almost-all inputs against* $\mathsf{prBPTIME}[n^{b+O(1)}]$*, where $O(1)$ denotes some universal constant, then* $\mathsf{prRP} \subseteq \mathsf{P}$.

Note that the hardness hypothesis of Theorem 3 necessitates the depth n^b of the uniform circuits computing the function f to be significantly less than their size. Otherwise, there exists even a deterministic algorithm that computes f in time $n^{b+O(1)}$.

The proof of Theorem 3 constructs a polynomial-time targeted hitting-set generator for prRP, which generically implies a polynomial-time targeted pseudo-random generator for prBPP, and thus that $\mathsf{prBPP} \subseteq \mathsf{P}$. Theorem 3 scales smoothly over the entire derandomization spectrum for prRP. Due to losses in the generic conversion from hitting sets to derandomizations for two-sided error, the corresponding result for prBPP does not scale that well. In particular, a low-end variant of Theorem 3 for prBPP remains open. That said, the results are robust in a similar sense as above with respect to input lengths. In fact, the approach inherently yields a much higher degree of robustness because it effectuates a hardness vs. randomness tradeoff on an input-by-input basis, as we explain further in the paragraph below about our techniques.

As a summary of the above discussion, Table 1 provides a qualitative overview of the lower bound equivalences for each of the three types of derandomization considered. We point out that, in the new setting of whitebox derandomizations that work on almost-all inputs, an actual equivalence along the lines of Chen and Tell [8] remains open due to the additional uniform-circuit depth requirement that is needed in the direction from hardness to derandomization. We refer to such results as *near-equivalences*. Follow-up works managed to obtain full-fledged equivalences in terms of other types of hardness, namely hardness of a computational problem related to Levin-Kolmogorov complexity [19] and hardness in the presence of efficiently-computable leakage [20].

AM setting. An equivalence corresponding to the first line of Table 1 is known throughout the entire spectrum [17, 22, 24]. The role of EXP is now taken over by $\mathsf{NEXP} \cap \mathsf{coNEXP}$, and the circuits are nondeterministic (or single-valued nondeterministic, or deterministic

Table 1 Equivalences between various types of derandomization and lower bounds.

Derandomization	Lower bound
blackbox, almost-all inputs	non-uniform
most inputs	uniform
whitebox, almost-all inputs	uniform, almost-all inputs

with oracle access to an NP-complete problem like SAT). The simulations use hitting-set generators for AM that are efficiently computable nondeterministically. Hitting-set generators are the natural constructs in the setting of AM because every Arthur-Merlin protocol can be efficiently transformed into an equivalent one with perfect completeness. As in the BPP setting, the lower bound equivalences for blackbox derandomization of prAM scale smoothly and are robust with respect to input lengths.

Regarding derandomizations that work on all but a negligible fraction of the inputs of a given length (the second line in Table 1), no hardness vs. randomness tradeoffs for AM were known prior to our work. What was known, are high-end results on derandomizations where no efficient nondeterministic algorithm can locate inputs on which the simulation is guaranteed to be incorrect [13, 25]. Indeed, the authors of [13] explicitly mention the average-case setting and why their approach fails to yield average-case simulations that are correct on a large fraction of the inputs. The setting corresponding to the third line in Table 1 was not studied before.

Main results. As our main results, we obtain near-equivalences in this third setting, i.e., between whitebox derandomizations of Arthur-Merlin protocols that work on almost-all inputs, on the one hand, and hardness on almost-all inputs against Arthur-Merlin protocols, on the other hand.

We start from a similar observation in the derandomization to hardness direction as the one Chen and Tell made for BPP at the high end of the spectrum.

▶ **Proposition 4.** *If* $\mathsf{prAM} \subseteq \mathsf{NP}$*, then for every constant* c *there exists a length-preserving function* f *that is computable in nondeterministic polynomial time with "a few" bits of advice, and is hard on almost-all inputs against* $\mathsf{AMTIME}[n^c]$*.*

We refer to Section 5.1 for the quantification of "a few".

Importantly, we are able to establish an almost-converse of Proposition 4. Under a slightly stronger hardness assumption, we construct a targeted hitting-set generator for prAM that is computable in nondeterministic polynomial time, yielding the following derandomization result.

▶ **Theorem 5.** *Let* f *be a length-preserving function computable in nondeterministic time* n^a *for some constant* a*. If* f *is hard on almost-all inputs against* $\mathsf{prAMTIME}[n^c]$ *for* $c = O((\log a)^2)$*, where* $O(\cdot)$ *hides some universal constant, then*

$$\mathsf{prAM} \subseteq \mathsf{NP}.$$

Note that, in contrast to Theorem 3 in the BPP setting, Theorem 5 in the AM setting has no uniform-circuit depth restriction on the function f. Together with Proposition 4, Theorem 5 represents a near-equivalence between $\mathsf{prAM} \subseteq \mathsf{NP}$ and hardness on almost-all inputs of

length-preserving[2] functions against Arthur-Merlin protocols. Whereas in the BPP setting, the remaining gap relates to uniform-circuit depth, in the AM setting the remaining gap relates to the advice and the technical distinction between AM and prAM protocols. We point out that the approaches in [19] and [20], which yield full-fledged equivalences in the BPP setting, do not seem compatible with the AM setting [18].

Both Proposition 4 and Theorem 5 scale quite smoothly across the derandomization spectrum. The generalization of Theorem 5 has the following form: Let f be a length-preserving function computable in nondeterministic time $T(n)$. If f is hard on almost-all inputs against $\mathsf{prAMTIME}[t(n)]$, then $\mathsf{prAM} \subseteq \mathsf{NTIME}[\mathsf{poly}(T(n))]$. Intuitively, we may think of $t(n)$ as only slightly smaller than $T(n)$ for high-end results and much smaller for low-end results. Pushing our techniques as far as possible toward the low end, we obtain the following variant of Theorem 5.

▶ **Theorem 6.** *Let f be a length-preserving function computable in nondeterministic exponential time. If f is hard on almost-all inputs against $\mathsf{prAMTIME}[n^{b(\log n)^2}]$ for all constants b, then for some constant c*

$$\mathsf{prAM} \subseteq \mathsf{NTIME}[2^{n^c}]. \tag{1}$$

As $\mathsf{prAM} \subseteq \mathsf{NEXP}$ trivially holds, the conclusion (1) of Theorem 6 represents the very low end of the derandomization spectrum. Note that a perfectly smooth scaling of Theorem 5 would only need a polynomial lower bound to arrive at the conclusion of Theorem 6, but the hypothesis of Theorem 6 requires a lower bound of $n^{\omega((\log n)^2)}$. We remark that the same discrepancy shows up in the current best-scaling uniform hardness vs. randomness tradeoffs for AM [25]. We refer to Theorem 27 in Section 4 for the full scaling and to Table 2 in the same section for other interesting instantiations.

Byproducts. Using our targeted hitting-set generators we are able to make progress on a number of related topics. We mention three representative ones here; more are described in the body of the paper.

First, there is the relationship between whitebox derandomization of prAM and the existence of targeted hitting-set generators for prAM. In the paper [9] where Goldreich introduced targeted pseudo-random generators for prBPP and showed that their existence is equivalent to whitebox derandomization of prBPP, he asked about analogous results for prAM. To the best of our knowledge, there have been no prior results along those lines. We take a first step toward an equivalence in this setting.

▶ **Theorem 7.** *If $\mathsf{prAMTIME}[2^{\mathsf{polylog}(n)}] \subseteq \text{io-}\mathsf{NEXP}$, then there exists a targeted hitting-set generator for prAM that yields the simulation $\mathsf{prAM} \subseteq \text{io-}\mathsf{NTIME}[2^{n^c}]/n^\epsilon$ for some constant c and all $\epsilon > 0$.*

Second, we establish the first hardness vs. randomness tradeoffs for Arthur-Merlin protocols in the average-case setting. Informally, under a high-end worst-case hardness assumption, we obtain nondeterministic polynomial-time simulations of prAM that are correct on all but a negligible fraction of the inputs.

[2] The focus on length-preserving functions f in Proposition 4 and Theorem 5 is for concreteness. For Proposition 4 to hold, the number of output bits needs to grow with n in an efficiently computable fashion. For Theorem 5 any number of output bits suffices as long as there are not so many that the function f becomes trivially hard for Arthur-Merlin protocols running in time n^c.

▶ **Theorem 8.** *If* $\mathsf{NTIME}[2^{an}] \cap \mathsf{coNTIME}[2^{an}] \not\subseteq \mathsf{BPTIME}[2^{(\log(a+1))^2 n}]^{\mathsf{SAT}}_{||}$ *for some constant* $a > 0$*, then for every problem in* prAM *and all* $e > 0$ *there exists a simulation of the problem in* NP *that is correct on all but a fraction* $1/n^e$ *of the inputs of length* n *for infinitely many lengths* n*.*

The class $\mathsf{BPTIME}[t(n)]^{\mathsf{SAT}}_{||}$ denotes probabilistic algorithms with bounded error that run in time $t(n)$ and can make parallel (i.e., non-adaptive) queries to an oracle for SAT. Theorem 8 answers a question in [13], which presents results in the different but related "pseudo" setting, where the simulation may err on many inputs of any given length, but no polynomial-time nondeterministic algorithm can pinpoint an error at that length. We remark that our technique also leads to identical results in the "pseudo" setting by replacing the hardness assumption with hardness against $\mathsf{AMTIME}[t(n)]$.

The model $\mathsf{prBPP}^{\mathsf{SAT}}_{||}$ was used as a proxy for prAM in the initial derandomization results for Arthur-Merlin protocols [17] and is seemingly more powerful. However, derandomization results for prAM typically translate into similar derandomization results for $\mathsf{prBPP}^{\mathsf{SAT}}_{||}$. In particular, the conclusion $\mathsf{prAM} \subseteq \mathsf{NP}$ of Theorem 5 implies that $\mathsf{prBPP}^{\mathsf{SAT}}_{||} \subseteq \mathsf{P}^{\mathsf{SAT}}_{||}$, and the conclusion $\mathsf{prAM} \subseteq \mathsf{NTIME}[2^{n^c}]$ for some constant c in Theorem 6 implies that $\mathsf{prBPP}^{\mathsf{SAT}}_{||} \subseteq \mathsf{DTIME}[2^{n^c}]^{\mathsf{SAT}}_{||}$ for some constant c. In the case of Theorem 8, we argue that the hardness assumption implies simulations of $\mathsf{prBPP}^{\mathsf{SAT}}_{||}$ in $\mathsf{P}^{\mathsf{SAT}}_{||}$ of the same strength as the simulations of prAM in NP. This way, we obtain a hardness vs. randomness tradeoff in which the hardness model and the model to-be-derandomized match, namely probabilistic algorithms with bounded error and non-adaptive access to an oracle for SAT.

As our third byproduct, we present an unconditional mild derandomization result for AM in the average-case setting. By a *mild* derandomization of AM we mean a nontrivial simulation on Σ_2-machines. Recall that $\mathsf{AM} \subseteq \Pi_2\mathsf{P}$, and proving that $\mathsf{AM} \subseteq \Sigma_2\mathsf{P}$ is a required step if we hope to show that $\mathsf{AM} \subseteq \mathsf{NP}$. It is known that AM can be simulated (at infinitely many input lengths n) on Σ_2-machines that run in subexponential time and take n^c bits of advice for some constant c [28]. It remains open whether AM can be simulated on Σ_2-machines in subexponential time with subpolynomial advice. Indeed, such a simulation for prAM would imply lower bounds against nondeterministic circuits that are still open [1]. We show an unconditional subexponential-time and subpolynomial-advice Σ_2-simulation for prAM in the average-case setting.

▶ **Theorem 9.** *For every problem in* prAM *and every constant* $\epsilon > 0$ *there exists a simulation of the problem in* $\Sigma_2\mathsf{TIME}[2^{n^\epsilon}]/n^\epsilon$ *that is correct on all but a fraction* $1/n^e$ *of the inputs of length* n*, for all constants* e *and infinitely many lengths* n*.*

In fact, we can extend Theorem 9 to $\mathsf{prBPP}^{\mathsf{SAT}}_{||}$ in lieu of prAM.

Techniques. For our main result, we develop an instance-wise transformation of hardness into targeted hitting sets tailored for AM. In the setting of BPP, Chen and Tell combine the Nisan-Wigderson pseudo-random generator construction [23] with the doubly-efficient proof systems of Goldwasser, Kalai, and Rothblum [12] (as simplified in [10]). The latter allows them to capture the computation of a uniform circuit of size T and depth d for f on a given input x by a downward self-reducible sequence of polynomials, which they use to instantiate the NW generator. In case the derandomization of a one-sided error algorithm on a given input x fails, a bootstrapping strategy à la [16], based on a learning property of the NW generator, allows them to retrieve the value of $f(x)$ in time $O(d \cdot \mathsf{polylog}(T))$. Thus, provided the depth d is small compared to the size T, either the derandomization on input x works or else the computation of $f(x)$ can be sped up.

A similar approach based on [12] applies to the AM setting by replacing the NW construction with a hitting-set generator construction for AM that also has the learning property. Like in the BPP setting, the construction is only of interest when the circuits for f have relatively small depth. Moreover, the construction can only handle a limited amount of nondeterminism in the computation for f, whereas the direction from derandomization to hardness seems to require more.

In order to remedy both shortcomings, we develop a new method to extract hardness from a nondeterministic computation on a given input x, based on probabilistically checkable proofs rather than [12]. The soundness of our method presupposes some type of resilience of the underlying regular pseudo-random generator. The required property was first identified and used by Gutfreund, Shaltiel and Ta-Shma [13] for the Miltersen-Vinochandran generator MV [22], and later by Shaltiel and Umans [25] for their recursive variant of the MV generator, RMV. We combine RMV with the probabilistically checkable proofs of Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan [4] to transform hardness into pseudo-randomness for AM in an instance-wise fashion, without any uniform-circuit depth restriction or limitation on the amount of nondeterminism.

We highlight one strong feature of *all* instance-wise approaches. If the hardness condition holds on almost-all inputs, then the derandomization works on almost-all inputs. This is the setting in which we stated the results of Chen and Tell and our main results. Similarly, if the hardness condition holds on all inputs of a given length, then the derandomization works on all inputs of that length. This is the robustness property that we alluded to earlier. However, an instance-wise approach yields much more, including average-case derandomization results: To obtain a nondeterministic simulation for some prAM problem that works with high probability over any given distribution, it suffices to assume that every prAM protocol can only compute the hard function f with low probability over that same distribution.

Our derandomization-to-hardness result follows by diagonalization, as does the one by Chen and Tell. To obtain our byproducts, we combine our targeted hitting-set generator with several other ingredients, including diagonalization, the "easy-witness" method and traditional hardness vs. randomness tradeoffs. Our average-case derandomization results require a modification of our targeted HSG so that it respects a stronger resilience property. Along the way to our unconditional mild derandomization result, we establish an "easy witness lemma" for Σ_2 computations, which may be of independent interest.

Organization. In Section 2, we develop the ideas behind our results and relate them to existing techniques. We start the formal treatment in Section 3 with definitions, notation, and other preliminaries. In Section 4, we construct our targeted HSG and establish our hardness-to-derandomization results that make use of it (Theorems 5 and 6). Section 5 presents the derandomization-to-hardness side of our near-equivalence, as well as a proof of our byproduct on derandomization to targeted hitting-set generators (Theorem 7). In Section 6, we derive our derandomization byproducts under uniform worst-case hardness (the average-case simulation of Theorem 8 as well as a simulation that works on all inputs of infinitely many lengths). Section 7 contains our unconditional mild derandomization result for AM (Theorem 9).

2 Technical overview

In this section, we start with an overview of techniques used in prior hardness vs. randomness tradeoffs for BPP and AM in a way that facilitates a high-level exposition of our main hardness-to-derandomization result for AM. We also provide the intuition for our derandomization-to-hardness result and for our byproducts.

2.1 Main results

We start with an overview of the techniques used for hardness-to-derandomization results in the traditional setting for BPP (lines 1 and 2 in Table 1), followed by those in the new setting (line 3 in Table 1). We then transition to AM, discuss the additional challenges, the known techniques in the traditional setting and, finally, our results in the new setting.

Traditional setting for BPP. The key ingredient in all known hardness vs. randomness tradeoffs is a pseudo-random generator construction G that takes a function h as an oracle and produces a pseudo-random distribution G^h with the following property: Any statistical test D that distinguishes G^h from uniform suffices as an oracle to efficiently learn h approximately from a small number of queries. Thus, if G^h does not "look random" to an efficient randomized process A on an input x, an approximation to h can be reconstructed efficiently when provided with x and the values of h on a small number of points, as well as oracle access to the distinguisher $D(r) = A(x, r)$, where $A(x, r)$ denotes the output of A on input x and random-bit string r. If the function h can be self-corrected (e.g., by being random self-reducible or by its truth table being a codeword in a locally-correctable error-correcting code), then the exact function h can be reconstructed efficiently.

In order to obtain hardness vs. randomness tradeoffs from pseudo-random generator constructions with the learning property, two questions need to be addressed:

1. How to obtain the distinguishers D?
2. How to obtain the answers to the learning queries?

The first question asks how to find inputs x on which the process A is not fooled by G^h. In the non-uniform setting such an input can be included in the advice. In the uniform setting for BPP, such inputs can be found by sampling x at random and testing for a difference in behavior of $D \doteq A(x, \cdot)$ between the uniform and the pseudo-random distributions, which can be done in prBPP.

Regarding the second question, in the non-uniform setting, the answers to the learning queries can also be provided as advice. In the uniform setting, [16] employs a function h that is not only random self-reducible but also downward self-reducible, and uses the downward self-reduction to answer the learning queries for length n by evaluating the circuit that resulted from the reconstruction for length $n - 1$. This bootstrapping strategy presupposes that the reconstruction works at almost-all input lengths. This is why we only know how to obtain simulations that are correct at infinitely many input lengths in the uniform setting for BPP.

New setting for BPP. In the setting of line 3 in Table 1, the role of pseudo-random generators is taken over by *targeted* pseudo-random generators. Whereas PRGs are oblivious to x (beyond its length), targeted PRGs take x as an input and are only supposed to fool the randomized process on that particular x. This approach obviates the problem of obtaining the distinguisher D (question 1 above) as we can use $D = A(x, \cdot)$ for the given x. Targeted PRGs can be constructed from a PRG G by instantiating G with an oracle $h = h_x$ that depends on x. This raises a third question in the application of a PRG for hardness vs. randomness tradeoffs:

3. How to obtain the function h_x from x?

Chen and Tell [8] use the doubly-efficient proof systems of Goldwasser, Kalai, and Rothblum [12] (as simplified in [10]) to obtain h_x from x and combine it with the Nisan-Wigderson pseudo-random generator construction [23]. The GKR proof system takes a

logspace-uniform family of circuits of size $T(n)$ and depth $d(n)$ computing a (multi-bit) Boolean function f, and transforms the circuit for f on a given input x into a downward self-reducible sequence of multi-variate low-degree polynomials $\hat{g}_{x,0} \ldots, \hat{g}_{x,d'(n)}$ where $d'(n) = O(d(n) \log(T(n)))$. The polynomial $\hat{g}_{x,0}$ is efficiently computable at any point given input x, and the value of $f(x)$ can be extracted efficiently from $\hat{g}_{x,d'(n)}$. We refer to the sequence of polynomials as a *layered arithmetization* of the circuit for f on input x.

Chen and Tell instantiate the NW generator with the Hadamard encoding of each of the polynomials $\hat{g}_{x,i}$ as the function $h = h_{x,i}$, and follow a bootstrapping strategy similar to [16] to construct $\hat{g}_{x,d'(n)}$ from $\hat{g}_{x,0}$. For the strategy to work, the NW reconstructor needs to succeed at every level. This is the reason why Chen and Tell only end up with a (targeted) *hitting-set* generator rather than a *pseudo-random* generator. The time required by the bootstrapping process is proportional to the number of layers and thus to the depth $d(n)$ of the circuit computing f. By setting the parameters of the arithmetization appropriately, the dependency on the size $T(n)$ is only polylogarithmic. This is what enables the reconstruction to compute $f(x)$ very quickly as long as the depth $d(n)$ is not too large.

Liu and Pass [20] also use the NW generator but obtain h_x as an encoding of the value of $f(x)$ itself, where f is an almost-all inputs leakage-resilient hard function (a function that remains hard even if some efficiently-computable information about $f(x)$ is leaked to an attacker). The answers to the learning queries are provided as part of the information about $f(x)$ that is leaked, which allows them to reconstruct $f(x)$ directly and efficiently. This approach leads to a (targeted) *pseudo-random* generator since it only involves a single instantiation of the NW generator. Reversing the hardness-to-derandomization direction yields an equivalence between derandomization of prBPP and the existence of almost-all inputs leakage-resilient hard functions.

Transition to AM. A number of changes are in order in terms of the requirements for similar results for AM. First, we need to handle *co-nondeterministic* distinguisher circuits D instead of deterministic ones. Co-nondeterministic circuits suffice because Arthur-Merlin protocols can be assumed to have perfect completeness. The only requirement for a correct derandomization is in the case of negative instances, in which case we want to hit the set of Arthur's random-bit strings for which Merlin cannot produce a witness. By the soundness property of the Arthur-Merlin protocol, the set contains at least half of the random-bit strings.

Second, we need to accommodate *nondeterministic* algorithms computing the function f. This is because the direction from derandomization to hardness seems to need them (see Proposition 4). On each input x, such an algorithm needs to have at least one successful computation path, and on every successful computation path, the output should equal $f(x)$.

Third, the algorithm for the targeted hitting-set generator can also be nondeterministic, which is natural when the algorithm for f is nondeterministic. In the case of a generator, the nondeterministic algorithm should still have at least one successful computation path on every input, but it is fine to produce different outputs on different successful computation paths. For any given x and D, on every successful computation path, the output should be a hitting set for D. This allows us to nondeterministically simulate a promise Arthur-Merlin protocol on input x as follows: Guess a computation path of the targeted HSG; if it succeeds, say with output S, guess a computation path for the Arthur-Merlin protocol on input x using each of the elements in S as the random-bit string, and accept if all of them accept; otherwise, reject.

Finally, we need to be able to run the reconstruction procedure as a (promise) *Arthur-Merlin protocol*. This is because we want the model in which we can compute $f(x)$ in case of a failed derandomization on input x, to match the class we are trying to derandomize. There are two requirements for the protocol to compute $f(x)$ on input x:

- *Completeness* demands that there exists a strategy for Merlin that leads Arthur to succeed with output $f(x)$ with high probability.
- *Soundness* requires that, no matter what strategy Merlin uses, the probability for Arthur to succeed with an output other than $f(x)$ is small.

The reconstructor naturally needs the power of nondeterminism in order to simulate the distinguisher D. Making sure the reconstructor is sound and needs no more power than prAM is the challenge.

Traditional setting for AM. In reference to the first two questions above, the answer to the one about obtaining a distinguisher D is similar as for BPP, except that in the uniform setting we do not know how to check in prAM for a difference in behavior of $D \doteq A(x, \cdot)$ between the uniform and the pseudo-random distributions. This is why average-case results remain open for AM. Instead, one assumes that some nondeterministic algorithm produces, on every successful computation path on input 1^n, an input x of length n on which the difference in behavior is guaranteed.

As for obtaining answers to the learning queries in the uniform setting for AM, we can make use of the nondeterminism allowed during the reconstruction and ask Merlin to provide the answers to the learning queries. However, we need to guard against a cheating Merlin. A strategy proposed by Gutfreund, Shaltiel and Ta-Shma in [13] consists of employing a function h that has a length-preserving instance checker. After Merlin has provided the supposed answers to the learning queries, to compute $h(z)$ for a given input z, we run the instance checker on input z and answer the queries y of the instance checker by running the evaluator part of the reconstruction process on input y. All the runs of the evaluator can be executed in parallel, ensuring a bounded number of rounds overall, which can be reduced to two in the standard way at the cost of a polynomial blowup in the running time [3].

To guarantee soundness, the reconstruction process needs to have an additional *resilience* property, namely that it remains partial single-valued even when the learning queries are answered incorrectly. Two hitting-set generators tailored for AM are known to have the property: the Miltersen-Vinodchandran generator MV [22], which is geared toward the high end, and a recursive version, RMV, developed by Shaltiel and Umans [25] to cover a broader range. MV is used for the high end in [13], and RMV for the rest of the spectrum in [25].

New setting for AM. We build a targeted hitting-set generator for AM based on the RMV hitting-set generator. To obtain h_x from x, we make use of Probabilistically Checkable Proofs (PCPs) for the nondeterministic computation of the string $f(x)$ from x. Let V denote the verifier for such a PCP system that uses $O(\log(T(n))$ random bits and $\mathsf{polylog}(T(n))$ queries for nondeterministic computations that run in time $T(n)$. On input x, our targeted HSG guesses the value of $f(x)$ and a candidate PCP witness y_i for the i-th bit of $f(x)$ for each i, and runs all the checks of the verifier V on y_i (by cycling through all random-bit strings for V). If all checks pass, our targeted HSG instantiates RMV with y_i for each i as (the truth table of) the oracle h_x, and outputs the union of all the instantiations as the hitting set, provided those nondeterministic computations all accept; otherwise, the targeted HSG fails.

For the reconstruction of the i-th bit of $f(x)$, Arthur generates the learning queries of the RMV reconstructor for the oracle y_i, and Merlin provides the purported answers as well as the value of the i-th bit of $f(x)$. Arthur then runs some random checks of the verifier V on

input x, answering the verifier queries by executing the evaluator of the RMV reconstructor. All the executions of the evaluator can be performed in parallel, ensuring a bounded number of rounds overall. The resilient partial single-valuedness property of the RMV reconstructor guarantees that the verifier queries are all consistent with some candidate proof $\tilde{y}_i$. The completeness and soundness of the PCP then imply the completeness and soundness of the reconstruction process for our targeted HSG. As V makes few queries and is very efficient, the running time of the process is dominated by the running time of the RMV reconstructor.

Abstracting out the details of our construction and how the distinguisher D is obtained, the result can be captured in two procedures: a nondeterministic one, H, which has at least one successful computation path for every input and plays the role of a targeted hitting-set generator, and a promise Arthur-Merlin protocol, R, which plays the role of a reconstructor for the targeted hitting-set generator. H and R have access to the input x and a co-nondeterministic circuit D, and have the following property.[3]

▶ **Property 10.** *For every $x \in \{0,1\}^*$ and for every co-nondeterministic circuit D that accepts at least half of its inputs, at least one of the following holds:*

1. *$H(x, D)$ outputs a hitting set for D on every successful computation path.*
2. *$R(x, D)$ computes $f(x)$ in a complete and sound fashion.*

Theorem 5 follows by considering nondeterministic running time $T(n) = n^a$ and co-nondeterministic circuits D of size n^c for some $c > 1$. In this regime, H runs in time $n^{O(a+c)}$ and R in time $n^{O(c(\log a)^2)}$. Under the hypothesis of Theorem 5, the second item in Property 10 cannot happen except for finitely many x of length n, so the first item needs to hold. For any constant $c' < c$, this yields a polynomial-time targeted hitting-set generator for $\mathsf{prAMTIME}[n^{c'}]$, which can be used for all of prAM by padding. Theorem 6 follows along the same lines; the running time is dictated by the RMV reconstructor.

We point out that the approach of Chen and Tell can be ported to the AM setting by replacing NW with a generator for AM that has the learning property and a reconstructor running in prAM. The nondeterminism allows us to run the bootstrapping process in parallel, so the number of rounds of Arthur and Merlin remains bounded, but the overall running time remains proportional to the depth of the circuits for f. This means that, like in the setting of BPP, this approach only yields meaningful results when the depth is small compared to the size. Nondeterministic circuits for f can be accommodated in this approach by treating them as deterministic circuits with nondeterministic guess bits as additional inputs. However, this limits the amount of nondeterminism that can be handled. Our approach based on PCPs remedies the limitations on depth as well as nondeterminism.

Derandomization to hardness. Our derandomization-to-hardness result is proven by diagonalization. Under the $\mathsf{prAM} \subseteq \mathsf{NP}$ assumption, every fixed-polynomial time AM protocol computing a length-preserving function can be simulated in nondeterministic fixed-polynomial time. We would like to diagonalize against these simulating nondeterministic machines to construct our hard function. Due to the lack of an almost-everywhere hierarchy result for NTIME, we do not know how to do this efficiently for generic nondeterministic machines. This is where the advice comes to rescue: We use advice to indicate which nondeterministic

[3] The dependency of H on D is only through the number of input bits of D. For R, blackbox access to D suffices (in addition to the input x). However, we may as well give both H and R full access to the input x and the circuit D. In the intended application, the co-nondeterministic circuit D is obtained by hardwiring the input x into the Arthur-Merlin protocol being derandomized, but this is not essential for the construction.

machines are *single-valued* at a particular input length. We only need to consider single-valued machines, and diagonalizing against them is easy for a nondeterministic machine with a little more running time, but figuring out which nondeterministic machines are single-valued at a given input length is hard.

2.2 Byproducts

In this section, we develop the intuition for our byproducts.

Targeted hitting-set generators from derandomization (Theorem 7). To obtain a targeted HSG from derandomization of prAM, we employ our targeted hitting-set generator in a win-win argument. Either a complexity class separation holds, in which case a result of [14] guarantees the existence of a regular (oblivious) hitting-set generator that yields the derandomization result, or we get a strong complexity class collapse. The collapse allows us to bypass some of the difficulties in diagonalizing against prAM protocols on almost-all inputs (one of the reasons we require advice in the derandomization-to-hardness direction of our near-equivalence), thus allowing us to do so efficiently and uniformly, and then instantiate our targeted hitting-set generator construction.

Average-case derandomization (Theorem 8). Our average-case derandomization results under worst-case hardness assumptions also make use of our targeted hitting-set generator construction, but in a different way. They do not exploit the potential of the hitting sets to depend on the input x. In fact, they set $f(x)$ to the truth table of the worst-case hard language L from the hypothesis at an input length determined by $|x|$. Instead, they hinge on the strong resilient soundness properties of the reconstructor.

As we are considering the average-case derandomization setting, the problem of obtaining the distinguisher D for the reconstruction resurfaces. Our approach is similar to the one for the traditional average-case derandomization setting for BPP. If the simulation fails for protocol A with noticeable probability over a random input, then we can sample multiple inputs $x_1, x_2, \ldots$ and construct a list of "candidate distinguishers" $D_{x_1} \doteq A(x_1, \cdot), D_{x_2} \doteq A(x_2, \cdot), \ldots$ such that the list contains, with high probability, at least one "true" distinguisher. Whereas in the BPP setting one can test each candidate and discard, with high probability, the ones that are not distinguishers, we do not know how to do that in the AM setting. Instead, we employ a different approach: We run the reconstructor with each distinguisher with the hope that every execution either fails or outputs the correct value.

This approach necessitates a stronger form of resilience than the one provided by the RMV generator: That its reconstruction is sound when given as input *any* co-nondeterministic circuit D, not just those that accept at least half of their inputs (as in Property 10). We don't know how to guarantee this with our prAM reconstruction, but we are able to do so in $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$ by approximating the fraction of inputs that D accepts and outright failing if the fraction is too low.

We point out that earlier works [13, 25] also manage to guarantee soundness of the reconstructor for co-nondeterministic circuits D that accept at least half of their inputs, based on the resilient partial single-valuedness of the reconstructor for MV or RMV. They do so by running an instance checker, which limits the hard function f to classes for which instance checkers are known to exist, such as complete problems for E and EXP. Instead, we achieve soundness of the reconstructor based on the soundness of a PCP. As PCPs exist for all nondeterministic computations, this makes our approach more suitable in this setting. In particular, we do not know how to obtain Theorem 8 along the lines of [13, 25].

Unconditional mild derandomization (Theorem 9). Our unconditional mild derandomization result relies on a similar win-win argument as in the proof of Theorem 7: Either some hardness assumption/class separation holds, in which case we get derandomization right away, or we get a complexity collapse that we use to construct, by diagonalization, a hard function f that has the efficiency requirements we need to obtain the derandomization result using our targeted hitting-set generator.

Since our result is unconditional, we cannot use derandomization assumptions to make diagonalizing against prAM protocols easier. Instead, we rely on the inclusion $\mathsf{prAM} \subseteq \Pi_2\mathsf{P}$, which allows for diagonalizing against such protocols in $\Sigma_2\mathsf{TIME}[n^{\omega(1)}]$. Our generator, however, requires the hard function to be computable by efficient nondeterministic algorithms. To help bridge the gap, we prove an "easy witness lemma" for Σ_2 computations that guarantees a strong collapse in case the aforementioned hardness assumption does not hold. The collapse then allows us to instantiate our targeted hitting-set generator construction with the diagonalizing function.

3 Preliminaries

We assume familiarity with standard complexity classes such as NP, AM, and prAM. We often consider inputs and outputs from non-Boolean domains, such as $\mathbb{F}^r$ for a field $\mathbb{F}$ and $r \in \mathbb{N}$. In such cases, we implicitly assume an efficient binary encoding for the elements of these domains. Finally, as is customary, all time bounds considered are implicitly assumed to be time-constructible.

3.1 Nondeterministic, co-nondeterministic and single-valued computation

We make use of nondeterministic, co-nondeterministic, and single-valued circuits in our results. A nondeterministic circuit is a Boolean circuit C with two sets of inputs, x and y. We say that C accepts x if there exists some y such that $C(x, y) = 1$, and that C rejects x otherwise. A co-nondeterministic circuit has a symmetric acceptance criterion: It accepts x if for all y it holds that $C(x, y) = 1$, and rejects x otherwise. A partial single-valued circuit also has two inputs, x and y; on input (x, y) it either fails (which we represent by $C(x, y) = \perp$) or succeeds and outputs a bit $b = C(x, y)$. Moreover, we require that for all y, y' such that both $C(x, y)$ and $C(x, y')$ succeed, $C(x, y) = C(x, y')$, i.e., the circuit computes a partial function on its first input. If, furthermore, for all x there exists a y such that $C(x, y)$ succeeds, we call the circuit total single-valued or just single-valued.

We are also interested in nondeterministic algorithms that compute multi-bit functions $f : \{0,1\}^* \to \{0,1\}^*$. Let $T(n)$ be a time bound. We say that $f \in \mathsf{NTIME}[T(n)]$ if there exists a nondeterministic algorithm N running in time $O(T(n))$ such that for all $x \in \{0,1\}^*$, there exists at least one computation path on which $N(x)$ succeeds, and $N(x)$ outputs $f(x)$ on all successful computation paths. Note, in particular, that if $f \in \mathsf{NTIME}[T(n)]$, then the language $L_f = \{(x, i, b) \mid f(x)_i = b\}$ is in $\mathsf{NTIME}[T(n)]$.

3.2 Arthur-Merlin protocols

A promise Arthur-Merlin protocol P is a computational process in which Arthur and Merlin receive a common input x and operate as follows in alternate rounds for a bounded number of rounds. Arthur samples a random string and sends it to Merlin. Merlin sends a string that depends on the input x and all prior communication from Arthur; the underlying function

is referred to as Merlin's strategy, which is computationally unrestricted. At the end of the process, a deterministic computation on the input x and all communication determines acceptance. The running time of the process is the running time of the final deterministic computation.

Any promise Arthur-Merlin protocol can be transformed into an equivalent one with just two rounds and Arthur going first, at the cost of a polynomial blow-up in running time, where the degree of the polynomial depends on the number of rounds [3]. As such, we often use the notation prAM to refer to promise Arthur-Merlin protocols with any bounded number of rounds, even though, strictly speaking, the notation refers to a two-round protocol with Arthur going first.

Promise Arthur-Merlin protocols can be simulated by probabilistic algorithms with oracle access to SAT: Instead of interacting with Merlin, Arthur asks the SAT oracle whether there exists a response of Merlin that would lead to acceptance. Similarly, $\mathsf{P}_{||}^{\mathsf{prAM}}$ can be simulated in $\mathsf{BPP}_{||}^{\mathsf{SAT}}$, the class of problems decidable by probabilistic polynomial-time algorithms with bounded error and non-adaptive oracle access to SAT. In fact, a converse also holds and helps to extend some of our results for prAM to the class $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$.

▶ **Lemma 11** ([5]). $\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \mathsf{P}_{||}^{\mathsf{prAM}}$.

In Lemma 11, the deterministic machines with oracle access to prAM on the right-hand side are guaranteed to work correctly irrespective of how the queries outside of the promise are answered, even if those queries are answered inconsistently, i.e., different answers may be given when the same query is made multiple times.

Arthur-Merlin protocols that output values. A promise Arthur-Merlin protocol P may also output a value. In this case, at the end of the interaction, the deterministic computation determines success/failure and, in case of success, an output value. We denote this value by $P(x, M)$, which is a random variable defined relative to a strategy M for Merlin. Similar to the setting of circuits, we indicate failure by setting $P(x, M) = \perp$, a symbol disjoint from the set of intended output values. Our choice of using success and failure for protocols that output values is to avoid confusion with the decisional notions of acceptance and rejection.

▶ **Definition 12** (Arthur-Merlin protocol with output). *Let P be a promise Arthur-Merlin protocol. We say that on a given input $x \in \{0,1\}^*$:*

- *P outputs v with completeness c if there exists a Merlin strategy such that the probability that P succeeds and outputs v is at least c. In symbols: $(\exists M)\Pr[P(x, M) = v] \geq c$.*
- *P outputs v with soundness s if, no matter what strategy Merlin uses, the probability that P succeeds and outputs a value other than v is at most s. In symbols: $(\forall M)\Pr[P(x, M) \notin \{v, \perp\}] \leq s$.*
- *P has partial single-valuedness s if there exists a value v such that P outputs v with soundness s. In symbols: $(\exists v)(\forall M)\Pr[P(x, M) \notin \{v, \perp\}] \leq s$.*

Note that if P on input x outputs v with completeness c and has partial single-valuedness s, then it outputs v with soundness s, provided $s > 1 - c$. If we omit c and s, then they take their default values of $c = 1$ (perfect completeness) and $s = 1/3$.

For a given function $f : X \to \{0,1\}^*$ where $X \subseteq \{0,1\}^*$, we say that P computes f with completeness $c(n)$ and soundness $s(n)$ if on every input $x \in X$, P outputs $f(x)$ with completeness $c(|x|)$ and soundness $s(|x|)$. Note that P may behave arbitrarily on inputs that are not in X. In contrast, an AM protocol computing f still computes some value in a complete and sound fashion on inputs $x \notin X$.

3.3 Learn-and-evaluate and commit-and-evaluate protocols

The reconstruction processes for hardness-based hitting-set generators for prAM are typically special types of promise Arthur-Merlin protocols. We distinguish between two types.

A *learn-and-evaluate protocol* is composed of two phases: A *learning* phase followed by an *evaluation* phase. In the learning phase, a probabilistic algorithm makes queries to a function f and produces an output (which we call a sketch). The evaluation phase then consists of a promise Arthur-Merlin protocol that computes $f(x)$ correctly on every input x when given the sketch as additional input.

▶ **Definition 13** (Learn-and-evaluate protocol). *A learn-and-evaluate protocol P consists of a probabilistic oracle algorithm A_{learn} and a promise Arthur-Merlin protocol P_{eval}. Let $f : X \to \{0,1\}^*$ where $X \subseteq \{0,1\}^*$. We say that P computes f with error $e(n)$ for completeness $c(n)$ and soundness $s(n)$ if on every input $x \in X$ of length n the following hold: The probability over the randomness of A_{learn} that P_{eval} with input x and additional input $\pi = A^f_{learn}(1^n)$ outputs $f(x)$ with completeness $c(n)$ and soundness $s(n)$ is at least $1 - e(n)$.*

The learning phase of a learn-and-evaluate protocol can be simulated by an Arthur-Merlin protocol with output, where Merlin guesses the queries that A_{learn} makes on a given random-bit string and answers them in parallel, and the output is a sketch of f. In this view, a learn-and-evaluate protocol becomes a pair of promise Arthur-Merlin protocols: one for the learning phase, and one for the evaluation phase. Note that the quality of the evaluation phase is only guaranteed when the learning queries are answered correctly, i.e., when Merlin is honest in the learning phase.

A *commit-and-evaluate* protocol [25] has the syntactic structure of a pair of promise Arthur-Merlin protocols without the restriction that Merlin in the first phase only answers queries about f. Semantically, a commit-and-evaluate protocol is more constrained than a learn-and-evaluate protocol. The first protocol of the pair now represents a commitment phase instead of a learning phase. In this phase, Arthur and Merlin interact and produce an output π, which we call a commitment. Similar to a learn-and-evaluate protocol, the commitment is given as input to the protocol of the evaluation phase. Whereas in a learn-and-evaluate protocol there are no guarantees whatsoever when Merlin is dishonest in the first phase, in a commit-and-evaluate protocol there is a strong guarantee: With high probability over Arthur's randomness in the commitment phase, the evaluation protocol is partial single-valued, meaning that Merlin cannot make Arthur output different values for the same input x with high probability. The guarantee is referred to as resilient partial single-valuedness.

▶ **Definition 14** (Commit-and-evaluate protocol). *A commit-and-evaluate protocol is a pair of promise Arthur-Merlin protocols $P = (P_{commit}, P_{eval})$. P has resilience $r(n)$ for partial single-valuedness $s(n)$ on domain $X \subseteq \{0,1\}^*$ if for all n, no matter what strategy Merlin uses during the commit phase, the probability that in the commitment phase, on input 1^n, P_{commit} succeeds and outputs a commitment π that fails to have the following property (2) is at most $r(n)$:*

For every x of length n in X, $P_{eval}(x, \pi)$ has partial single-valuedness $s(n)$. (2)

In symbols: $(\forall n)(\forall M_{commit})$

$$\Pr[(\forall x \in X \cap \{0,1\}^n) P_{eval}(x, \pi) \text{ has partial single-valuedness } s(n)] \geq 1 - r(n),$$

where $\pi = P_{commit}(1^n, M_{commit})$.

A commit-and-evaluate protocol naturally induces a promise Arthur-Merlin protocol: On input x, run P_{commit} on input $1^{|x|}$. If this process succeeds, let π denote its output and run P_{eval} on input (x, π).

3.4 Hitting-set generators and targeted hitting-set generators

In the setting of prBPP, Goldreich [11] discusses two equivalent definitions of targeted pseudo-random generators: one for deterministic linear-time machines that take both the input x and the random-bit string r as inputs, and one based on circuits D that only take the random-bit string r as input. The circuit D can be obtained by first constructing a circuit C that simulates the machine on inputs of length $|x|$, and then hardwiring the input x. The difference between a regular and targeted pseudo-random generator lies in the dependency of the output on x (in the first definition) or the circuit D (in the second definition): For a regular PRG the output can only depend on $|x|$ or the size of D, whereas for a targeted PRG it can depend on x and D proper.

In the setting of prAM, without loss of generality, we can assume that promise Arthur-Merlin protocols have perfect completeness. Therefore, we only need to consider targeted hitting-set generators, the variant of targeted PRGs for one-sided error. Similar to the BPP setting, there are two equivalent definitions of targeted HSGs for prAM. We propose a third, hybrid, and also equivalent definition, where the targeted generator is given access to both x and the circuit C. For prAM with perfect completeness the circuit C (as well as D) is co-nondeterministic. For regular HSGs, the output can only depend on the size of C. Our definition highlights that, in principle, there are two types of obliviousness that regular PRGs/HSGs exhibit: With respect to the input (where only dependencies on its size are allowed) and with respect to the algorithm being derandomized (where only dependencies on its running time are allowed). Since the algorithm description can be incorporated as part of the input, the dependency on C can be avoided. This is essentially why all three definitions are equivalent. In our targeted hitting-set generator constructions the dependency will only be through x and the size of C.

We start by defining hitting sets for co-nondeterministic circuits.

▶ **Definition 15** (Hitting set for co-nondeterministic circuits)**.** *Let D be a co-nondeterministic circuit of size m. A set S of strings of length m is a hitting set for D if there exists at least one $z \in S$ such that $D(z) = 1$ (where D might take a prefix of z as input if necessary). In that case, we say that S hits D.*

The notion allows us to define targeted hitting-set generators for prAM as follows, where we assume, without loss of generality, perfect completeness and soundness $1/2$. Regular hitting-set generators are viewed as a special case.

▶ **Definition 16** (Regular and targeted hitting-set generator for prAM)**.** *A targeted hitting-set generator for* prAM *is a nondeterministic algorithm that, on input $x \in \{0,1\}^*$ and a co-nondeterministic circuit C, has at least one successful computation path, and if $\Pr_r[C(x,r) = 1] \geq 1/2$, outputs a hitting set for $D(r) \doteq C(x,r)$ on every successful computation path. A regular hitting-set generator for* prAM *is a targeted hitting-set generator where the output only depends on the size of C.*

For completeness, we state the standard way of obtaining the co-nondeterministic circuits C and D capturing promise Arthur-Merlin protocols.

▶ **Proposition 17.** *There exists an algorithm that, on input* 1^n *and the description of a (Boolean output, two-round)* prAMTIME$[t(n)]$ *protocol* P*, runs in time* $O(t(n)^2)$ *and outputs a co-nondeterministic circuit* C *of size* $m = O(t(n)^2)$ *that simulates and negates the computation of* P *for input length* n*, i.e., the input of* C *is comprised of* $x \in \{0,1\}^n$ *and Arthur's random-bit string* r*, and it co-nondeterministically verifies that there is no Merlin message that would lead to acceptance. In particular:*

- *If* P *with input* x *accepts all random inputs, then* $D_x(r) \doteq C(x,r)$ *rejects every input.*
- *If* P *with input* x *rejects at least a fraction* $1/2$ *of its random-bit strings, then* $D_x(r) \doteq C(x,r)$ *accepts at least a fraction* $1/2$ *of its inputs.*

3.5 PCPs and low-degree extensions

We use the following construction that follows from the PCP of proximity of Ben-Sasson, Goldreich, Harsha, Sudan, and Vadhan [4].

▶ **Lemma 18** ([4])**.** *Let* T *be a time bound. For every* $s = s(n) : \mathbb{N} \to (0,1]$ *and every language* $L \in$ NTIME$[T(n)]$ *there exists a PCP verifier* V *with perfect completeness, soundness* s*, randomness complexity* $\log(1/s) \cdot (\log T(n) + O(\log\log T(n)))$*, non-adaptive query complexity* $\log(1/s) \cdot \mathsf{polylog}(T(n))$*, and verification time* $\log(1/s) \cdot \mathsf{poly}(n, \log T(n))$*. More precisely,*

- V *has oracle access to a proof of length* $T(n) \cdot \mathsf{polylog}(T(n))$*, uses* $\log(1/s) \cdot (\log T(n) + O(\log\log T(n)))$ *random bits in any execution, makes* $\log(1/s) \cdot \mathsf{polylog}(T(n))$ *non-adaptive queries to the proof and runs in time* $\log(1/s) \cdot \mathsf{poly}(n, \log T(n))$*.*
- *If* $x \in L$*,* $|x| = n$*, then there exists* y *of length* $T(n) \cdot \mathsf{polylog}(T(n))$ *such that* $\Pr[V^y(x) = 1] = 1$*.*
- *If* $x \notin L$*,* $|x| = n$*, then for all* y' *of length* $T(n) \cdot \mathsf{polylog}(T(n))$*,* $\Pr[V^{y'}(x) = 1] \leq s$*.*

We also need standard low-degree extensions. Let $g : \{0,1\}^\ell \to \{0,1\}$ be a function, $\mathbb{F} = \mathbb{F}_p$ be the field with p elements (for prime p) and h and r integers such that $h^r \geq 2^\ell$. The low-degree extension of g with respect to p, h, r is the unique r-variate polynomial $\hat{g} : \mathbb{F}^r \to \mathbb{F}$ with degree $h-1$ in each variable, for which $\hat{g}(\vec{v}) = g(y)$ for all $\vec{v} \in [h]^r$ representing a $y \in \{0,1\}^\ell$ and $\hat{g}(\vec{v}) = 0$ for the $\vec{v} \in [h]^r$ that do not represent a string y. The total degree of $\hat{g}$ is $\Delta = hr$ and $\hat{g}$ is computable in time $\mathsf{poly}(h^r, \log p, r)$ given oracle access to g.

3.6 Average-case simulation

The instance-wise nature of our technique allows us to conclude derandomization on average with respect to arbitrary distributions by assuming hardness with respect to that same distribution. The notion of average-case simulation that we use is the one where the simulation works correctly with high probability over inputs drawn from the distribution. We typically want good simulations to exist with respect to every efficiently sampleable distribution (where the simulation may depend on the distribution). This is usually referred to as the "heuristic" setting.

▶ **Definition 19** (Heuristic)**.** *Let* Π *be a promise-problem,* $\mu : \mathbb{N} \to [0,1)$*,* $\mathcal{C}$ *a complexity class and* $\mathbf{x} = \{\mathbf{x}_n\}_{n\in\mathbb{N}}$ *an ensemble of distributions where* $\mathbf{x}_n$ *is supported on* $\{0,1\}^n$ *and such that for all* n*, every* x *in the support of* $\mathbf{x}_n$ *satisfies the promise of* Π*. We write*

$$\Pi \in \mathsf{Heur}_{\mathbf{x},\mu}\mathcal{C}$$

if there exists a language $L \in \mathcal{C}$ such that for all sufficiently large n, $\Pr_{x \in \mathbf{x}_n}[L(x) \neq \Pi(x)] \leq \mu(n)$. We write

$$\Pi \in \mathsf{Heur}_{\mu}\mathcal{C}$$

if the above property holds for every polynomial-time sampleable ensemble of distributions with the above support restriction.

The notions of average-case simulation extend to the infinitely-often setting in the natural way.

4 Targeted hitting-set generator construction

In this section, we develop our targeted HSG construction, which leads to our instance-wise hardness vs. randomness tradeoffs for Arthur-Merlin protocols.

Our construction builds on the RMV generator due to Shaltiel and Umans [25], which is a recursive variant of the MV generator that shares the desired resilience property with MV. We start with the definition of the RMV generator in Section 4.1 and state its reconstruction properties in terms of a commit-and-evaluate protocol. We present our construction and analysis in Section 4.2 and the derandomization consequences in Section 4.3.

4.1 Recursive Miltersen-Vinodchandran generator

We need a couple of ingredients to describe how the RMV generator works. The first one is a local extractor for the Reed-Müller code. A local extractor is a randomness extractor that only needs to know a few bits of the sample. In the following definition the sample is provided as an oracle, and the structured domain from which the sample is drawn is given as an additional parameter.

▶ **Definition 20** (Local extractor). *Let S be a set. A (k, ϵ) local S-extractor is an oracle function $E : \{0,1\}^s \to \{0,1\}^t$ that is computable in time $\mathsf{poly}(s,t)$ and has the following property: For every random variable X distributed on S with min-entropy at least k, $E^X(U_s)$ is ϵ-close to uniform.*

We make use of the following local extractor for Reed-Müller codes.

▶ **Lemma 21** (Implicit in [24]). *Fix parameters $r < \Delta$, and let S be the set of polynomials $\hat{g} : \mathbb{F}^r \to \mathbb{F}$ having total degree at most Δ, where $\mathbb{F} = \mathbb{F}_p$ denotes the field with p elements. There is a $(k, 1/k)$ local S-extractor for $k = \Delta^5$ with seed length $s = O(r \log p)$ and output length $t = \Delta$.*

Note that for every subcube with sides of size $\frac{\Delta}{r}$ and choice of values at its points, there exists an interpolating polynomial $\hat{g}$ with the parameters of Lemma 21. It takes $(\Delta/r)^r \log p$ bits to describe these polynomials, but the local extractor only accesses $\mathsf{poly}(\Delta, r, \log p)$ bits.

When instantiated with a polynomial $\hat{g} : \mathbb{F}^r \to \mathbb{F}$, the RMV generator groups variables and operates over axis-parallel (combinatorial) lines over the grouped variables.[4] Shaltiel and Umans call these MV lines, which we define next.

[4] In the original construction [25], the RMV generator is defined with the number d of groups of variables as an additional parameter. Eventually, d is set to 2, which is the value we use for our results as well.

▶ **Definition 22** (MV line). *Let $\mathbb{F} = \mathbb{F}_p$ for a prime p. Given a function $\hat{g} : \mathbb{F}^r \to \mathbb{F}$ where r is an even integer, we define $B = \mathbb{F}^{r/2}$ and identify $\hat{g}$ with a function from B^2 to $\mathbb{F}$. Given a point $\vec{a} = (\vec{a}_1, \vec{a}_2) \in B^2$ and $i \in \{1, 2\}$, we define the line passing through $\vec{a}$ in direction i to be the function $L : B \to B^2$ given by $L(\vec{z}) = (\vec{z}, \vec{a}_2)$ if $i = 1$ and $L(\vec{z}) = (\vec{a}_1, \vec{z})$ if $i = 2$. This is an axis-parallel, combinatorial line, and we call it an MV line. Given a function $\hat{g} : \mathbb{F}^r \to \mathbb{F}$ and an MV line L we define the function $\hat{g}_L : B \to \mathbb{F}$ by $\hat{g}_L(z) = \hat{g}(L(z))$.*

The input for the RMV construction is a multivariate polynomial $\hat{g} : \mathbb{F}^r \to \mathbb{F}$ of total degree at most Δ, and the output is a set of m-bit strings for $m \leq \Delta^{1/100}$. The construction is recursive and requires that r is a power of 2 and that p is a prime larger than Δ^{100} (say, between Δ^{100} and $2\Delta^{100}$). Let E be the $(k, 1/k)$-local extractor from Lemma 21 for polynomials of degree Δ in $(r/2)$ variables over $\mathbb{F}$. Remember that $k = \Delta^5$ and that the extractor uses seed length $O(r \log p)$ and output length $t = \Delta \geq m$. By using only a prefix of the output, we have it output exactly m bits.

The operation of the RMV generator on input $\hat{g}$ is as follows: Set $B = \mathbb{F}^{r/2}$. For every $\vec{a} \in B^2$ and $i \in \{1, 2\}$, let $L : B \to B^2$ be the MV line passing through $\vec{a}$ in direction i. Compute $E^{\hat{g}_L}(y)$ for all seeds y. For $r = 2$, output the set of all strings of length m obtained over all $\vec{a} \in B^2$, MV lines L through $\vec{a}$, and seeds y. For $r > 2$, output the union of this set and the sets output by the recursive calls $\mathrm{RMV}(\hat{g}_L)$ for each of the aforementioned MV line L.

The construction runs in time $p^{O(r)}$ and therefore outputs at most that many strings. If the set output by the procedure fails as a hitting set for a co-nondeterministic circuit D of size m, then there exists an efficient commit-and-evaluate protocol P for $\hat{g}$ with additional input D. This is the main technical result of [25], which we present in a format that is suitable for obtaining our results.[5]

▶ **Lemma 23** ([25]). *Let Δ, m, r, p be such that $m \leq \Delta^{1/100}$, r is a power of 2 and p is a prime between Δ^{100} and $2\Delta^{100}$. Let also $\mathbb{F} = \mathbb{F}_p$ and $s \in (0, 1]$. There exists a commit-and-evaluate protocol $P = (P_{commit}, P_{eval})$ with additional input D, where D is a co-nondeterministic circuit of size m, such that the following holds for any polynomial $\hat{g} : \mathbb{F}^r \to \mathbb{F}$ of total degree at most Δ.*

- Completeness: *If D rejects every element output by $\mathrm{RMV}(\hat{g})$ then there exists a strategy M_{commit} for Merlin in the commit phase such that P_{eval} on input (z, D, π) outputs $\hat{g}(z)$ with completeness 1 for every $z \in \mathbb{F}^r$, where $\pi \doteq P_{commit}(1^n, M_{commit})$.*
- Resilience: *If D accepts at least a fraction $1/2$ of its inputs then P has resilience s for partial single-valuedness s on domain $\mathbb{F}^r$.*
- Efficiency: *Both P_{commit} and P_{eval} have two rounds. P_{commit} runs in time $\log(1/s) \cdot \mathsf{poly}(\Delta, r)$ and P_{eval} runs in time $(\log(1/s))^2 \cdot \Delta^{O((\log r)^2)}$.*

P only needs blackbox access to the deterministic predicate that underlies D.

4.2 Targeted generator and reconstruction

In this section, we present our targeted HSG construction, which works as follows: On input x and a co-nondeterministic circuit D of size m, it guesses a PCP (as in Lemma 18) for each bit of $f(x)$ and verifies each PCP deterministically by enumerating over the PCP verifier's

[5] Shaltiel and Umans present the evaluation protocol as a multi-round protocol (with $\log r$ rounds). We collapse it into a two-round protocol by standard amplification (which also amplifies the crucial resilience property) [3, 25].

randomness. It encodes each PCP as a low-degree polynomial (as in Section 3.5), instantiates the RMV generator with each of the polynomials and outputs the union of the outputs for each instantiation. For the reconstruction, we have Merlin send a bit b and commit to the low-degree extension of a proof that the i-th bit of $f(x)$ equals b. Arthur then runs the PCP verifier using the evaluation protocol to answer proof queries. The protocol succeeds and outputs b if and only if the PCP verifier accepts. Here is the formal statement of the result.

▶ **Theorem 24.** *Let $T(n)$ be a time bound and $f \in \mathsf{NTIME}[T(n)]$. There exists a nondeterministic algorithm H (the generator) that always has at least one successful computation path per input, and a promise Arthur-Merlin protocol R (the reconstructor) such that for every $x \in \{0,1\}^*$ and every co-nondeterministic circuit D that accepts at least half of its inputs, at least one of the following holds.*

1. *$H(x, D)$ outputs a hitting set for D on every successful computation path.*
2. *$R(x, D)$ computes $f(x)$ with completeness 1 and soundness 1/3.*

The construction also has the following properties:

- Resilient soundness: *In either case, the probability that $R(x, D)$ outputs a value other than $f(x)$ is at most $1/3$.*
- Efficiency: *On inputs x of length n and D of size m, H runs in time $\mathsf{poly}(T(n), m)$, and R, given an additional index i, computes the i-th bit of $f(x)$ in time $\mathsf{poly}(n) \cdot (m \cdot \log T(n))^{O((\log r)^2)}$ for $r = O(\log(T(n))/\log m)$.*

Moreover, $H(x, D)$ only depends on x and the size of D, and $R(x, D)$ only needs blackbox access to the deterministic predicate that underlies D.

Proof. Let $f \in \mathsf{NTIME}[T(n)]$, consider the language $L_f = \{(x, i, b) \mid f(x)_i = b\}$ and note that $L_f \in \mathsf{NTIME}[T(n)]$. Let V be the PCP verifier of Lemma 18 for L_f with soundness $s = s(n) = (100T(n)^{-1})$. Let also $h = h(m) = m^{100}$, $r = r(n, m)$ be the smallest power of 2 such that h^r is greater than the proof length of V on input length n and $p = p(n, m)$ be the smallest prime in the interval $[\Delta^{100}, 2\Delta^{100}]$ for $\Delta = h \cdot r$. Note, in particular, that $h^r = \mathsf{poly}(T(n), m)$ and $r = O(\log(T(n))/\log m)$.

Generator. The generator H, on input x and a co-nondeterministic circuit D of size m, first guesses the value of $z = f(x)$ and a proof y_i of the correct length $T(n) \cdot \mathsf{polylog}(T(n))$ for the i-th bit of z for each i. Then it verifies that $\Pr[V^{y_i}(x, i, z_i) = 1] = 1$ for all i by deterministically enumerating over the $\mathsf{poly}(T(n))$ random-bit strings for V. If any of the verifications fail, it fails. Otherwise, it views each y_i as a function $g_i : \{0,1\}^\ell \to \{0,1\}$ for $\ell = \log |y_i|$ and outputs $\mathrm{RMV}(\hat{g}_i)$, where $\hat{g}_i$ is the low-degree extension of g_i with parameters p, h and r. The initial verification step takes time $\mathsf{poly}(T(n))$, and executing $\mathrm{RMV}(\hat{g})$ takes time $p^{O(r)} = \mathsf{poly}(T(n), m)$ and outputs strings of length m. This culminates in a running time of $\mathsf{poly}(T(n), m)$. Finally, since for the correct output $z = f(x)$ there always exist proofs y_i that are accepted with probability 1 for each i, there always exists a nondeterministic guess that leads the generator to succeed.

Reconstructor. We describe and analyze the prAM protocol R, which uses the commit-and-evaluate protocol $P = (P_{\mathrm{commit}}, P_{\mathrm{eval}})$ of Lemma 23 with soundness parameter $s' = s'(n) = (100T(n) \cdot q)^{-1}$, where $q = q(n) = \mathsf{polylog}(T(n))$ denotes the query complexity of V at input length n. On inputs x, D and an index i, Arthur and Merlin play the commit phase P_{commit}, which produces a commitment π_i to be fed into the evaluation phase. In parallel, Merlin also sends a bit b to Arthur. The idea is for an honest Merlin to send $b = f(x)_i$ and commit to

the low-degree extension $\hat{g}_i$ of a proof y_i that witnesses $(x, i, b) \in L_f$ (or $f(x)_i = b$), though a dishonest Merlin may send a different bit and/or commit to some different function. Let γ_i denote the function that Merlin committed to via P_{commit}, which may be accessed with high probability by executing the evaluation protocol P_{eval} with input π_i. The restriction of γ_i to $[h]^r$ defines a candidate PCP proof $\tilde{y}_i$. Arthur then runs the verifier $V^{\tilde{y}_i}(x, i, b)$, employing Merlin's help to evaluate $\tilde{y}_i$ whenever V makes a query to it (where binary queries are first converted into the respective $\vec{v} \in \mathbb{F}_p^r$ and all queries are evaluated in parallel). If $V^{\tilde{y}_i}(x, i, b)$ accepts, then R succeeds and outputs b, otherwise it fails.

Completeness. If D is not hit by $H(x, D)$, then for all indices i there exists at least one proof y_i that witnesses $(x, i, f(x)_i) \in L_f$ and such that $\text{RMV}(\hat{g}_i)$ fails to hit D, where $\hat{g}_i$ is the low-degree extension of y_i with parameters p, h and r. In that case, an honest Merlin can commit to such a $\hat{g}_i$ with probability 1 by the completeness property of Lemma 23 as well as send the correct value of $f(x)_i$ during the first phase. Then perfect completeness of V and P_{eval} guarantee that R succeeds and outputs $f(x)_i$ with probability 1.

(Resilient) soundness. If D accepts at least half of its inputs, then for a fixed index i the resilience property of P in Lemma 23 guarantees that with probability at least $1 - s'$, the commit phase is successful and thus the evaluation protocol with input π_i has partial single-valuedness s'. In that case, by a union bound over the at most q queries that V makes, with probability at least $1 - (100T(n))^{-1} = 1 - s$, every execution of the evaluation protocol results in the evaluation of a fixed function $\gamma_i : \mathbb{F}^r \to \mathbb{F}$. If Merlin sends the incorrect value of $b \neq f(x)_i$ in the first round (the only way he could try to have Arthur output the wrong value), the soundness property of V in Lemma 18 guarantees that R fails with probability at least $1 - s$ since $(x, i, b) \notin L_f$. By a union bound over these three "bad" events, all of which have probability at most s since $s \geq s'$, for any fixed index i, $R(x, D)$ with additional input i either fails or outputs $f(x)_i$ with probability at least $1 - 3s$. Finally, a union bound over the at most $T(n)$ possible indices i guarantees that R either fails or outputs $f(x)$ with soundness $1/3$. In particular, if completeness also holds then $R(x, D)$ computes $f(x)$ with completeness 1 and soundness $1/3$.

Efficiency. The commit phase takes time $\log(1/s') \cdot \mathsf{poly}(\Delta, r) = \mathsf{poly}(m, \log T(n))$ and two rounds of communication. Afterwards, evaluating each query made by $V(x, i, b)$ with P_{eval} takes time $(\log(1/s'))^2 \cdot \Delta^{O((\log r)^2)} = (m \cdot \mathsf{polylog}(T(n)))^{O((\log r)^2)}$. The verification step for V takes time $\log(1/s) \cdot \mathsf{poly}(n, \log T(n)) = \mathsf{poly}(n, \log T(n))$, and it makes at most $\log(1/s) \cdot \mathsf{polylog}(T(n)) = \mathsf{polylog}(T(n))$ queries, resulting in a total running time of $\mathsf{poly}(n) + (m \cdot \log T(n))^{O((\log r)^2)}$. Moreover, because V is non-adaptive, each execution of the evaluation protocol can be carried out in parallel, and thus the total number of rounds is four. Collapsing this protocol into a two-round one [3] leads to a prAM protocol with running time $\mathsf{poly}(n) \cdot (m \cdot \log T(n))^{O((\log r)^2)}$.

For the moreover part, we observe that computing $\text{RMV}(\hat{g}_i)$ for each i only requires knowledge of m, the size of circuit D (instead of the circuit itself) and thus the generator H also only requires knowledge of m. Similarly, the commit-and-evaluate protocol in Lemma 23 only requires blackbox access to the deterministic predicate that underlies the circuit D, and thus so does our reconstructor R since it just gives D as input to P. ◀

We remark that we can amplify the resilient soundness property for the reconstructor so that the probability that it outputs a value outside of $\{f(x)_i, \perp\}$ is at most 2^{-t} by running it $\Theta(t)$ times in parallel and outputting $\perp$ as soon as at least one of the answers is $\perp$ or the answers are inconsistent, and outputting the consistent answer bit otherwise.

We also present a version of the generator with a stronger resilient soundness property at the expense of increasing the complexity of the reconstructor from a promise Arthur-Merlin protocol to a probabilistic algorithm with parallel access to SAT. This version is useful for obtaining our byproducts in the average-case setting.

▶ **Corollary 25.** *Let $T(n)$ be a time bound and $f \in \mathsf{NTIME}[T(n)]$. There exists a nondeterministic algorithm H (the generator) and a probabilistic algorithm R (the reconstructor) with parallel access to* SAT *that have the same properties as in Theorem 24 but such that the resilient soundness property holds for* every *co-nondeterministic circuit.*

In the setting of Corollary 25, item 2 of Theorem 24 should be interpreted as saying that $R(x, D)$ outputs $f(x)$ with probability at least $2/3$. We refer to the stronger resilience property in Corollary 25 as *strong resilient soundness*.

The idea behind Corollary 25 is for the reconstructor to first check whether the co-nondeterministic circuit D accepts at least somewhat less than half of its inputs. This is where the parallel access to an oracle for SAT comes in; it allows us to distinguish with high probability between the cases where the fraction of accepted inputs is, say, at most $1/3$ and at least $1/2$. In the former case, the new reconstructor indicates failure with high probability. Otherwise, we boost the fraction of accepted inputs to at least $1/2$ by trying D on two independent inputs, and then run the old reconstructor on the corresponding co-nondeterministic circuit D'.

Proof of Corollary 25. Let H' be the generator and R' the reconstructor of Theorem 24 instantiated with function f and amplified to have (resilient) soundness $1/6$.

Generator. The generator H, on input x and D of size m, first constructs the circuit D' of size $2m$ as $D'(r_1 r_2) = D(r_1) \vee D(r_2)$. We then define $H(x, D)$ as $\mathrm{Left}(H'(x, D')) \cup \mathrm{Right}(H'(x, D'))$, where $\mathrm{Left}(S)$ and $\mathrm{Right}(S)$ output the set of the left and right halves of every string in S, respectively.

Reconstructor. On input (x, D) and an index i, the reconstructor R estimates up to error $1/12$ and with probability of failure $1/6$ the fraction of inputs accepted by D by evaluating circuit D on $O(1)$ random inputs of length m, which can be done in probabilistic time $\mathsf{poly}(m)$ with $O(1)$ parallel queries to a SAT oracle. If the estimated fraction is less than $5/12$ (the midpoint between $1/3$ and $1/2$), then R declares failure. In parallel, R builds the circuit D' in the same way as H, samples Arthur's randomness for protocol R' with inputs (x, D') and i and makes three queries to the SAT oracle to obtain the protocol's output: Whether there is a Merlin response that leads to success and whether there are Merlin responses that lead to outputting 0 and 1. If the first query is answered negatively, or the last two queries give inconsistent answers, then R declares failure. Otherwise, R outputs whatever R' does.

Strong resilient soundness. Consider two cases in relation to circuit D: Either D accepts fewer than $1/3$ of its inputs, or it accepts at least a $1/3$ of its inputs. In the first case, the initial verification fails with probability at least $5/6$. In the second case, D' accepts at least $2/3 - 1/9 = 5/9 > 1/2$ of its inputs. The resilient soundness property of protocol R' guarantees that with probability at least $5/6$, R either fails or outputs $f(x)$ correctly. In either case, it follows that R outputs an incorrect value for $f(x)$ with probability at most $1/6 \leq 2/3$.

Correctness. If a co-nondeterministic circuit D accepts at least half of its inputs, so does the circuit D'. Moreover, if $H(x, D)$ fails to hit D, then $H'(x, D')$ fails to hit D'. The correctness of protocol R' then guarantees that there exists a strategy for Merlin that makes R' output $f(x)$ with probability 1, and no strategy can make R' output an incorrect value for $f(x)$ with probability at least $1/6$. It follows that the second parallel phase of R yields $f(x)$ with probability at least $5/6$. Accounting for the error probability of $1/6$ in the initial verification, we conclude that R outputs $f(x)$ with probability at least $2/3$.

Efficiency. The running time of H is asymptotically identical to that of H', and the running time of R is polynomial in the running time of R'.

Finally, the moreover part follows right away from the moreover part of Theorem 24. ◀

Similar to the case of Theorem 24, we can amplify the strong resilient soundness property for the reconstructor so that the probability that it outputs a value outside of $\{f(x)_i, \perp\}$ (or different from $f(x)_i$ in case D is not hit by the generator) is at most 2^{-t} by running it $\Theta(t)$ times in parallel and outputting the majority answer.

4.3 Derandomization consequences

First, we present a generic derandomization result for prAM that works under hardness against arbitrary distributions.

▶ **Theorem 26.** *There exists a constant c such that the following holds. Let $t, T : \mathbb{N} \to \mathbb{N}$ be time bounds such that $t(n) \geq n$, $\Pi \in \mathsf{prAMTIME}[t(n)]$ and $\{\mathbf{x}_n\}_{n\in\mathbb{N}}$ be an ensemble of distributions such that $\mathbf{x}_n$ is supported over $\{0,1\}^n$ and such that for all n, every x in the support of $\mathbf{x}_n$ satisfies the promise of Π. Assume that for $\mu : \mathbb{N} \to [0, 1)$ there exists a length-preserving function $f \in \mathsf{NTIME}[T(n)]$ such that for every $\mathsf{prAMTIME}[t(n)^{O((\log r)^2)}]$ protocol P for $r = O(\log(T(n))/\log(t(n)))$, it holds that the probability over $x \sim \mathbf{x}_n$ that $P(x) = f(x)$ is at most $\mu(n)$ for all but finitely many n. Then, it holds that*

$$\Pi \in \mathsf{Heur}_{\mathbf{x},\mu}\mathsf{NTIME}[T(n)^c].$$

Proof. First, notice that if $t(n) \leq \log T(n)$, then the conclusion is trivial and if $t(n) \geq T(n)$ then the premise is impossible, so we focus on the case that $\log T(n) \leq t(n) \leq T(n)$. Let $\Pi \in \mathsf{prAMTIME}[t(n)]$ and let P be a two-round protocol for Π running in time $O(t(n))$ on inputs of length n. On input $x \in \{0,1\}^n$, compute the circuit D_x of Proposition 17 with protocol P, and note that D_x has size $O(t(n)^2)$. Then, instantiate the HSG of Theorem 24 with f. Feed H inputs x and D_x and run the usual derandomization procedure for protocol P with the set output by $H(x, D_x)$: For each string $\rho \in H(x, D_x)$, nondeterministically guess Merlin's message y_ρ and compute the output of P with randomness ρ and message y_ρ, accepting if and only if P accepts for every $\rho \in H(x, D_x)$. The entire procedure runs in nondeterministic time $\mathsf{poly}(T(n), t(n)) = O(T(n)^c)$ for some constant c, since $T(n) \geq t(n)$.

Assume, with the intent of deriving a contradiction, that with probability at least $\mu(n)$ over $x \sim \mathbf{x}_n$, this derandomization fails for input x. First, notice that by the perfect completeness of P it must be the case that such an x lies in Π_N and that P with input x accepts every string in $H(x, D_x)$. Therefore, D_x acts as a distinguisher for $H(x, D_x)$, i.e., it rejects every string output by D_x while accepting at least half of its inputs. By computing D_x and feeding it to the prAM protocol R of Theorem 24, we obtain a prAM protocol that computes individual bits of $f(x)$ correctly for every x for which the derandomization fails, i.e., with probability at least $\mu(n)$ over $x \sim \mathbf{x}_n$. By running this protocol n times in parallel to compute every bit of $f(x)$, we obtain a prAM protocol that runs in time

$$\mathsf{poly}(n) \cdot (t(n) \cdot \log T(n))^{O((\log r)^2)} = t(n)^{O((\log r)^2)}$$

since $t(n) \geq \log T(n)$ and $t(n) \geq n$. This is a contradiction to the hardness of f so we are done. ◀

We remark that we require hardness not just against AM protocols but against prAM protocols, which may not respect the completeness and/or soundness conditions on some inputs. However, an input of length n only contributes to the success fraction $\mu(n)$ provided the completeness and soundness conditions are met on that input.

As a consequence of Theorem 26, if the hardness assumption holds for almost-all inputs, then we obtain full derandomization of prAM.

▶ **Theorem 27.** *There exists a constant c such that the following holds. Let $t, T : \mathbb{N} \to \mathbb{N}$ be time bounds such that $t(n) \geq n$. If there is a length-preserving function $f \in \mathsf{NTIME}[T(n)]$ that is hard on almost-all inputs against $\mathsf{prAMTIME}[t(n)^{O((\log r)^2)}]$ for $r = O(\log(T(n))/\log(t(n)))$ then*

$$\mathsf{prAMTIME}[t(n)] \subseteq \mathsf{NTIME}[T(n)^c].$$

Moreover, there exists a targeted hitting-set generator that achieves this derandomization result.

Proof. The statement follows from Theorem 26 by noting that the assumption that f is hard on almost-all inputs implies that f is hard for all possible distributions $\mathbf{x}_n$ with success probability $\mu(n) = 0$. In particular, the following nondeterministic algorithm is a hitting-set generator for prAM: On input $x \in \{0,1\}^*$ and a co-nondeterministic circuit C of size m, output $H(x, D)$ where H is the generator of Theorem 24 and $D \doteq C(x, \cdot)$. This algorithm has a successful computation path for any input and, on every successful computation path on inputs where D accepts at least half of its inputs, it outputs a set that hits D. The running time of the generator is $\mathsf{poly}(T(n), m)$. ◀

Table 2 Derandomization consequences that follow from different instantiations of Theorem 27.

Setting	$T(n)$	Hard for	Derandomization
high end	n^a	$n^{O((\log a)^2)}$	$\mathsf{prAM} \subseteq \mathsf{NP}$
middle-of-the-road	$2^{\mathsf{polylog}(n)}$	$n^{O((\log \log n)^2)}$	$\mathsf{prAM} \subseteq \mathsf{NTIME}[2^{\mathsf{polylog}(n)}]$
low end	$2^{n^{o(1)}}$	$n^{o((\log n)^2)}$	$\mathsf{prAM} \subseteq \mathsf{NTIME}[2^{n^{o(1)}}]$
very low end	$2^{\mathsf{poly}(n)}$	$n^{b(\log n)^2}$ $\forall b$	$\exists c$ $\mathsf{prAM} \subseteq \mathsf{NTIME}[2^{n^c}]$

By setting parameters in Theorem 27, we obtain the derandomization results listed on Table 2. In particular, the first line of Table 2 establishes Theorem 5 and the last line establishes Theorem 6. We now provide more details on how to obtain each line of Table 2:

- For the high end, set $t(n) = n$, in which case $r = O(a)$. Then, $\mathsf{prAMTIME}[n] \subseteq \mathsf{NP}$ follows as long as f is hard on almost-all inputs against $\mathsf{prAMTIME}[n^{O((\log a)^2)}]$. The result for prAM follows by padding.
- For the middle-of-the-road result, set $t(n) = n$, in which case $r = \mathsf{polylog}(n)$. Then, $\mathsf{prAMTIME}[n] \subseteq \mathsf{NTIME}[2^{\mathsf{polylog}(n)}]$ follows as long as f is hard on almost-all inputs against $\mathsf{prAMTIME}[n^{O((\log \log n)^2)}]$. The result for prAM follows by padding.

- For the low end, let $\nu = \nu(n) = o(1)$ be such that $T(n) = 2^{n^\nu}$ and set $t(n) = n$. In this case, $r \leq n^\nu$. Then, $\mathsf{prAMTIME}[n] \subseteq \mathsf{NTIME}[\mathsf{poly}(n, 2^{n^\nu})]$ follows as long as f is hard on almost-all inputs against $\mathsf{prAMTIME}[n^{O((\nu \log n)^2)}]$. Since $\mathsf{poly}(n, 2^{n^\nu}) = 2^{n^{o(1)}}$ and $n^{O((\nu \log n)^2)} = n^{o((\log n)^2)}$, the result for prAM follows by padding.
- For the very low end, set $t(n) = n^b$ for a constant b, in which case $r = \mathsf{poly}(n)$. Then, $\mathsf{prAMTIME}[n^b] \subseteq \mathsf{NTIME}[2^{n^c}]$ for some constant c follows as long as f is hard on almost-all inputs against $\mathsf{prAMTIME}[n^{O(b(\log n)^2)}]$. To get the result for prAM, it suffices for hardness to hold for all b.

5 Consequences of derandomization

In this section, we prove the derandomization-to-hardness and derandomization-to-targeted HSGs directions of our near-equivalences.

5.1 Hardness on almost-all inputs

We start with our derandomization-to-hardness implication: If $\mathsf{prAM} \subseteq \mathsf{NP}$ then for all constants c there is a length-preserving function f computable in nondeterministic polynomial time (with a few bits of advice) that is hard on almost-all inputs against $\mathsf{AMTIME}[n^c]$. The basic idea is that, under the derandomization hypothesis, every (single-bit) AM protocol that runs in time n^c can be simulated by a single-valued nondeterministic machine without too much time overhead. If we have as advice whether a particular nondeterministic machine is single-valued or not at input length n, we can negate its input efficiently, obtaining a function f computable in nondeterministic time $\mathsf{poly}(n)$ that is almost-all inputs hard against AM protocols that run in time n^c. We now state Proposition 4 formally.

▶ **Proposition 28** (Formal version of Proposition 4). *If* $\mathsf{prAM} \subseteq \mathsf{NP}$*, then for every constant* c *and increasing function* $\alpha : \mathbb{N} \to \mathbb{N}$ *there exists a length-preserving function* $f \in \mathsf{NP}/\alpha(n)$ *that is hard on almost-all inputs against* $\mathsf{AMTIME}[n^c]$.

Proof. Assume that $\mathsf{prAM} \subseteq \mathsf{NP}$ and let c' be a constant to be defined later (which depends on c). The basic idea for the function f is as follows: On an input x of length n, we set its i-th output bit (for $1 \leq i \leq \min(n, \alpha(n))$) to the opposite of the i-th bit output by the i-th nondeterministic Turing machine N_i on input x (if N_i is single-valued and halts in at most $n^{c'+2}$ steps at input length n), and otherwise we set it to 0. Formally, on input x of length n and for $1 \leq i \leq n$

$$f(x)_i = \begin{cases} 1 - N_i(x)_i & \text{if } i \leq \alpha(n),\ N_i \text{ is single-valued and halts in at most } n^{c'+2} \text{ steps,} \\ 0 & \text{otherwise.} \end{cases}$$

Note that f is computable by a single-valued nondeterministic machine running in time $O(n^{c'+3})$ with $\alpha(n)$ bits of advice (indicating whether N_i is single-valued and halts in at most $n^{c'+2}$ steps at input length n for $1 \leq i \leq \alpha(n)$).[6] This holds because, when N_i is single-valued, computing $1 - N_i(x)_i$ can be done by guessing a path on which N_i succeeds, which must result in the unique value $N_i(x)$, and then outputting the opposite of the i-th bit of that. Assume, with the intent of deriving a contradiction, that there exists an AM protocol P that runs in time $O(n^c)$ and computes f on an infinite set of inputs $X \subseteq \{0,1\}^*$.

[6] The nondeterministic machine computing f is only guaranteed to be single-valued when given the correct advice string.

Consider the protocol P' that takes as regular input a triple (x, i, b) and accepts iff the i-th bit of the output of protocol P with input x equals b (if $i > |x|$ then P' rejects). Note that P' induces a language L in $\mathsf{AMTIME}[n^c]$. Since $\mathsf{prAM} \subseteq \mathsf{NP}$ and $\mathsf{prAMTIME}[n^c]$ has a complete problem under linear-time reductions, it follows that there exists a constant c' such that $\mathsf{AMTIME}[n^c] \subseteq \mathsf{NTIME}[n^{c'}]$.[7] Let N be a nondeterministic machine that runs in time $n^{c'}$ and computes L. Note that for every $x \in \{0,1\}^*$ and $1 \leq i \leq |x|$, $N(x, i, b) = 1$ for exactly one $b \in \{0, 1\}$, and when $x \in X$, $N(x, i, b) = 1$ if and only if $f(x)_i = b$.

Now consider the following procedure N': On input $x \in \{0,1\}^n$, guess a value b_i and a witness y_i for each $1 \leq i \leq n$ and run $N(x, i, b_i; y_i)$. If for all i, $N(x, i, b_i; y_i)$ accepts, N' succeeds and prints the concatenation of the guessed b_i's, otherwise N' fails. Note that N' is a nondeterministic machine that runs in time $O(n^{c'+1})$. Moreover, by our assumption that P is an AM protocol and that $\mathsf{prAM} \subseteq \mathsf{NP}$, N' is single-valued on *every* input. By construction, the single value equals $f(x)$ for all $x \in X$.

Let i be the index of N' in our enumeration, i.e., $N_i = N'$. By definition of f, for every input $x \in \{0,1\}^*$ of sufficiently large length $n \geq \alpha^{-1}(i)$ (so that it has a chance to negate the output of N_i), and in particular for all sufficiently large $x \in X$, we have that $f(x)_i = 1 - N'(x)_i = 1 - f(x)_i$, which is a contradiction. ◀

This result extends to other parameter settings. As an example, we state a version of Proposition 28 at the very low end.

▶ **Proposition 29.** *If there exists a constant c such that that $\mathsf{AM} \subseteq \mathsf{NTIME}[2^{n^c}]$, then for every increasing function $\alpha : \mathbb{N} \to \mathbb{N}$ there exists a function $f \in \mathsf{NEXP}/\alpha(n)$ that is hard on almost-all inputs against AM protocols running in polynomial time.*

Proof (Sketch). The proof is essentially identical to that of Proposition 28, but with a different time bound. Since $\mathsf{AM} \subseteq \mathsf{NTIME}[2^{n^c}]$, the diagonalizing machine N needs to diagonalize against single-valued nondeterministic algorithms running in time $2^{n^{c'}}$ for some fixed constant $c' > c$, and thus we get a nondeterministic algorithm that runs in time $O(2^{n^k})$ for any constant $k > c'$. ◀

We conclude this section by noting in more detail where the gaps between our hardness-to-derandomization and derandomization-to-hardness results lie. The first gap lies in the fact that in the derandomization-to-hardness direction, the hard function f we construct requires a few bits of advice that we don't know how to handle in the other direction. There is, however, a subtler difference – In the hardness-to-derandomization direction, we require hardness against prAM protocols, which may not obey the AM promise on all inputs (though we only consider the protocol as computing $f(x)$ on input x if it obeys the promise and respects both completeness and soundness on input x). In the derandomization-to-hardness direction, we can only guarantee hardness against AM protocols, which necessarily obey the AM promise on all inputs. We remark that a similar problem shows up in other hardness vs. randomness tradeoffs for AM [13, 25]. For example, to conclude almost-everywhere derandomization of AM, the authors of [13] require hardness of EXP against AM protocols for which completeness only holds infinitely-often. Finally, we also note that, while Chen and Tell only state their derandomization-to-hardness result for BPP [8], in that setting one can actually achieve hardness against prBPP (where the probabilistic algorithm might not have a high-probability output for every input).

[7] While our argument only requires that there exists a constant c' such that $\mathsf{AMTIME}[n^c] \subseteq \mathsf{NTIME}[n^{c'}]$, we use the assumption $\mathsf{prAM} \subseteq \mathsf{NP}$ instead of $\mathsf{AM} \subseteq \mathsf{NP}$ since it is unknown whether $\mathsf{AMTIME}[n^c]$ contains a complete problem under linear-time reductions.

5.2 Targeted hitting-set generator

In this section, we prove Theorem 7 along the lines of the intuition provided in Section 2.2. We make use of a win-win argument: Either the EXP $\neq$ NEXP hardness assumption holds, in which case there is a regular (oblivious) HSG that guarantees the derandomization result [14]. Or else we may assume that EXP = NEXP, which allows us to construct a function f that is hard against prAM protocols by diagonalization, with which we then instantiate Theorem 24 to obtain the targeted HSG.

We need the following result that follows from the "easy-witness" method.

▶ **Lemma 30** ([14]). *If* NEXP $\neq$ EXP *then* prAM $\subseteq$ io-NTIME$[2^{n^\epsilon}]/n^\epsilon$ *for every* $\epsilon > 0$. *Moreover, there exists a (regular) HSG that achieves this derandomization.*

We now prove Theorem 7, which we restate here for convenience.

▶ **Theorem 7.** *If* prAMTIME$[2^{\text{polylog}(n)}] \subseteq$ io-NEXP*, then there exists a targeted hitting-set generator for* prAM *that yields the simulation* prAM $\subseteq$ io-NTIME$[2^{n^c}]/n^\epsilon$ *for some constant* c *and all* $\epsilon > 0$.

Proof. If EXP $\neq$ NEXP, we are done by Lemma 30. Otherwise, it holds that NEXP = EXP. We use this collapse to construct a length-preserving multi-bit function $f \in$ EXP that is hard against prAMTIME$[n^{(\log n)^3}]$. We then instantiate Theorem 24 with f to obtain the targeted HSG. Hardness against protocols running in this time bound suffices along the lines of Theorem 6.

Before constructing f, we make an observation: Due to the instance-wise nature of our construction, to obtain an infinitely-often derandomization result using Theorem 24 it suffices to have an *infinitely-often all-inputs hardness assumption*. More precisely, we require the following: For every prAMTIME$[n^{(\log n)^3}]$ protocol P, there exist infinitely many input lengths n such that P fails to compute f for every x of length n. Thus, we construct a function f with this requirement in mind.

Under the hypothesized derandomization assumption and because prAMTIME$[n^{(\log n)^3}]$ has a complete problem under linear-time reductions, it follows that there exists a constant k such that prAMTIME$[n^{(\log n)^3}] \subseteq$ io-NTIME$[2^{n^k}]$. Since NTIME$[2^{n^k}]$ also has a complete problem under linear-time reductions, under the assumption EXP = NEXP, there exists a constant k' such that prAMTIME$[n^{(\log n)^3}] \subseteq$ io-DTIME$[2^{n^{k'}}]$. In that case, it suffices to diagonalize against fixed-exponential time machines to construct f. Similar to Proposition 28, we define the i-th bit of $f(x)$ to be the opposite of the i-th bit output by $M_i(x)$ when it runs for at most $2^{|x|^{k'+1}}$ steps, where M_i is the i-th deterministic Turing machine. Formally, on input x of length n and for $1 \leq i \leq n$,

$$f(x)_i = \begin{cases} 1 - M_i(x) & \text{if } M_i(x) \text{ halts in at most } 2^{n^{k'+1}} \text{ steps,} \\ 0 & \text{otherwise.} \end{cases}$$

Note that f is computable by a deterministic machine running in time $O(n \cdot 2^{n^{k'+1}})$ and thus $f \in$ EXP.

Assume, with the intent of deriving a contradiction, that there exists a prAMTIME$[n^{(\log n)^3}]$ protocol P such that for almost-all input lengths n, P computes f on at least one input $x \in \{0,1\}^n$, and call the set of inputs where P computes f correctly X. Again, similar to the proof of Proposition 28, P induces a problem Π in prAMTIME$[n^{(\log n)^3}]$, and by our assumptions, there is a language $L \in$ DTIME$[2^{n^{k'}}]$ such that L and Π agree on infinitely

many input lengths. Let M be a deterministic Turing machine running in time $O(2^{n^{k'}})$ that decides L. Recall that yes-instances of Π are triples (x, i, b) such that $x \in X$ and $f(x)_i = b$ while no-instances have $x \in X$ and $f(x)_i \neq b$. Let M' be the deterministic Turing machine that, on input x of length n, outputs $M'(x)$ of length n such that $M'(x)_i = 1$ if and only if M accepts $(x, i, 1)$ for $1 \leq i \leq n$. Note that M' runs in time $2^{n^{k'+1}}$. By construction and our assumption on P, for infinitely many input lengths n there exists at least one $x \in X \cap \{0,1\}^n$ such that $M'(x) = f(x)$.

Let i be the index of M' in our enumeration. By definition of f, for every input $x \in \{0,1\}^*$ of sufficiently large length $n \geq i$ (so that it has a chance to negate the output of M'), and in particular for all sufficiently large inputs $x \in X$, we have that $f(x)_i = 1 - M'(x)_i = 1 - f(x)_i$, a contradiction. Finally, we instantiate Theorem 26 with f to obtain a targeted HSG for prAM that runs in exponential time, which suffices to obtain the conclusion. ◀

6 Derandomization under uniform worst-case hardness

Our technique also leads to new results in the traditional uniform worst-case setting. Under worst-case hardness against probabilistic algorithms with non-adaptive oracle access to SAT, we obtain average-case derandomization results for prAM. Moreover, by further strengthening the hardness assumption, we may also conclude full (infinitely-often) derandomization of prAM. As previously mentioned, these results extend to average-case derandomization of $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$.

6.1 Average-case simulation

In this section, we develop our average-case derandomization results for prAM under worst-case uniform hardness assumptions (where hardness is against $\mathsf{BPTIME}_{||}^{\mathsf{SAT}}$). Our results in this setting work as follows: Assume there exists a hard language $L \in \mathsf{NTIME}[T(n)] \cap \mathsf{coNTIME}[T(n)]$. To derandomize some prAM protocol P on input length n, we first consider the hard language L at some suitable input length ℓ, which depends on the hardness of L (for Theorem 8, for example, we take $\ell = \Theta(\log n)$). Then we let f be the function that maps any input $x \in \{0,1\}^n$ to the truth table of L at input length ℓ, and it follows from the complexity of L that $f \in \mathsf{NTIME}[2^\ell \cdot T(\ell)]$. Finally, we instantiate our targeted HSG construction H with f and use it to derandomize P.

For the reconstruction, we make use of the strong resilient soundness property of Corollary 25. If the average-case derandomization fails, to decide whether z of length ℓ is in L, we first sample multiple candidate "good" strings x that hopefully lead to a distinguisher D_x for the generator (enough so that we expect at least one "good" x with high probability). Then, we run the reconstruction for all of them, accepting if and only if at least one of those outputs 1. By the strong resilient soundness property and amplification, with high probability every execution either fails or outputs $f(x)_z = L(z)$, and in the high probability case that we sample at least one "good" x, some execution outputs $L(z)$, meaning we can compute L efficiently on input length ℓ.

First, we present such a result at the high end of the derandomization spectrum.

▶ **Theorem 31** (Strengthening of Theorem 8). *If* $\mathsf{NTIME}[2^{an}] \cap \mathsf{coNTIME}[2^{an}]$ *is not included in* $\mathsf{BPTIME}[2^{(\log(a+1))^2 n}]_{||}^{\mathsf{SAT}}$ *for some constant* $a > 0$*, then for all* $e > 0$ *it holds that*

$$\mathsf{prAM} \subseteq \text{io-}\mathsf{Heur}_{1/n^e}\mathsf{NP}$$
$$\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \text{io-}\mathsf{Heur}_{1/n^e}\mathsf{P}_{||}^{\mathsf{SAT}}.$$

Proof. We first argue the result for prAM. Consider derandomizing a prAM protocol P for a problem Π running in time $O(n^k)$ for some constant k. Let S be an $O(n^s)$-time sampler for a distribution in $\{0,1\}^n$ and e be a constant such that we want to "fool" S with probability at least $1 - 1/n^e$. Let f be a function mapping every $x \in \{0,1\}^n$ to the truth table of the hard language $L \in \mathsf{NTIME}[2^{an}] \cap \mathsf{coNTIME}[2^{an}]$ at input length $\ell = \ell(n) = \Theta(\log n)$ to be set precisely later. Note that $f \in \mathsf{NTIME}[T(n)]$ for $T(n) = 2^{(a+1)\ell}$. Instantiate the generator H of Corollary 25 with f, run H on input $x = 0^n$ (recall f maps every string in $\{0,1\}^n$ to the same truth table) and co-nondeterministic circuit size $m = O(n^{2k})$, and use it to attempt to derandomize P in nondeterministic time $\mathsf{poly}(T(n), n^{2k}) = \mathsf{poly}(n)$.

If the derandomization fails for almost-all input lengths, even heuristically, then for almost-all input lengths n, $S(1^n)$ outputs with probability at least $1/n^e$ a string $x \in \{0,1\}^n$ such that the simulation errs on x, i.e., the circuit D_x obtained from x and P using Proposition 17 is a distinguisher for $H(0^n, D_x)$. To compute L at input length ℓ, it then suffices to do the following: On input $z \in \{0,1\}^\ell$, first use S to sample $t = \Theta(n^e)$ inputs $x_1, \ldots, x_t$ and use these to construct a list $D_{x_1}, \ldots, D_{x_t}$ of candidate distinguishers for $H(0^n, D_x)$. With high probability, this list contains an actual distinguisher for the generator. Let R be the algorithm of Corollary 25, amplified by parallel repetition to have negligible soundness 2^{-n}, i.e., with probability at least $1 - 2^n$, the algorithm outputs either $f(x)$ or $\perp$. Finally, run R with inputs 0^n, index z (recall $f(0^n)$ equals the truth table of L at input length ℓ) and D_{x_i} for every sampled input x_i, and accept if and only if some execution outputs 1. To see that this is correct, note that by a union bound, with high probability every execution of R is successful in the sense that it either outputs $f(0^n)_z = L(z)$ or $\perp$. Conditioned on there being a distinguisher in the list, we are guaranteed to output the correct value of $L(z)$ with high probability.

The running time for the reconstruction is $O(n^{e+s})$ for generating the $t = \Theta(n^e)$ samples, and $O(n^{2k})^{O((\log r)^2)}$ per sample for running R, where $r = O(((a+1)\ell)/(k \log n))$, for a total of $O(n^e(n^s + n^{O(k(\log r)^2)}))$. By setting $\ell = dk \log n$, we have that $r = O(d(a+1))$ and we can upper bound the total running time by $n^{O(e+s+k(\log(d(a+1)))^2)}$. In terms of the input length ℓ, this is $2^{(\log(a+1))^2 \ell}$ when d is a sufficiently large constant depending on a, e, s. This concludes the argument for prAM.

Now, we argue the result for $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$. To do so, we use the containment $\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \mathsf{P}_{||}^{\mathsf{prAM}}$ [5]. It suffices to show that every deterministic polynomial-time algorithm with non-adaptive oracle access to a paddable prAM-complete problem $\Gamma \in \mathsf{prAMTIME}[n]$ can be simulated by deterministic polynomial-time algorithms with non-adaptive oracle access to SAT. Let M be a deterministic algorithm with non-adaptive oracle access to Γ running in time $O(n^b)$ and S be an $O(n^s)$-time sampler that we want to "fool" with probability at least $1 - 1/n^e$. Since Γ is paddable, we may assume that every query made by M on inputs of length n is of length $O(n^b)$ (at the expense of increasing its running time to $O(n^{2b})$). To simulate M on input x, let f be a function mapping every $x \in \{0,1\}^n$ to the truth table of L at input length $\ell = \ell(n) = \Theta(\log n)$. As before, $f \in \mathsf{NTIME}[2^{(a+1)\ell}]$. Instantiate the generator H of Corollary 25 with f and use it to derandomize Γ at input length $O(n^b)$ in order to obtain a $\mathsf{P}_{||}^{\mathsf{SAT}}$ simulation for M. Whenever M with input x queries Γ, we instead query the SAT oracle whether the nondeterministic simulation of Γ using H with input 0^n and co-nondeterministic circuit size $m = O(n^{2b})$ accepts. This simulation runs in $\mathsf{P}_{||}^{\mathsf{SAT}}$ since M is non-adaptive.

If this derandomization fails on almost-all input lengths n, then as before we can use S to sample $t = \Theta(n^e)$ inputs $x_1, \ldots, x_t$ such that with high probability the simulation fails on some x_i. Let $Q(M, x)$ be the set of queries to Γ made by M on input x. If the simulation fails

on x_i, it must be the case that some query q in $Q(M, x_i)$ (and also in the promise of Γ) was answered incorrectly. Since the protocol for Γ has perfect completeness, it must be the case that $q \in \Pi_N$ and that D_q is a distinguisher for $H(0^n, D_q)$. The reconstruction is as before though we use the sets $Q(M, x_i)$ for $i \in [t]$ to obtain the list of candidate generators, and correctness follows by the same argument as in the prAM case. The running time analysis is similar to the one for the case of prAM. ◀

At the low end, we are able to obtain a slightly stronger average-case derandomization result. Instead of having a different simulation for each sampler, we obtain a single simulation (depending on the problem in $\mathsf{prAM}/\mathsf{prBPP}_{||}^{\mathsf{SAT}}$ and the constant ϵ) that "fools" every polynomial-time sampler.

▶ **Theorem 32.** *If* $\mathsf{NEXP} \cap \mathsf{coNEXP} \not\subseteq \mathsf{BPTIME}[n^{b((\log n)^2)}]_{||}^{\mathsf{SAT}}$ *for all* $b > 0$*, then for every* $\epsilon > 0$ *and all* $e > 0$

$$\mathsf{prAM} \subseteq \text{io-}\mathsf{Heur}_{1/n^e}\mathsf{NTIME}[2^{n^\epsilon}]$$
$$\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \text{io-}\mathsf{Heur}_{1/n^e}\mathsf{DTIME}[2^{n^\epsilon}]_{||}^{\mathsf{SAT}}.$$

Moreover, for any Π *in* prAM *or* $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$ *and* $\epsilon > 0$*, there is a single simulation that works for all* $e > 0$*.*

Proof. We begin with the argument for prAM. Let L be a hard language in $\mathsf{NTIME}[2^{n^a}] \cap \mathsf{coNTIME}[2^{n^a}]$ for some constant $a \geq 1$. Consider derandomizing a protocol P for a problem $\Pi \in \mathsf{prAMTIME}[n^k]$ for constant k. Let $\epsilon > 0$ and f be the function mapping every $x \in \{0,1\}^n$ to the truth table of L at input length $\ell = n^\epsilon$. Note that $f \in \mathsf{NTIME}[T(n)]$ for $T(n) = 2^{n^{a\epsilon}}$. Instantiate the generator H of Corollary 25 with f, run H on input $x = 0^n$ and co-nondeterministic circuit size $m = O(n^{2k})$, and use it to derandomize P. The simulation runs in nondeterministic time $\mathsf{poly}(T(n), n^{2k})$, which is at most $2^{n^{\epsilon'}}$ for any $\epsilon' > 0$ by taking a sufficiently small $\epsilon > 0$.

The reconstruction is identical to that of Theorem 31 but with $\ell = n^\epsilon$. The running time is $O(n^{e+s})$ to generate the samples and $(n^{2k})^{O((\log r)^2)}$ per sample for running R, where $r = O(\log(T(n))/\log n)$, for a total of $O(n^e(n^s + n^{O((\log r)^2)}))$. Given our parameter choices, $r = O(n^{a\epsilon})$, and the expression is upper bounded by $O(n^e(n^s + n^{O((a\epsilon \log n)^2)}))$. As the input length is $\ell = n^\epsilon$ for constant ϵ, there exists a constant b (depending on a, e, s, ϵ) such that the running time is upper bounded by $\ell^{b(\log n)^2}$. If hardness holds for all $b > 0$, then the same simulation works for any constant value of s and e, i.e., for any polynomial-time sampler and any inverse-polynomial error probability.

The proof for $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$ is also almost identical to that of Theorem 31, where we derandomize the "oracle" Γ using the generator H from Corollary 25 instantiated with the function f that maps every $x \in \{0,1\}^n$ to the truth table of L at input length $\ell = n^\epsilon$ and use a set of queries instead of a set of inputs to obtain the list of candidate distinguishers for the reconstruction. This approach naturally leads to a simulation in $\mathsf{P}_{||}^{\mathsf{NTIME}[2^{n^\epsilon}]}$, and we obtain the $\mathsf{DTIME}[2^{n^\epsilon}]_{||}^{\mathsf{SAT}}$ simulation by replacing the original queries with padded SAT queries. ◀

6.2 Infinitely-often all-input simulation

By introducing nondeterminism in the algorithms we require hardness for, we are able to extend Theorem 8 to conclude full (infinitely-often) derandomization of prAM. We have shown that, if the HSG construction of Theorem 8 fails to obtain average-case derandomization

of prAM, then we are able to efficiently sample candidate distinguishers with the hope that at least one is "good". However, if the HSG fails in the worst case, it is harder to pinpoint exactly where it does so as to obtain a distinguisher. To solve this, we have Merlin send a "good" input x. This necessitates a lower bound against $\mathsf{MATIME}^{\mathsf{SAT}}_{||}$, but allows for concluding full (infinitely-often) derandomization of prAM and $\mathsf{prBPP}^{\mathsf{SAT}}_{||}$.

▶ **Theorem 33.** *If* $\mathsf{NTIME}[2^{an}] \cap \mathsf{coNTIME}[2^{an}] \not\subseteq \mathsf{MATIME}[2^{(\log(a+1))^2\ell)}]^{\mathsf{SAT}}_{||}$ *for some constant* $a > 0$*, then*

$$\mathsf{prAM} \subseteq \text{io-}\mathsf{NP}$$
$$\mathsf{prBPP}^{\mathsf{SAT}}_{||} \subseteq \text{io-}\mathsf{P}^{\mathsf{SAT}}_{||}.$$

Proof. We argue the result for prAM first. Let $\Pi \in \mathsf{prAMTIME}[n^k]$ for some constant k and let L be a hard language in $\mathsf{NTIME}[2^{an}] \cap \mathsf{coNTIME}[2^{an}]$. Let f be a function mapping every string in $\{0,1\}^n$ to the truth table of L at input length $\ell = \Theta(\log n)$ to be set precisely later. Note that $f \in \mathsf{NTIME}[T(n)]$ for $T(n) = 2^{(a+1)\ell}$. Instantiate the generator H of Corollary 25 with f, run H on input 0^n and co-nondeterministic circuit size $m = O(n^{2k})$, and use it to derandomize P in time $\mathsf{poly}(T(n), n) = \mathsf{poly}(n)$.

If the simulation fails for some input of almost-all input lengths, then for almost-all input lengths n there exists an $x \in \Pi_N$ of length n such that the simulation errs on x, i.e., the circuit D_x of Proposition 17 instantiated with the protocol for Π and x is a distinguisher for $H(0^n, D_x)$. Let R be the reconstructor of Corollary 25 and consider the following Merlin-Arthur protocol for L, where the protocol has parallel oracle access to SAT: On input $z \in \{0,1\}^\ell$, Merlin sends x, and Arthur runs $R(0^n, D_x)$ to compute the z-th bit of $f(0^n)$ (which equals $L(z)$). If R outputs $\perp$, then the protocol rejects, otherwise, it accepts if and only if R outputs 1. Because R is a probabilistic algorithm with parallel access to an oracle for SAT, Arthur can sample the randomness required for it and then run the underlying deterministic parallel-SAT-oracle computation, meaning this is indeed a $\mathsf{MA}^{\mathsf{SAT}}_{||}$ protocol. Completeness follows since Merlin can send a correct value of x, and soundness follows from the strong resilience property of R: Even if Merlin sends a "bad" x', R is still guaranteed to either fail or output $L(z)$ with high probability.

To finish the argument for prAM, note that the running time of the protocol is just the running time of R, which is $\mathsf{poly}(n) \cdot (m \cdot \log T(n))^{O((\log r)^2)}$ for $r = O(\log(T(n))/\log m)$. Since $m = O(n^{2k})$ and setting $\ell = dk \log n$, we have $r = O(d(a+1))$ and the running time for the protocol is upper bounded by $n^{O(k(\log(d(a+1)))^2)}$. In terms of the input length ℓ, this is $2^{(\log(a+1))^2\ell)}$ when d is a sufficiently large constant depending on a.

The simulation for $\mathsf{prBPP}^{\mathsf{SAT}}_{||}$ is similar to before and the reconstruction is identical to the prAM case: If the simulation fails, then there is a query q of length $O(n^k)$ (which results in a distinguisher of size $O(n^{2k})$) that Merlin can send Arthur to make Arthur output $L(z)$ with high probability. Soundness also follows exactly as in the prAM case and the running time is again $2^{(\log(a+1))^2\ell)}$. ◀

We only state the previous result for the high-end parameter setting because stronger results are already known for the low end. For example, to conclude a subexponential derandomization of prAM, it suffices for there to exist a language in $\mathsf{NEXP} \cap \mathsf{coNEXP}$ that is hard for a subclass of $\mathsf{MA}^{\mathsf{SAT}}_{||}$ [1]. In comparison with ours, other results that conclude the same derandomization either require hardness of nondeterministic algorithms against much larger deterministic time bounds, e.g., $\mathsf{NE} \cap \mathsf{coNE} \not\subseteq \mathsf{DTIME}[2^{2^{n^\epsilon}}]$ for some $\epsilon > 0$ [14] or hardness of deterministic algorithms against slightly less space, e.g., $\mathsf{E} \not\subseteq \mathsf{SPACE}[2^{\epsilon n}]$ for some $\epsilon > 0$ [21].

7 Unconditional mild derandomization

In this section, we establish our unconditional mild derandomization result for prAM and extend it to $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$. We employ a similar win-win argument to that of the proof of Theorem 7: Either some hardness assumption/class separation holds (here, $\Sigma_2\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$), in which case we get derandomization right away. Or else we get a complexity collapse which we can use to construct a hard function f that has the efficiency requirements we need to apply one of our targeted hitting-set constructions (in this case Theorem 32, which requires hardness against $\mathsf{BPTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{SAT}}$).

As a first step toward the win-win argument, we prove an "easy-witness lemma" for $\Sigma_2\mathsf{EXP}$, which allows for the collapse $\mathsf{P}^{\Sigma_2\mathsf{EXP}} \subseteq \mathsf{EXP}$ from the assumption that $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$. Then we consider two cases:

- $\Sigma_2\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$. In this case, the derandomization result follows from standard hardness vs. randomness tradeoffs.
- $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$. In this case, we diagonalize against $\mathsf{BPTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{SAT}}$ in $\mathsf{P}^{\Sigma_2\mathsf{EXP}} = \mathsf{EXP}$, and then instantiate Theorem 32 to conclude the proof.

To diagonalize against $\mathsf{BPTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{SAT}}$, we make use of the inclusion $\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \mathsf{P}_{||}^{\mathsf{prAM}}$ and diagonalize against deterministic algorithms with non-adaptive oracle access to prAM instead.

7.1 Nondeterministic easy witnesses

In this section, we prove our "easy witness lemma" for $\Sigma_2\mathsf{EXP}$. One way of thinking of Σ_2 computations is as follows: On input x, guess a string y and then run a co-nondeterministic verifier on input (x, y). This view allows us to abstract the co-nondeterministic verification and think of y as a witness for x. In this section, we show that if $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$, then every language in $\Sigma_2\mathsf{EXP}$ has witnesses that are the truth tables of functions computed by polynomial-size single-valued circuits. To do so, we use the following result to convert hardness against single-valued circuits into hitting sets for co-nondeterministic circuits.

▶ **Lemma 34** ([27]). *There is a universal constant b and a deterministic polynomial-time algorithm that, on input 1^m and a truth table y of a function with single-valued circuit complexity at least m^b, outputs a set S of size $O(|y|^b)$ that hits co-nondeterministic circuits of size m that accept at least half of their inputs.*

We also need the following equivalence from [1].

▶ **Lemma 35** ([1]). $\Sigma_2\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$ *if and only if* $\mathsf{prAM} \subseteq \text{io-}\Sigma_2\mathsf{TIME}[2^{n^\epsilon}]/n^\epsilon$ *for all* $\epsilon > 0$.

We are now ready to prove our easy witness result for $\Sigma_2\mathsf{EXP}$.

▶ **Theorem 36.** *Assume $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$. Then $\Sigma_2\mathsf{EXP}$ has single-valued witnesses of polynomial size, i.e., for every $L \in \Sigma_2\mathsf{EXP}$ and linear-time (in its input length) co-nondeterministic verifier H for L, the following holds: For every $x \in L$, there exists a single-valued circuit C_x of size $\mathsf{poly}(|x|)$ such that $H(x, \cdot)$ accepts the exponential-length truth table of C_x.*

Proof. We show that $\Sigma_2\mathsf{E}$ has single-valued witness circuits of size n^c for some constant c. The result for $\Sigma_2\mathsf{EXP}$ then follows by padding.

Assume that $\Sigma_2\mathsf{E}$ does not have single-valued witness circuits of size n^c for any constant c. This implies that for all $c \geq 1$, there is a co-nondeterministic verifier H_c that takes as input a string x and a string y of length $2^{O(|x|)}$, runs in time $2^{O(|x|)}$, and has the following property: For infinitely many n, there is a input x' of length n such that $H_c(x', y')$ accepts for some y', but *every* y accepted by $H_c(x', \cdot)$ has single-valued circuit complexity at least n^c. Thus, there are infinitely many n such that, if we give x' as n bits of advice, guess a string y of length $2^{O(n)}$, and verify that $H_c(x', y)$ accepts (using co-nondeterminism), we are guaranteed that y encodes the truth table of a function with single-valued circuit complexity at least n^c. This gives us a Σ_2-procedure for obtaining hard functions, which we use to derandomize prAM and obtain a contradiction to Lemma 35.

Let $\Pi \in \mathsf{prAM}$ and let P be a protocol for Π that runs in time $O(\ell^a)$ on input length ℓ. By Proposition 17, to derandomize P it suffices to have a set S that hits any co-nondeterministic circuit of size $O(\ell^{2a})$ that accepts at least half of its inputs. To obtain such a set using Lemma 34, we need to first obtain a truth table of single-valued circuit complexity at least $\Omega(\ell^{2ab})$, where b is the constant from the lemma. Recall that our objective is to obtain a subexponential (time 2^{n^ϵ} for all $\epsilon > 0$) simulation. To this end, let $\epsilon > 0$ be sufficiently small and consider the verifier H_c for $c = 3ab/\epsilon$ on inputs of length $n = \ell^\epsilon$. If n is one of the infinitely many input lengths for which there exists x' such that every string accepted by $H_c(x', \cdot)$ has single-valued circuit complexity at least $n^c = \ell^{3ab}$, then we can obtain such a hard string by having x' as advice, guessing $y \in \{0,1\}^{2^{O(\ell^\epsilon)}}$ and verifying that $H_c(x', y)$ accepts.

In parallel, apply Lemma 34 to y to obtain a set S of size $2^{O(\ell^\epsilon)}$, and use S to derandomize the prAM computation (guessing a Merlin response for each string in S). Finally, accept if and only if both $H_c(x', y)$ and the prAM simulation accept. All of this can be carried out in $\Sigma_2\mathsf{TIME}[2^{O(\ell^\epsilon)}]/\ell^\epsilon$. Since ϵ is an arbitrarily small constant and the simulation works for infinitely many input lengths ℓ, we obtain a contradiction to Lemma 35. ◀

Theorem 36 allows us to establish the following complexity class collapse in case $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$. The corollary represents the role our easy witness result plays in the proof of Theorem 9.

▶ **Corollary 37.** *If* $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$*, then* $\mathsf{P}^{\Sigma_2\mathsf{EXP}} = \mathsf{EXP}$.

Proof. Under the hypothesis from the statement, we show that $\Sigma_2\mathsf{EXP} = \mathsf{coNEXP}$, which suffices by combining Lemma 35 and Lemma 30. The hypothesis and Lemma 35 guarantee the negation of $\mathsf{prAM} \subseteq \text{io-}\Sigma_2\mathsf{TIME}[2^{n^\epsilon}]/n^\epsilon$ for all ϵ, which in turn implies the negation of $\mathsf{prAM} \subseteq \text{io-}\mathsf{NTIME}[2^{n^\epsilon}]/n^\epsilon$ for all ϵ, and thus the contrapositive of Lemma 30 implies $\mathsf{EXP} = \mathsf{NEXP}$ and therefore $\Sigma_2\mathsf{EXP} = \mathsf{coNEXP} = \mathsf{EXP}$. Finally, we have $\mathsf{P}^{\Sigma_2\mathsf{EXP}} = \mathsf{P}^{\mathsf{EXP}} = \mathsf{EXP}$.

To show that $\Sigma_2\mathsf{EXP} = \mathsf{coNEXP}$, by padding, it suffices to show that every $L \in \Sigma_2\mathsf{E}$ is in coNEXP. Fix $L \in \Sigma_2\mathsf{E}$. By Theorem 36, L has single-valued witnesses of size n^c for some constant c. On input $x \in \{0,1\}^n$, we cycle through all nondeterministic circuits C of size n^c and compute their truth tables in time $O(2^{n^c})$. For each truth table T, we then run $V(x, T)$ (where V is a co-nondeterministic verifier for L), accepting if and only if some verification accepts. All of this runs in exponential co-nondeterministic time, so we are done. ◀

7.2 Simulation

We now execute our win-win strategy and establish Theorem 9 and its strengthening for $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$ in lieu of prAM. We first consider the case where $\Sigma_2\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$. In this case simulations of the required type that work on all inputs of a given length are provided by Lemma 35 for prAM. We argue the same simulations follow for $\mathsf{prBPP}_{||}^{\mathsf{SAT}}$.

▶ **Lemma 38.** *If* $\Sigma_2\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$*, then for every* $\epsilon > 0$

$$\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \text{io-}\Sigma_2\mathsf{TIME}[2^{n^\epsilon}]/n^\epsilon.$$

Proof. We use the inclusion $\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \mathsf{P}_{||}^{\mathsf{prAM}}$. Let k be a constant and M be an $O(n^k)$-time deterministic machine with non-adaptive oracle access to a paddable prAM-complete problem $\Gamma \in \mathsf{prAMTIME}[n]$. We assume that all queries made by M on inputs of length n are of length $O(n^k)$ at the expense of increasing M's running time to $O(n^{2k})$.

Our approach is to use Lemma 34 to derandomize the queries made to Γ while making sure that the overall simulation of M can be carried out in subexponential Σ_2-time. To derandomize Γ at input length $O(n^k)$ using the lemma, we need to obtain a truth table of single-valued circuit complexity at least $\Omega(n^{2bk})$, where b is the constant from the lemma. Let $\epsilon > 0$ and $L \in \Sigma_2\mathsf{E}$ be a language that has nondeterministic circuit complexity at least $n^{3bk/\epsilon}$ for infinitely many input lengths (which is guaranteed to exist by the hypothesis of the theorem). The simulation of M on inputs x goes as follows: Given as advice the number of strings of length n^ϵ that are in L, the Σ_2 algorithm guesses the truth table of L at input length n^ϵ, verifies it, and uses it as the string y in Lemma 34. More precisely, after guessing the truth table, the algorithm performs the following operations in parallel:

- It uses an existential and a universal guess to verify that the guessed truth table for L is correct. This is possible because the algorithm has as advice the number of strings of length n^ϵ that are in L, and thus it can existentially guess which strings are in L and only verify those, with the guarantee that the others are not in L.
- It guesses which of the queries to Γ that M makes on input x are answered positively and which are answered negatively. For each query that is guessed to be answered positively, it uses the set S from Lemma 34 and the existential phase to verify that there is a random-bit string in S for which Merlin can provide a witness. Similarly, it uses S and the universal phase to verify each query that is guessed to be answered negatively.

We note that the only existential computation paths that survive the computation are the ones where the truth table of L at input length n^ϵ was guessed correctly. In this case, and in the case that n^ϵ is one of the infinitely many input lengths where L has nondeterministic circuit complexity at least $n^{3bk/\epsilon}$, it holds that the guessed truth table has high enough (single-valued) nondeterministic circuit complexity such that S hits the co-nondeterministic circuits given by Proposition 17 for negative instances of Γ at input length $O(n^k)$. This further guarantees that the surviving existential computation paths are those that correctly guess the answers to all queries M makes on input x that are in the promise of Γ. This suffices to obtain a simulation of M that is correct on infinitely many input lengths since M is insensitive to variations in answers to queries that are outside the promise (even when the same query is answered differently on different occasions). Finally, we note that the entire procedure runs in time $2^{O(n^\epsilon)}$, which can be made smaller than $2^{n^{\epsilon'}}$ for any $\epsilon' > 0$ by taking ϵ to be sufficiently small. ◀

The other case of the win-win analysis is when $\Sigma_2\mathsf{EXP} \subseteq \mathsf{NP/poly}$. In this case, we use the collapse $\mathsf{P}^{\Sigma_2\mathsf{EXP}} = \mathsf{EXP}$ given by Corollary 37 and our targeted hitting-generator construction to obtain the desired simulation. We conclude:

▶ **Theorem 39** (Strengthening of Theorem 9)**.** *For every* $\epsilon > 0$ *and every* $e > 0$

$$\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \text{io-}\mathsf{Heur}_{1/n^e}\Sigma_2\mathsf{TIME}[2^{n^\epsilon}]/n^\epsilon.$$

Proof. If $\Sigma_2\mathsf{EXP} \not\subseteq \mathsf{NP/poly}$, then it follows that $\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \Sigma_2\mathsf{TIME}[2^{n^\epsilon}]/n^\epsilon$ for all $\epsilon > 0$ by Lemma 38. Otherwise, by Corollary 37, we have that $\mathsf{P}^{\Sigma_2\mathsf{EXP}} = \mathsf{EXP}$. By Theorem 31, all we need to show is that $\mathsf{P}^{\Sigma_2\mathsf{EXP}} \not\subseteq \bigcup_{b\in\mathbb{N}} \mathsf{BPTIME}[n^{b((\log n)^2)}]_{||}^{\mathsf{SAT}}$. Given the containment $\mathsf{prBPP}_{||}^{\mathsf{SAT}} \subseteq \mathsf{P}_{||}^{\mathsf{prAM}}$ and a padding argument, it follows that $\bigcup_{b\in\mathbb{N}} \mathsf{BPTIME}[n^{b((\log n)^2)}]_{||}^{\mathsf{SAT}} \subseteq \mathsf{DTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{prAM}}$. It remains to show that $\mathsf{P}^{\Sigma_2\mathsf{EXP}} \not\subseteq \mathsf{DTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{prAM}}$, which we do by diagonalization.

Fix a prAM-complete problem Γ and note that if $L \in \mathsf{DTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{prAM}}$, then there exists a Turing machine M running in time $2^{\mathsf{polylog}(n)}$ with non-adaptive oracle access to Γ that computes L. Thus, it suffices to diagonalize against such machines with Γ as an oracle. Let S be the following $\Sigma_2\mathsf{EXP}$-oracle machine: On input $x \in \{0,1\}^n$, interpret x as a non-adaptive oracle Turing machine M_x with an oracle for Γ. Then, using binary search and the $\Sigma_2\mathsf{EXP}$ oracle, compute the number q of queries that M_x on input x makes that are answered negatively, where we let M_x run for at most 2^n steps. This is possible in $\mathsf{P}^{\Sigma_2\mathsf{EXP}}$ because $\mathsf{prAM} \subseteq \Pi_2\mathsf{P}$, so we can verify negative instances in $\Sigma_2\mathsf{EXP}$. Once we know q, we can simulate $M_x(x)$ for at most 2^n steps in $\Sigma_2\mathsf{EXP}$ as follows: Guess which q queries are negative and verify them in $\Sigma_2\mathsf{EXP}$ (again using the fact that $\mathsf{prAM} \subseteq \Pi_2\mathsf{P}$); then assume that the other queries are answered positively and simulate $M_x(x)$ directly with these answers. By querying the $\Sigma_2\mathsf{EXP}$ oracle S then outputs the opposite of this simulation. By construction, the language of S is in $\mathsf{P}^{\Sigma_2\mathsf{EXP}} \setminus \mathsf{DTIME}[2^{\mathsf{polylog}(n)}]_{||}^{\mathsf{prAM}}$. ◀

This concludes our discussion of the byproducts of our main results.

References

1. Barış Aydınlıoğlu and Dieter van Melkebeek. Nondeterministic circuit lower bounds from mildly derandomizing Arthur-Merlin games. *Computational Complexity*, 26(1):79–118, 2017. doi:10.1007/s00037-014-0095-y.
2. László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993. doi:10.1007/BF01275486.
3. László Babai and Shlomo Moran. Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes. *Journal of Computer and System Sciences*, 36(2):254–276, 1988. doi:10.1016/0022-0000(88)90028-1.
4. Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil Vadhan. Robust PCPs of proximity, shorter PCPs, and applications to coding. *SIAM Journal on Computing*, 36(4):889–974, 2006. doi:10.1137/S0097539705446810.
5. Venkatesan T. Chakaravarthy and Sambuddha Roy. Arthur and Merlin as oracles. *Computational Complexity*, 20(3):505–558, 2011. doi:10.1007/s00037-011-0015-3.
6. L. Chen, R. D. Rothblum, and R. Tell. Unstructured hardness to average-case randomness. In *Symposium on Foundations of Computer Science (FOCS)*, pages 429–437, 2022. doi:10.1109/FOCS54457.2022.00048.
7. Lijie Chen, Ron D. Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds: Extended abstract. In *Symposium on Foundations of Computer Science (FOCS)*, pages 13–23, 2020. doi:10.1109/FOCS46700.2020.00010.
8. Lijie Chen and Roei Tell. Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise. In *Symposium on Foundations of Computer Science (FOCS)*, 2021. doi:10.1109/FOCS52979.2021.00021.
9. Oded Goldreich. In a world of P=BPP. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 191–232. Springer, 2011. Part of the Lecture Notes in Computer Science book series (LNCS, volume 6650). doi:10.1007/978-3-642-22670-0_20.

10 Oded Goldreich. On doubly-efficient interactive proof systems. *Foundations and Trends in Theoretical Computer Science*, 13:157–246, 2018. doi:10.1561/0400000084.

11 Oded Goldreich. Two comments on targeted canonical derandomizers. In *Computational Complexity and Property Testing: On the Interplay Between Randomness and Computation*, pages 24–35. Springer, 2020. doi:10.1007/978-3-030-43662-9_4.

12 Shafi Goldwasser, Yael Tauman Kalai, and Guy N. Rothblum. Delegating computation: Interactive proofs for muggles. *Journal of the ACM*, 62(4), 2015. doi:10.1145/2699436.

13 Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. Uniform hardness versus randomness tradeoffs for Arthur-Merlin games. *Computational Complexity*, 12(3):85–130, 2003. doi:10.1007/s00037-003-0178-7.

14 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002. doi:10.1016/S0022-0000(02)00024-7.

15 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Symposium on Theory of Computing (STOC)*, page 220–229, 1997. doi:10.1145/258533.258590.

16 Russell Impagliazzo and Avi Wigderson. Randomness vs time: Derandomization under a uniform assumption. *Journal of Computer and System Sciences*, 63(4):672–688, 2001. doi:10.1006/jcss.2001.1780.

17 Adam R. Klivans and Dieter van Melkebeek. Graph nonisomorphism has subexponential size proofs unless the polynomial-time hierarchy collapses. *SIAM Journal on Computing*, 31(5):1501–1526, 2002. doi:10.1137/S0097539700389652.

18 Yanyi Liu. Personal communication, October 2022.

19 Yanyi Liu and Rafael Pass. Characterizing derandomization through hardness of Levin-Kolmogorov complexity. In *Computational Complexity Conference (CCC)*, volume 234, pages 35:1–35:17, 2022. doi:10.4230/LIPIcs.CCC.2022.35.

20 Yanyi Liu and Rafael Pass. Leakage-resilient hardness v.s. randomness. In *Computational Complexity Conference (CCC)*, 2023. URL: https://eccc.weizmann.ac.il/report/2022/113/.

21 Chi-Jen Lu. Derandomizing Arthur-Merlin games under uniform assumptions. *Computational Complexity*, 10(3):247–259, 2001. doi:10.1007/s00037-001-8196-9.

22 Peter Bro Miltersen and N. V. Vinodchandran. Derandomizing Arthur–Merlin games using hitting sets. *Computational Complexity*, 14(3):256–279, 2005. doi:10.1007/s00037-005-0197-7.

23 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994. doi:10.1016/S0022-0000(05)80043-1.

24 Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *Journal of the ACM*, 52(2):172–216, 2005. doi:10.1145/1059513f.1059516.

25 Ronen Shaltiel and Christopher Umans. Low-end uniform hardness versus randomness tradeoffs for AM. *SIAM Journal on Computing*, 39(3):1006–1037, 2009. doi:10.1137/070698348.

26 Luca Trevisan and Salil Vadhan. Pseudorandomness and average-case complexity via uniform reductions. *Computational Complexity*, 16(4):331–364, 2007. doi:10.1007/s00037-007-0233-x.

27 Christopher Umans. Pseudo-random generators for all hardnesses. *Journal of Computer and System Sciences*, 67(2):419–440, 2003. doi:10.1016/S0022-0000(03)00046-1.

28 R. Ryan Williams. Natural proofs versus derandomization. *SIAM Journal on Computing*, 45(2):497–529, 2016. doi:10.1137/130938219.

Tight Correlation Bounds for Circuits Between AC0 and TC0

Vinayak M. Kumar
Department of Computer Science, University of Texas at Austin, TX, USA

Abstract

We initiate the study of generalized AC^0 circuits comprised of arbitrary unbounded fan-in gates which only need to be constant over inputs of Hamming weight $\geq k$ (up to negations of the input bits), which we denote $\mathsf{GC}^0(k)$. The gate set of this class includes biased LTFs like the k-OR (outputs 1 iff $\geq k$ bits are 1) and k-AND (outputs 0 iff $\geq k$ bits are 0), and thus can be seen as an interpolation between AC^0 and TC^0.

We establish a tight multi-switching lemma for $\mathsf{GC}^0(k)$ circuits, which bounds the probability that several depth-2 $\mathsf{GC}^0(k)$ circuits do not simultaneously simplify under a random restriction. We also establish a new depth reduction lemma such that coupled with our multi-switching lemma, we can show many results obtained from the multi-switching lemma for depth-d size-s AC^0 circuits lifts to depth-d size-$s^{.99}$ $\mathsf{GC}^0(.01 \log s)$ circuits with *no loss in parameters* (other than hidden constants).

Our result has the following applications:

- Size-$2^{\Omega(n^{1/d})}$ depth-d $\mathsf{GC}^0(\Omega(n^{1/d}))$ circuits do not correlate with parity (extending a result of Håstad (SICOMP, 2014)).
- Size-$n^{\Omega(\log n)}$ $\mathsf{GC}^0(\Omega(\log^2 n))$ circuits with $n^{.249}$ arbitrary threshold gates or $n^{.499}$ arbitrary symmetric gates exhibit exponentially small correlation against an explicit function (extending a result of Tan and Servedio (RANDOM, 2019)).
- There is a seed length $O((\log m)^{d-1} \log(m/\varepsilon) \log\log(m))$ pseudorandom generator against size-m depth-d $\mathsf{GC}^0(\log m)$ circuits, matching the AC^0 lower bound of Håstad up to a $\log\log m$ factor (extending a result of Lyu (CCC, 2022)).
- Size-m $\mathsf{GC}^0(\log m)$ circuits have exponentially small Fourier tails (extending a result of Tal (CCC, 2017)).

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Pseudorandomness and derandomization

Keywords and phrases AC^0, TC^0, Switching Lemma, Lower Bounds, Correlation Bounds, Circuit Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.18

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2023/045/

Funding *Vinayak M. Kumar*: Supported by NSF Grant CCF-2008076 and a Simons Investigator Award (#409864, David Zuckerman).

Acknowledgements The author thanks David Zuckerman and Chin Ho Lee for many valuable discussions, Xin Lyu for explaining his work in [18], anonymous reviewers for valuable feedback and for pointing us to the construction presented in Theorem 54, and Jeffrey Champion, Shivam Gupta, Michael Jaber and Jiawei Li for helpful comments.

1 Introduction

Proving superpolynomial circuit lower bounds against explicit functions is one of the most central questions in complexity theory. However, after the initial flurry of work resulting in Blum's lower bound of $3n - o(n)$ [5], followed by a recent revival 30 years later leading to the state of the art $3.1n - o(n)$ size lower bound by Li and Yang [15], this problem has proven to be extremely difficult. Furthermore, there are various proof barriers that give strong evidence that our current intuition is not developed enough to tackle this problem [4, 25, 1].

In order to gain more understanding on this problem, researchers considered circuits with constant depth whose gates are AND, OR, or NOT with unbounded fanin. To this end there has been a fruitful line of work culminating in the state of the art average case hardness of depth d size $2^{\Omega(n^{\frac{1}{d-1}})}$ AC^0 circuits computing the parity and majority functions [7, 9] (with this result being tight for parity). A natural followup question to ask is how powerful AC^0 would then be if $\oplus$ (parity) or MAJ (majority) gates were added, corresponding to the circuit classes $\mathsf{AC}^0[\oplus]$ and TC^0. With regard to $\mathsf{AC}^0[\oplus]$, Oliveira, Santhanam, and Srinivasan [21] proved that it is average case hard for any size-$2^{\Omega(n^{1/(2d-4)})}$ $\mathsf{AC}^0_d[\oplus]$ circuit to compute MAJ, improving earlier work by Razborov [24]. Smolensky [29] proved exponential size lower bounds even if one replaces the $\oplus$ gate with a MOD_p gate for prime p (MOD_p is the gate that outputs 0 iff p divides the sum of the input bits.).

As we see, MAJ is a hard function demonstrating exponential circuit lower bounds for almost all the circuit classes mentioned thus far, and so we would guess TC^0 is extremely powerful and thus challenging to show circuit lower bounds for. This is evident in the current state of the art for TC^0, which is stark in contrast with the landscape of $\mathsf{AC}^0[\oplus]$. In 1993, Impagliazzo, Paturi, and Saks [10] showed that parity is hard for depth-d size-$\Omega(n^{1+\varepsilon_{IPS}^{-d}})$ circuits for some constant $C_{IPS} > 1$, which remains as the current state of the art modulo the case of $d = 2$, where Kane and Williams established a $n^{2.49}$-size lower bound [11]. In fact, a bootstrapping result by Chen and Tell [6] shows that if one slightly improves (e.g. decreases C_{IPS}) this superlinear lower bound against certain NC^1-complete problems (NC^1 is the class of $O(\log n)$-depth, polysized, constant fan-in circuits), we would immediately get superpolynomial lower bounds and a separation of TC^0 and NC^1, attesting to the hardness of this task.

Due to the halted state of affairs for TC^0 circuits, we study a circuit class not as strong as TC^0, but still captures the motivation of analyzing "AC^0 with the power of majority." After it had been shown AC^0 circuits cannot efficiently compute the majority of n bits, it seemed natural that the next step would be to add unbounded MAJ gates to AC^0 to create TC^0. However, due to having unbounded fan-in, TC^0 gives a size-s circuit the power to calculate the majority of up to s bits. Hence, one could argue the reason why size s TC^0 circuits are much harder to analyze than AC^0 is because they are getting much more power than simply calculating the majority of n bits when $s \gg n$. In order to maintain the unbounded fan-in property of the circuit but also ration the computational power we give AC^0 to be "just sufficient" to compute the majority of n bits, one can consider the following circuit class.

▶ **Definition 1** ($\mathsf{AC}^0(k)$ Circuits)**.** *Define the unbounded fan-in gates k-OR to output 1 iff there are at least k ones in the input string, and k-AND to output 0 iff there are at least k zeros in the input string. Define the class of constant depth circuits created by negations and $\{k'\text{-AND}, k'\text{-OR}\}$s for $k' \leq k$ to be $\mathsf{AC}^0(k)$.*

One can observe that $\mathsf{AC}^0(n/2)$ is a natural circuit class that contains the majority of n bits and doesn't add "extra power" like the majority of a much larger quantity of bits. Therefore, analyzing $\mathsf{AC}^0(n/2)$ will give us a better understanding on how much power majority gives to circuits. More generally, $\mathsf{AC}^0(k)$ also allows us to nicely interpolate between AC^0 and TC^0, since a size-s AC^0 circuit is characterized by $\mathsf{AC}^0(1)$, while a size-s TC^0 circuit is characterized by $\mathsf{AC}^0(s/2)$. Hence, studying $\mathsf{AC}^0(k)$ for increasing k is a necessary step and a compelling intermediary model that can help us understand the power of TC^0.

For how large of a k will $\mathsf{AC}^0(k)$ trivially collapse to AC^0? An immediate observation is that $\mathsf{AC}^0(k)$ contains the majority gate over $2k$ bits, for which we know $2^{\Omega(k^{1/2d})}$-size AC^0 lower bounds. Hence, for $k = \mathsf{polylog}(n)$, we have a superpolynomial size seperation between

AC^0 and $\mathsf{AC}^0(k)$. Even for any $k = \omega(1)$, it is unknown whether $\mathsf{AC}^0(k)$ is equivalent to AC^0. A standard argument would be to represent a k-OR with fan-in m as a width-k DNF with $\binom{m}{k}$ clauses (check over all size k subsets of input bits to see if some subset are all 1s) or a width-$(m-k)$ CNF with $\binom{m}{k}$ clauses (check over all size $m-k$ subsets of input bits to see if all subsets contain some 1). Therefore, if we have a size s circuit made from k-OR and k-AND gates, we can turn this into an AC^0 circuit with size $s \cdot \binom{s}{k} \approx s^k$ (since a gate in the original circuit can have fan in size up to s) and depth $d+1$ (one can naively get depth $2d$, but by alternating CNFs and DNFs, we can collapse the depth to $d+1$). Hence, we see that a size s lower bound for AC^0_d translates to a size $s^{1/k}$ lower bound for $\mathsf{AC}^0_{d-1}(k)$. This reduction is not an equivalence, as we pay with a reduction in depth, as well as an asymptotically weaker size lower bound for any $k = \omega(1)$. For example, a polynomial size bound for AC^0 cannot be converted to a polynomial size lower bound for $\mathsf{AC}^0(k)$ for any superconstant k. Consequently, the relationship between AC^0 and $\mathsf{AC}^0(k)$ already becomes nontrivial in the mild regime of $k = \omega(1)$.

In this paper, we study an *even more* general class of circuits, which we denote as $\mathsf{GC}^0(k)$.

▶ **Definition 2** ($\mathsf{G}(k)$ gates/$\mathsf{GC}^0(k)$ circuits)**.** *Let $\mathsf{G}(k)$ be the set of all unbounded fan-in gates that are constant over all input bits with $\geq k$ ones, or over all input bits with $\geq k$ zeros (notice for $k \geq 1$ this includes negations by definition). Define $\mathsf{GC}^0(k)$ to be the class of constant depth circuits created from $\mathsf{G}(k)$ gates.*

Some concrete examples of $\mathsf{G}(k)$ gates are arbitrary gates of fan-in k, majority of $2k$ bits, the k-OR, and functions that compute parity if the input has $< k$ ones, and is 0 otherwise. Notice that this is indeed a generalization of $\mathsf{AC}^0(k)$.

On top of being an alternative generalization of AND/OR gates which may be of independent interest, one nice property about $\mathsf{G}(k)$ is that it includes a generalized notion of k-AND and k-OR gates to arbitrary LTFs (functions of the form $\operatorname{sgn}(\sum w_i x_i - \theta)$).

▶ **Definition 3** (k-balanced LTFs)**.** *Let $f(x) = sgn(\sum_{i=1}^n w_i x_i - \theta)$ be an* LTF*, and let $\sigma : [n] \to [n]$ be a permutation sorting the w_i in increasing magnitude (i.e. $|w_{\sigma(1)}| \leq \cdots \leq |w_{\sigma(n)}|$). We say f is k-*balanced *if k is the smallest index j such that $-\sum_{i \leq j} |w_{\sigma(i)}| + \sum_{i > j} |w_{\sigma(i)}| < |\theta|$.*

One can verify that k-balanced LTFs are indeed in $\mathsf{G}(k)$ (see Theorem 51). Therefore, our results can also be seen as a study of arbitrary LTFs that are biased.

Various notions of *balancedness* (or *regularity* in some literature) for LTFs has been defined in previous work about threshold functions [26, 22, 8], but are all distinct from the combinatorial definition we have proposed. In light of being able to show lower bounds for this characterization of *balanced*, it may be of interest to explore this class of balanced LTFs in other contexts regarding LTF circuit complexity.

1.1 Our Results

We outline all the results we obtain regarding $\mathsf{AC}^0(k)$ (or more generally $\mathsf{GC}^0(k)$) circuits. The core result from which all the other results are derived from is an optimal multi-switching lemma for $\mathsf{GC}^0(k)$ circuits. We state the result without getting into the fine-grained definitions.

▶ **Theorem 4** (Multi-Switching Lemma for $\mathsf{GC}(k)$ Circuits (Informal))**.** *Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of $\mathsf{G}(k) \circ \mathsf{AND}_w$ circuits on $\{0,1\}^n$. Then*

$$\Pr_{\rho \sim R_p}[\mathcal{F}|_\rho \text{ do not all simultaneously "simplify"}] \leq (2^k m)^{t/r}(O(pw))^t$$

The theorem statement and proof is formally written in Theorem 20. This bound can be proven to be optimal in the regime of large t. See the appendix (Theorem 52) for the proofs of this claim.

It is illuminating to compare this result to the multi-switching lemma for AC^0 circuits, which bounds the probability by the very similar expression of $m^{t/r}(O(pw))^t$. The only difference is that in the new lemma, the m and 2^k are coupled in the base of the exponent. This seems to hint that as long as $2^k = O(m)$, one gets the same probability bound when using either the AC^0 or $\mathsf{GC}^0(k)$ version of the multi-switching lemma. In practice, the parameter m is upper bounded by s, the size of the circuit. Hence, we would intuit that any result obtained from the multi-switching lemma for depth d size s AC^0 circuits can then be lifted to size s $\mathsf{GC}^0_d(\log s)$ circuits. This indeed turns out to be the case as we demonstrate through four different results. We obtain a surprising lifting theorem: any depth d size s AC^0 lower bound obtained by the multi-switching lemma immediately lifts to depth d size $s^{.99}$ $\mathsf{GC}^0(.01 \log s)$-circuits *with no loss in parameters*. We demonstrate three different results exhibiting this phenomenon.

For the first result, denote PAR to be the parity gate.

▶ **Theorem 5** (Optimal Correlation Bounds Against Parity). *Let C be a size m depth d $\mathsf{GC}^0(k)$-circuit. Then the correlation of C against parity is*

$$|\mathbb{E}_x[(-1)^{C(x)+\mathsf{PAR}(x)}]| \leq 2^{-\Omega_d(n/(k+\log m)^{d-1})+k}.$$

In particular, we get a $2^{\Omega(n^{1/d})}$-size lower bound for $\mathsf{GC}^0_d(\Omega(n^{1/d}))$ circuits almost matching the lower bound of $2^{\Omega(n^{1/(d-1)})}$ we know for AC^0! This is especially surprising in light of the fact that $\mathsf{GC}^0_d(\Omega(n^{1/d}))$ is a much stronger class than AC^0; there exist singleton $\mathsf{G}(n^{1/d})$ gates that cannot be computed by size $O(2^{n^{1/2d}})$ AC^0_d circuits. This can be seen as a limited dual result to [24], who showed AC^0_d augmented with parity gates requires size $2^{\Omega(n^{1/2d})}$ to compute majority, whereas we show AC^0_d augmented with $n^{1/d}$-biased majority gates requires size $2^{\Omega(n^{1/d})}$ to compute parity. It also contrasts with [21], who surprisingly showed that adding parity gates to AC^0 improved optimal circuit constructions of majority. Here, we show that majority gates whose threshold value is shifted to $\Omega(n^{1/d})$ has no effect on AC^0's ability to calculate parity, even though such gates adds a lot of power to AC^0. (majority gates whose threshold has been biased to $n^{1/d}$ cannot be computed by size $2^{\Omega(n^{1/2d^2})}$ AC^0_d circuits).

Notice that this result is tight in an extremely sensitive way. Letting PAR_n denote the parity gate over n bits, we see $\mathsf{PAR}_{n^{1/d}} \in \mathsf{G}(n^{1/d})$, and we can calculate the parity of n bits by creating a depth d $n^{1/d}$-ary tree of $\mathsf{PAR}_{n^{1/d}}$ gates, where the ith layer from the bottom has $n^{1-i/d}$ $\mathsf{PAR}_{n^{1/d}}$ gates that take the parity of all the bits fed below it in blocks of $n^{1/d}$. This is a depth d size $O(n^{1-1/d})$ circuit computing parity. Therefore, we have a simple counterexample of a $\mathsf{GC}^0_d(n^{1/d})$ circuit computing parity (which is sublinear in size!). This demonstrates a sharp threshold behavior where the exponential lower bound of $2^{\Omega(n^{1/d})}$ is tight up to the hidden constant factor of the $\Omega(\cdot)$ in $\mathsf{GC}^0_d(\Omega(n^{1/d}))$, and if the constant is too large, we suddenly go from requiring exponentially large circuits to only needing sublinear size ones.

This theorem is tight in all other parameters as well. We show that this result is tight in the size parameter by giving a size-$2^{\Omega(n^{1/d})}$ $\mathsf{GC}^0(.1n^{1/d})$ circuit computing parity. Furthermore, we show that the correlation bound is tight by giving a size-m $\mathsf{GC}^0(k)$ circuit that approximates parity.

For what k will analyzing $\mathsf{AC}^0(k)$ give implications for TC^0? A result by Allender and Koucký ([2], Theorem 3.8) states that there exists an absolute constant C_{AK} such that MAJ_n can be written as an $\mathsf{AC}^0(n^\varepsilon)$ circuit with depth $\leq C_{AK}/\varepsilon$ and size $O(n^{1+\varepsilon})$. Therefore, beating the current state of the art depth d size $\Omega(n^{1+C_{IPS}^{-d}})$ lower bound for TC^0 reduces to beating depth $C_{AK}d/\varepsilon$ size $n^{(1+50^{-d})(2+\varepsilon)}$ lower bounds for $\mathsf{GC}^0(n^\varepsilon)$ circuits for any choice of ε. In our paper, we show exponential size lower bounds against parity when $\varepsilon = 1/d$ but at depth d rather than $C_{AK}d^2$. It would be interesting to see whether with some NC^1-complete problem can display strong lower bounds for $\mathsf{AC}^0(n^{1/d})$ for depth larger than d, even if it may be less than $C_{AK}d^2$ (but a function other than parity would need to be considered).

Another angle researchers have taken towards understanding the power of threshold circuits has been to start with an AC^0 circuit and augment some of the gates to arbitrary threshold gates [33, 17, 27]. Our multi-switching lemma shows that we can instead start with a base $\mathsf{GC}^0(\log s)$ circuit and obtain the same state of the art parameters as [27] if we started from an AC^0 circuit.

▶ **Theorem 6.** *There exists a function* $\mathsf{RW} \in \mathsf{P}$ *(introduced by Razborov and Widgerson [23]) and absolute constant* τ *such that for* C*, a size* $n^{\Omega(\log n)}$ $\mathsf{GC}^0(\Omega(\log^2 n))$*-circuit with* $n^{.249}$ THR *gates, we have*

$$|\mathbb{E}_x[(-1)^{\mathsf{RW}(x)+C(x)}]| \leq 2^{-\tilde{\Omega}(n^{.249})}.$$

The original motivation to study AC^0 with a small number of THR gates was to use this to gradually convert circuits gate by gate from AC^0 to TC^0. This result "speeds up" this process by augmenting *all* AC^0 gates to $\mathsf{G}(\log^2 n)$ gates (which contain unbalanced LTFs as discussed above). If one tried proving this theorem by expanding the $\mathsf{GC}^0(\log^2 n)$ circuits into an AC^0 circuit, completing the proof would require solving a longstanding open problem regarding correlation bounds against $\omega(\log n)$-party NOF protocols! In Section 4.2, we point out this observation explicitly along with the proof.

As another application, we can create PRGs for $\mathsf{GC}^0(\log m)$ circuits whose seed length matches that of size m AC^0 circuits. This is accomplished by fully derandomizing Theorem 4 and using the partition-based PRG approach in [18]. The resulting PRG for $\mathsf{GC}^0_d(\log m)$ has identical seed length as Lyu's PRG, thereby also matching Håstad's AC^0 lower bound barrier up to a $\log\log m$ factor (see [31, 28, 12] for a discussion on why an $o(\log^d(m/\varepsilon))$ seed length implies breakthrough circuit lower bounds).

▶ **Theorem 7.** *For every* $m, n, d \geq 3$ *and* $\varepsilon > 0$*, there is an* ε*-PRG for size-*m $\mathsf{GC}^0_d(\log m)$ *with seed length* $O(\log^{d-1}(m)\log(m/\varepsilon)\log\log m)$

The proof is covered in Section 4.3. Notice that if we had simply expanded out all gates as width $\log m$ CNF/DNFs, we would have a size $\approx m^{\log m}$ AC^0_{d+1} circuit, and plugging in Lyu's near-optimal PRG would yield us a suboptimal seed length of $O((\log^2 m + \log(1/\varepsilon))\log^{2d} m \log\log m)$.

Finally, we establish results on the Fourier spectrum of $\mathsf{GC}^0(k)$ circuits. It can be shown that every Boolean function, when written as a map $\{\pm1\}^n \to \{\pm1\}$, can be uniquely expressed as a multivariate polynomial $f(x) = \sum_{S\subset[n]} \widehat{f}(S)\prod_{i\in S} x_i$. We show exponentially small Fourier tail bounds for any $C \in \mathsf{GC}^0(k)$. More concretely,

▶ **Theorem 8.** *For arbitrary* $C \in \mathsf{GC}^0_d(k)$ *of size* m*, the following is true for any* $0 \leq \ell \leq n$*.*

$$\sum_{|S|\geq\ell} \widehat{C}(S)^2 \leq 2^{-\Omega\left(\frac{\ell}{(k+\log m)^{d-1}}\right)+k}.$$

Linial, Kushilevitz, Mansour, and Tal [13, 16, 19, 30] showed that with small Fourier tails, one can get a variety of Fourier structure results, efficient learning algorithms, and correlation bounds. We demonstrate applications of such techniques to $\mathsf{GC}^0(k)$ in detail in Section 4.4.

1.2 The Switching Lemma

To grasp this section, an understanding of Lyu's witness/transcript proof of the switching lemma [18] would be helpful. We still give an overview of the proof here and provide intuition from an information-theoretic lens, which differs in certain places than the intuition presented in [18]. After the overview, we will highlight the necessary changes needed to prove the more general lemma for $\mathsf{GC}^0(k)$ circuits.

Say we have a k-$\mathsf{OR} \circ \mathsf{AND}$ circuit F (in the formal proof, we consider general $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ circuits). For $\Lambda \subset [n]$ and $z \in \{0,1\}^n$, denote $\rho(\Lambda, z)$ to be the restriction/partial assignment where all variables whose indices are in Λ are kept alive/unfixed, and all remaining variables x_i with $i \notin \Lambda$ are fixed to the corresponding bit in z, z_i. Consider a random restriction $\rho(\Lambda, z)$, where $z \sim \{0,1\}^n$ is a uniformly random ground assignment, and Λ is a random subset of $[n]$ such that each element is added with probability p. To show that with low probability, $F|_\rho$ has decision tree depth $\geq t$, it suffices to create a specific *canonical decision tree* (CDT) for each ρ and argue that this tree has depth $\geq t$ with low probability (because if the decision tree depth of $F|_\rho$ is $\geq t$, then surely the canonical decision tree has depth $\geq t$). We consider the following CDT, where we first initialize a counter $ctr \leftarrow 0$, and then scan the bottom layer clauses from left to right.

- If the clause is fixed to 1 and $ctr = k$, terminate since we know that F evaluates to 1. Otherwise increment ctr and move to the next clause.
- If the clause is fixed to 0, move to the next clause.
- If the clause is ambiguous, query all variables in the clause, and behave accordingly as above.

If we think of our CDT as an algorithm that queries certain bits of the input, then *bad* ρ that creates a depth $\geq t$ CDT will produce a unique "transcript" of large size recording the behavior of CDT (i.e. the clauses and variables the CDT queries from). Like [18], we consider transcripts that store (ℓ_i), the indices of the clauses queried, along with a set P that further elucidates which variables in the clauses were queried in an information-efficient manner. We get the following inequality

$$\begin{aligned}\Pr_\rho[\mathsf{DT}(F|_\rho) \geq t] &\leq \Pr_\rho[\mathsf{CDT}(F|_\rho) \geq t] \\ &\leq \sum_{\substack{\text{large transcripts} \\ (\ell_i, P)}} \Pr_\rho[(\ell_i, P) \text{ is a large transcript for } \rho] \qquad (1)\end{aligned}$$

via the union bound. A natural thought is to then bound each term in the sum. Unfortunately, it turns out that the number of transcripts (ℓ_i, P), when counted naively by multiplying the total possible lists (ℓ_i) by the total possible sets P, is far too large to get our switching lemma due to the vast amount of possible (ℓ_i).

However, it turns out that (ℓ_i) contains *redundant* information. Say P is a partial transcript for ρ if it can be completed with a suitable (ℓ_i) to form a transcript for ρ. We can show that given ρ and P that is a partial transcript for it, there is a unique list (ℓ_i) that completes P to a full transcript. Hence

$$\sum_{\substack{\text{large transcripts} \\ (\ell_i, P)}} \Pr_{\rho}[(\ell_i, P) \text{ is a large transcript for } \rho]$$
$$= \sum_{\text{partial transcripts } P} \Pr_{\rho}[P \text{ is a partial transcript for } \rho] \tag{2}$$

which is a sum of far fewer terms, making the union bound feasible. It remains to bound each individual term in the sum.

We want to bound the probability a particular P is a partial transcript for ρ. If we were given the complementary (ℓ_i)'s, this would be easy. The (ℓ_i) along with P would give a transcript of the specific set of $\geq t$ variables that the CDT queried, which ρ must keep alive in order to have any hope of (ℓ_i, P) being a transcript for ρ. This would happen with probability $\leq p^t$, which is a sufficiently small probability to apply the union bound. However, the trickiness arises due to ℓ_i not being specified. It turns out different (ℓ_i) might couple with the same P to form transcripts for different ρ! Therefore if we use no information about ρ, then we have no hope of recovering a fixed (ℓ_i).

On the other hand, if we were given complete information about ρ, then we can recover a unique (ℓ_i) or deduce none exists. However, this eliminates all randomness of ρ and we get the trivial large upper bound of 1 for each term. Therefore, for such an approach to work, we need to condition on partial information about ρ and hope that it is enough information to recover (ℓ_i) but not too much information to the point where we get a weak bound on the probability due to the lack of randomness.

This motivates us to think of a restriction by first assigning a uniform random string z to x and then covering up a p-subset Λ with stars to create a restriction $\rho(\Lambda, z)$. The intuition for this is that hopefully the random string z, combined with P, will be enough information from ρ to fix (ℓ_i), from which we can use the remaining randomness in ρ (namely Λ) to obtain the p^t bound. In particular, we hope that there is a "transcript searcher" $\mathcal{S}$, which on input (z, P), can recover a completed transcript (ℓ_i, P) such that all ρ designed by initially assigning $x = z$ will have partial transcript P only if (ℓ_i, P) is its transcript. If such a function exists, then we could say

$$\begin{aligned}
\Pr_{\rho}[P \text{ is a partial transcript for } \rho] &= \mathbb{E}_{z \sim U_n} \Pr_{\Lambda}[P \text{ is a partial transcript for } \rho(\Lambda, z)] \\
&= \mathbb{E}_{z \sim U_n} \Pr_{\Lambda}[\mathcal{S}(z, P) \text{ is a transcript for } \rho(\Lambda, z)] \\
&\leq p^t
\end{aligned}$$

where the last inequality follows since ρ must keep the variables in the transcript alive. Alas, such an $\mathcal{S}$ cannot exist. There can exist different restrictions created from the same ground assignment z that are witnessed by different completions of P (this ambiguity is an unavoidable side effect of not being able to condition on all information about ρ). We cannot hope for a unique completion, but what if our $\mathcal{S}$ output all of these potential completions with decent probability over the randomness in z? Say ρ is *good* if P is a partial transcript for it. In formal terms, say we can construct $\mathcal{S}$ such that for any good ρ, $\Pr_z[\mathcal{S}(z, P) \text{ is a partial transcript for } \rho] \geq \gamma$ (earlier we were demanding $\gamma = 1$, which turned out to be impossible). Then we can deduce

$$\Pr_{\rho}[P \text{ is a partial transcript for } \rho] \tag{3}$$

$$= \mathbb{E}_{\Lambda} \Pr_{z}[\rho(\Lambda, z) \text{ is good}]$$

$$= \mathbb{E}_{\Lambda} \frac{\Pr_z[\mathcal{S}(z, P) \text{ is a transcript for } \rho(\Lambda, z)]}{\Pr_z[\mathcal{S}(z, P) \text{ is a transcript for } \rho(\Lambda, z) | \rho(\Lambda, z) \text{ is good}]}$$

$$\leq \frac{1}{\gamma} \mathbb{E}_{\Lambda} \Pr_{z}[\mathcal{S}(z, P) \text{ is a transcript for } \rho(\Lambda, z)$$

$$= \frac{1}{\gamma} \mathbb{E}_{z} \Pr_{\Lambda}[\mathcal{S}(z, P) \text{ is a transcript for } \rho(\Lambda, z)]$$

$$\leq p^t/\gamma. \tag{4}$$

Stringing Equations (1),(2), and (4) lets us bound

$$\Pr_{\rho}[\mathsf{DT}(F|_{\rho}) \geq t] \leq (p^t/\gamma) \cdot \#\{\text{partial transcripts } P\}$$

It turns out we can define our partial transcripts P and construct a transcript searcher such that the above term is small enough to give us the desired switching lemma. See Theorem 20 for the technical details.

1.2.1 Comparison to Lyu [18]

Although the proof structure for proving our switching lemma is similar to Lyu's [18] proof of the AC^0 switching lemma, some changes are necessary to accommodate the general structure of $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ circuits.

- We need to create a more complex CDT that can compute $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ circuits, and a corresponding new definition of witnesses/partial witnesses that records the transcript of the complex CDT so that our witness searcher can effectively reconstruct a transcript given information about ρ and a partial witness.
- Because our CDT contains more steps, there will naturally be more possible transcripts/witnesses. As the switching lemma hinges on a low quantity of possible partial witnesses to union bound over, we need to argue with our new CDT, the number of partial witnesses can be controlled by the parameter k. This makes designing the CDT and partial witness to be an act of balancing contrasting parameters

  - For example, the more complicated a CDT procedure is, the closer to the true decision tree depth it will reach (and hence a tighter bound on $\Pr_{\rho}[\mathsf{CDT}(F|_{\rho}) \geq t]$ can be expected), but the larger the possible number of transcripts it will have (thereby increasing the number of terms we union bound over). Therefore, this approach demands the designed CDT to be complicated enough to give a small depth decision tree with high probability, but simple enough to be tractable to analyze with a union bound.
  - Similarly, the more that a partial witness keeps track of, the larger amount of possible partial witnesses we will need to union bound over. However, if we keep track of too little, there will not exist an effective witness searcher that can use the information from the partial witness to construct the whole witness. Hence we need to keep track of just the right amount of information.

- In the argument for AC^0 circuits, one would show a multi-switching lemma on depth 2 AC^0 circuits. In other words, one would argue that a collection of AC^0_2 circuits simultaneously simplify after a one random restriction is applied to all of them. Rather than the natural idea of proving a switching lemma for the analogous $\mathsf{GC}^0_2(k)$ circuits, we consider the hybrid class of $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ circuits. It turns out a switching lemma on these simpler circuit classes suffice to depth reduce and prove bounds on $\mathsf{GC}^0_d(k)$ as we will see below.

1.3 The Depth Reduction Lemma

The multi-switching lemma gives a simplification lemma for depth 2 circuits. To extend this to constant depth circuits, we would like to iteratively decrease the depth of the circuit and induct. The argument for AC^0 circuits was quite simple. Say we have a depth 3 $\mathsf{OR} \circ \mathsf{AND} \circ \mathsf{OR}$ circuit F. Using the switching lemma, we can say with high probability, $F|_\rho$ is an $\mathsf{OR} \circ \mathsf{DT}_t$ circuit. We now expand each bottom layer decision tree into an $\mathsf{OR} \circ \mathsf{AND}_t$ circuit by enumerating over all 1-paths. Consequently this simplifies $F|_\rho$ to a $\mathsf{OR} \circ (\mathsf{OR} \circ \mathsf{AND}_t) = \mathsf{OR} \circ \mathsf{AND}_t$ circuit, since an OR of OR of variables is simply a single OR over all variables involved, getting us a depth reduction from depth 3 to 2.

What happens when we try the same argument for a k-$\mathsf{OR} \circ k$-$\mathsf{AND} \circ \mathsf{OR}$ circuit F? By our switching lemma, $F|_\rho$, with high probability will simplify to a k-$\mathsf{OR} \circ \mathsf{DT}_t$ circuit. We can then unravel the decision trees into $\mathsf{OR}_{2^t} \circ \mathsf{AND}_t$ CNFs, resulting in a k-$\mathsf{OR} \circ \mathsf{OR}_{2^t} \circ \mathsf{AND}_t$ circuit. Here, we reach an issue: a k-$\mathsf{OR} \circ \mathsf{OR}$ circuit is not necessarily itself a k-OR function! We could have up to $(k-1)2^t$ input bits of a k-$\mathsf{OR} \circ \mathsf{OR}_{2^t}$ be 1 while still evaluating to 0 (set all 2^t bits of $k-1$ of the bottom depth ORs to be 1). The best we can do is say the function is in $\mathsf{G}((k-1)2^t+1)$, which is too large of a blowup in the "k" parameter for our switching lemma to handle.

We rewind a bit to our k-$\mathsf{OR} \circ \mathsf{DT}_t$ circuit and unravel the bottom-layer trees to $\mathsf{OR}_{2^t} \circ \mathsf{AND}_t$ DNFs by enumerating over 1-paths. But we now make the key observation about each DNF which follows from the fact any assignment uniquely defines a path on a decision tree: *any assignment of x makes at most* 1 AND_t *clause true.* To use more standard terminology, the DNF created from decision trees is *unambiguous.* This means the pathological case above of all clauses under $k-1$ OR_{2^t} gates being satisfied cannot happen. In fact, we can prove something stronger. Since at most one clause under each OR_{2^t} gate can be satisfied in the unraveled k-$\mathsf{OR} \circ \mathsf{OR}_{2^t} \circ \mathsf{AND}_t$ circuit, the number of middle layer OR_{2^t} clauses that are satisfied will be precisely the total number of bottom layer AND_t clauses that are satisfied. Hence, a k-OR over the OR gates is exactly the same as a k-OR over the AND_t clauses themselves, and we can indeed collapse to a k-$\mathsf{OR} \circ \mathsf{AND}_t$ circuit! This gets us our depth reduction. A slightly more involved argument is carried out to show the more general $\mathsf{G}(k) \circ \mathsf{DT}_t$ circuit can be calculated by a $\mathsf{G}(k) \circ \mathsf{AND}_t$ circuit, but the heart of the argument is captured in the k-OR case itself.

1.4 Putting It All Together

We now have all the ingredients to simplify $\mathsf{GC}^0_d(k)$ circuits. The argument will be the following inductive process, where we are effectively inducting on circuits of the form $\mathsf{GC}^0_d(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ rather than GC^0_d directly. Given a $\mathsf{GC}^0_d(k)$ circuit,

1. Add a trivial $(d+1)$-st layer at the bottom that is simply the identity gate (think of it as an AND_1 gate)
2. By the multi-switching lemma, we know the depth 2 $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ subcircuits simplify to DT_t trees with high probability, resulting in a $\mathsf{GC}^0_{d-1}(k) \circ \mathsf{DT}_t$ circuit.

3. By the depth reduction lemma, each of the bottom depth 2 $\mathsf{G}(k) \circ \mathsf{DT}_t$ subcircuits can be calculated by a $\mathsf{G}(k) \circ \{\mathsf{AND}_t, \mathsf{OR}_t\}$ circuit, resulting in a $\mathsf{GC}^0_{d-1}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ circuit.
4. The depth has reduced by 1, so we go back to Step 2 and induct.

This argument allows us to use the multi-switching lemma along with the depth reduction lemma to establish size lower bounds for $\mathsf{GC}^0_d(k)$ bounds. We show a formal argument of this outline in Section 3.

2 Preliminaries

2.1 Notation

$[n] = \{1, 2, \ldots, n\}$ denotes the set of the first n positive integers. $\binom{[n]}{k}$ denotes the set of all size k subsets of $[n]$. log is assumed to be in base 2. This paper concerns *constant*-depth circuits, and so the depth variable, d, should be treated as a constant. In particular hidden constants in $O(\cdot)$ or $\Omega(\cdot)$ may depend on d. For $S \subset [n]$, we denote $x^S = \prod_{i \in S} x_i$.

2.2 Random Restrictions and Partial Assignments

A *partial assignment* or *restriction* is a string $\rho \in \{0, 1, \star\}^n$. Intuitively, a $\star$ represents an index that is still "alive" and hasn't been fixed to a value yet.

An alternative way of defining a restriction is by the set of alive variables and a "ground assignment" string. Given a "$\star$ set" Λ and a ground assignment $z \in \{0,1\}^n$, we define $\rho(\Lambda, z)$ to be the partial assignment where we assign

$$\rho(\Lambda, z)_i = \begin{cases} \star & i \in \Lambda \\ z_i & i \notin \Lambda \end{cases}$$

Sometimes, Λ may be in the form of an indicator $\{0,1\}^n$ string, where the set is defined to be the set of indices containing a 1.

We also define a composition operation on partial assignments. For two restrictions ρ^1, ρ^2, define $\rho^1 \circ \rho^2$ so that

$$(\rho^1 \circ \rho^2)_i = \begin{cases} \rho^1_i & \rho^1_i \neq \star \\ \rho^2_i & \rho^1_i = \star. \end{cases}$$

Intuitively, one can see this as fixing bits determined by ρ^1 first, and then out of the remaining alive positions, fix them according to ρ^2.

A *random restriction* is simply a distribution over restrictions. A common random restriction we will use is R_p, the distribution where each index will be assigned $\star$ with probability p, and $0, 1$ each with probability $\frac{1-p}{2}$.

The main reason for defining restrictions is to observe their action on functions. Given a restriction ρ and function $f : \{0,1\}^n \to \{0,1\}$, we define $f|_\rho : \{0,1\}^n \to \{0,1\}$ to be the function mapping $f|_\rho(x) := f(\rho \circ x)$.

2.3 Models of Computation

Circuits

We measure the size of a circuit by the total number of wires (including input wires) in it. We define the width of a DNF or CNF to be the maximum number of variables in any of its clauses. We also use k-DNF (resp. k-CNF) to denote DNF (resp. CNF) of width at most

k. AC^0_d are depth d circuits with unbounded fan-in whose gate set is $\{\mathsf{AND}, \mathsf{OR}, \mathsf{NOT}\}$. In general, if we have a gate G, a subscript G_k will refer to its fan-in (in this case, G is fixed to have fan-in k). We now define more general circuit classes that we analyze in this work.

▶ **Definition 9** (k-OR/k-AND/$\mathsf{AC}^0_d(k)$)**.** *Define k-$\mathsf{OR}_m : \{0,1\}^m \to \{0,1\}$ to be a function that evaluates to 1 iff x contains $\geq k$ ones. Analogously define k-AND_m to be 0 iff x contains $\geq k$ zeros. Define $\mathsf{AC}^0_d(k)$ to be the class of depth d circuits with unbounded fan-in whose gate set is $\{k'$-AND, k'-$\mathsf{OR}, \mathsf{NOT}\}$ for all $k' \leq k$.*

In more generality, we define $\mathsf{G}(k)$ gates and $\mathsf{GC}^0_d(k)$ circuits.

▶ **Definition 10** ($\mathsf{G}(k)/\mathsf{GC}^0(k)$)**.** *Define a gate set $\mathsf{G}(k)$ to be the set of all arbitrary fan-in gates such that they are constant on inputs with $\geq k$ ones (we call such gates* orlike*) or are constant on inputs with $\geq k$ zeros (we call such gates* andlike*). $\mathsf{GC}^0(k)$ is the class of constant depth circuits made by $\mathsf{G}(k)$ gates.*

In the rest of the paper, we may write circuit classes $\mathsf{GC}^0_d(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ or $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$. In the literature, this usually refers to the circuit class whose gates above the bottom layer are in $\mathsf{G}(k)$, and whose bottom layer gates can either be AND or OR with no restriction on the choice. However, in this paper, assume this notation implicitly restricts AND gates to only be under orlike $\mathsf{G}(k)$ gates and OR gates to only be under andlike $\mathsf{G}(k)$ gates.

On top of being an alternate generalization of AND/OR gates, $\mathsf{G}(k)$ gates capture arbitrary LTFs that are "unbalanced" in some sense. We will use the $\{\pm 1\}$ bits to define these, but one can convert between $\{0,1\}$ and $\{\pm 1\}$ via the map $b \to (-1)^b$.

▶ **Definition 11** (Balance of an $\mathsf{LTF}/\mathsf{TC}^0(k)$)**.** *Consider an arbitrary LTF $f : \{\pm 1\} \to \{\pm 1\}$ with $f(x) = sgn(\sum w_i x_i - \theta)$. Let $\sigma : [n] \to [n]$ be a permutation ordering (w_i) such that $|w_{\sigma(1)}| \leq \cdots \leq |w_{\sigma(n)}|$. Define the* balance *of f (denoted as $bal(f)$) to be the smallest integer k such that $-\sum_{i \leq k} |w_i| + \sum_{i>k} |w_i| < |\theta|$. Now denote $\mathsf{TC}^0(k)$ to be the class of constant depth circuits made out of THR gates with balance $\leq k$.*

We prove that up to negations in the inputs and output, THR gates with balance k are in $\mathsf{G}(k)$ in the appendix (Theorem 51). All results in this paper hold for $\mathsf{TC}^0(k)$, but from now on, we will only refer to $\mathsf{GC}^0(k)$ as it is the more general class.

Decision Trees

We assume knowledge of decision trees (see Definition 3.13 in [20] for a reference). We will be using slightly more complex models of decision trees in this work.

▶ **Definition 12** (Partial Decision Trees)**.** *For a collection of functions $\mathcal{F} = \{F_1, \ldots, F_m\}$, we say $\mathcal{F}$ can be computed by an r-partial depth-t DT if there exists a singe depth r tree such that for all F_i and paths π of T, $F_i|_\pi$ can be computed by a depth t decision tree (here, $F|_\pi$ is F acted on by the restriction induced by taking path π down T).*

▶ **Definition 13** ($(d, \mathcal{C})$-tree)**.** *Let d be an integer and $\mathcal{C}$ a computational model (e.g. a circuit class). A function is computable by a $(d, \mathcal{C})$-tree if it is computable by a depth t decision tree with $\mathcal{C}$ functions as its leaves. That is, there exists a depth d decision tree T such that for every path π in T, $F|_\pi \in \mathcal{C}$.*

2.4 Pseudorandomness and Probability

We will use various pseudorandom primitives and terminology. We will use U_n to denote the uniform distribution over n bits unless specified otherwise.

▶ **Definition 14** (ε-error PRG/Seed Length). *A distribution D over $\{0,1\}^n$ is called an ε-error PRG for a computational model $\mathcal{C}$ if for all $C \in \mathcal{C}$,*

$$|\mathbb{E}_{x\sim U_n}[C(x)] - \mathbb{E}_{x\sim D}[C(x)]| \leq \varepsilon$$

The seed length s of D is defined to be the minimal quantity s such that the following is true: there exists a polytime computable function $G : \{0,1\}^s \to \{0,1\}^n$ such that the distribution of $G(z)$ over $z \sim U_s$ is exactly D.

▶ **Definition 15** ((ε, k)-wise independent source). *A distribution D over $\{0,1\}^n$ is an (ε, k)-wise independent source if for all $1 \leq i_1 < \cdots < i_k \leq n$ and $\alpha \in \{0,1\}^k$,*

$$|\Pr_{x\sim D}[x_{i_1}x_{i_2}\ldots x_{i_k} = \alpha] - 2^{-k}| < \varepsilon.$$

There exists constructions of these sources with seed length $O(\log\log n + k + \log(1/\varepsilon))$ [3].

▶ **Definition 16** (k-wise Independent Hash Family). *Let $\mathcal{H}$ be a distribution over hash functions mapping $\{0,1\}^n \to \{0,1\}^m$. We say that $\mathcal{H}$ is k-wise independent if for any k input-output pairs $(x_1, y_1), \ldots, (x_k, y_k) \in \{0,1\}^n \times \{0,1\}^m$ where $x_1, \ldots, x_t$ are distinct, it holds that*

$$\Pr_{h\sim\mathcal{H}}[\forall i \in [k], h(x_i) = y_i] = 2^{-km}.$$

Such functions can be sampled using $O(k(n+m))$ bits (Chapter 3.5.5 of [32]).

▶ **Definition 17** (k-wise p-bounded Subset). *Let Λ be a random subset of $[n]$. Λ is a k-wise p-bounded subset iff for all subsets $S \subset [n]$ of size $\leq k$, $\Pr_\Lambda[S \subset \Lambda] \leq p^{|S|}$.*

For example, R_p is n-wise p-bounded.

2.5 Fourier Analysis

Every Boolean function $f : \{\pm 1\}^n \to \{\pm 1\}$ has a unique representation as a multilinear real polynomial

$$f(x) = \sum_{S\subset[n]} c_S x^S.$$

Given f, we can think of the Fourier transform of f, $\widehat{f}$ to be a function mapping $2^{[n]} \to \mathbb{R}$ such that $\widehat{f}(S) = c_S$. This is well defined by the uniqueness of the polynomial representation of S. One can explicitly compute $\hat{f}(S) = \mathbb{E}_x[f(x)x^S]$. By Parseval's, one can derive $\sum_{S\subset[n]} \widehat{f}(S)^2 = 1$. There are various quantities involving the Fourier coefficients that we will work with.

▶ **Definition 18** (Fourier Tails). *For a Boolean function f, define*

$$W^{\geq k}[f] := \sum_{|S|\geq k} \widehat{f}(S)^2.$$

▶ **Definition 19** (Discrete Derivative/Influence). *For Boolean f and $i \in [n]$, define the discrete derivative*

$$D_i f(x) = \frac{f(x^{(i\to 1)}) - f(x^{(i\to -1)})}{2}$$

where $x^{(i\to b)} = (x_1, \ldots, x_{i-1}, b, x_{i+1}, \ldots, x_n)$. Now for $S \subset [n]$ with $S = \{i_1, \ldots, i_k\}$, define

$$D_S f = D_{i_1} D_{i_2} \ldots D_{i_k}.$$

Now for $S \subset [n]$, define the influence

$$Inf_S(f) = \mathbb{E}_{x\sim\{\pm 1\}^n}[D_S f(x)^2].$$

Finally, define the degree k influence

$$Inf^k(f) = \sum_{|S|=k} Inf_S(f).$$

3 Simplification Theorem of $\mathsf{GC}^0_d(k)$ Circuits

▶ **Theorem 20.** *Let F be computable by a depth-2 $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuit. Let Λ be a $(t+w)$-wise p-bounded subset of $[n]$, and x a uniform string. Then*

$$\Pr_{\Lambda,x}[\mathsf{DT}(F|_{\rho(\Lambda,x)}) \geq t] \leq (20pw)^t 2^k.$$

Proof. The proof will follow that of Section 5 in [18]. We urge the reader to first read the overview given in Section 1.2. As discussed there, the main differences between this proof and the one presented there are present in the constructions of the canonical decision tree and witness searchers, the definition of witnesses, and the counting of partial witnesses. These are altered to support the more general $\mathsf{G}(k)$ gates. Besides this, the general proof strategy remains the same. Let m be the fan-in of the F. We present a procedure that constructs a decision tree (which we deem the "Canonical Decision Tree") in Algorithm 1.

The difference between the CDT defined in [18] and the one presented here is the use of ctr. Intuitively, this is added in to keep track of the number of satisfied clauses we see before we reach our limit of k. We rigorously prove this in the following claim.

▷ Claim 21. The CDT correctly outputs $F(\alpha)$.

Proof. The CDT scans the clauses in order to find the first one not fixed to zero. There are only two return statements in the algorithm, so we consider the two cases of terminating on each one. Suppose we terminate at the first return statement and output $F(1^m)$. Notice ctr is incremented each time the CDT encounters a satisfied clause. Therefore, when $ctr = k$, at least k C_i evaluate to 1, and therefore $F(C_1, \ldots, C_m) = F(1^m)$ by virtue of $F \in \mathsf{G}(k)$ being orlike, proving correctness. Now suppose we terminate after the while loop and output $C(x \circ 0)$. If CDT finishes the while loop without terminating, that must mean all clauses must be determined by the partial assignment x. This is because for any clause $C_{j'}$, if the clause wasn't already determined in the algorithm when $j^* = j'$, all unknowns of $C_{j'}$ would have been queried and fixed in the partial assignment x, thereby determining it. Therefore, in this case, $C(x)$ is determined, and in particular is equal to $C(x \circ 0^n)$. ◁

Algorithm 1 Canonical Decision Tree.

Input: (orlike $G(k)) \circ \mathsf{AND}_w$ circuit $F = G(C_1, \ldots, C_m)$, black-box access to a string $\alpha \in \{0,1\}^n$.
initialize:
  $j^* \leftarrow 0$
  $x \leftarrow (\star)^n$
  $ctr \leftarrow 0$
while $j^* < m$ **do**
  Find the first $j > j^*$ such that $C_j(x) \not\equiv 0$. If no such j exists, exit the loop.
  $B_j \leftarrow$ the set of unknown variables in $C_j(x)$ (may be empty).
  Query α_{B_j}.
  Set $x_{B_j} \leftarrow \alpha_{B_j}$.
  if $C_j(x) = 1$ **then**
    $ctr \leftarrow ctr + 1$;
    if $ctr = k$ **then**
      return $G(1^m)$
    end
  end
  $j^* \leftarrow j$
end
return $F(x \circ 0^n)$.

Therefore, CDT is indeed a decision tree computing F. Define $\mathsf{CDT}(F)$ to be the depth of the canonical decision tree T_F.If ρ is bad (i.e. $\mathsf{DT}(F|_\rho) \geq t$), then clearly $\mathsf{CDT}(F|_\rho) \geq \mathsf{DT}(F|_\rho) \geq t$ and so $T_{F|_\rho}$ will result in at least t queries for *some* choice of α (this is equivalent to saying that *some* path of $T_{F|_\rho}$ must have length $\geq t$). We define a witness that will effectively be the transcript of the algorithm on this particular α.

▶ **Definition 22.** *Let F be the circuit described above and ρ a restriction. Let $t \geq 1$. Consider the tuple $(r, \ell_i, s_i, B_i, \alpha_i)$ where*

- *$r \in [1, t+k]$ is an integer*
- *$(\ell_1, \ldots, \ell_r) \in [m]^r$ is an increasing list of indices*
- *$(s_1, \ldots, s_r)$ is a list of non-negative integers, at most k of which are allowed to be 0, such that $s := \sum_{i=1}^r s_i \in [t, t+w-1]$*
- *$(B_1, \ldots, B_r)$ is a list of (potentially empty) subsets of $[w]$ satisfying $|B_i| = s_i$.*
- *$(\alpha_1, \ldots, \alpha_r)$ is a list of (potentially empty) bit strings satisfying $|\alpha_i| = s_i$.*

$(r, \ell_i, s_i, B_i, \alpha_i)$ is called a t-witness for ρ if there exists an $\alpha \in \{0,1\}^n$ such that

- *When we run $T_{F|_\rho}$ on α, C_{ℓ_i} is the i-th term queried by $T_{F|_\rho}$.*
- *$T_{F|_\rho}$ queries s_i variables in C_{ℓ_i}, and the relative location of those variables within C_{ℓ_i} are specified by set B_i.*
- *$T_{F|_\rho}$ receives α_i in response to its i-th batch query.*

The size *of the witness $(r, \ell_i, s_i, B_i, \alpha_i)$ is defined to be $s := \sum_{i=1}^r s_i$. We may denote the size of a witness W as $\mathsf{size}(W)$.*

▷ Claim 23. For every ρ such that $\mathsf{DT}(F|_\rho) \geq t$, there exists a t-witness for ρ

Proof. We simply run $T_{F|_\rho}$ on α that causes at least t queries to be issued. We then record the transcript *until the number of variables queried exceeds* t, after which we halt. To be more explicit, we let r be the number of times the CDT stops at a clause before either outputting a bit or exeeding t queries. At the ith stop, say on clause C_j, we set $\ell_i = j$, s_i to be the number of unknown variables queried (which may be 0), B_i to be the subset of $[w]$ indicating the relative positions of the variables in the clause, and α_i being the query replies received from the black-boxed α. We can first verify this creates a valid tuple.

- $r \in [1, t+k]$. Every time we stop at a clause, either it evaluates to 1 and we increment ctr, or we have to query at least 1 variable. Since ctr can incremented at most k times and we can query variables from at most t clauses before reaching our quota of t queried variables, it follows we stop at most $t+k$ times.
- $(\ell_1, \ldots, \ell_r) \in [m]^r$ is an increasing list of indices since the CDT linearly sweeps the clauses in increasing index order.
- $(s_1, \ldots, s_r)$ is a list of non-negative integers, at most c of which are allowed to be 0, such that $s := \sum_{i=1}^{r} s_i \in [t, t+w-1]$. A particular s_i being zero implies that the CDT's ith stop was at a clause C_j that was already determined to be 1, and hence ctr was incremented. Since ctr can be incremented at most k times, at most k of the s_i's are zero. s is the total number of variables queried before halting. Notice after the penultimate clause is queried, there are $< t$ clauses queried. Consequently when the ultimate clause is queried, there clearly will be $< t + w$ variables queried, since k is the width of the clause. Of course, since the transcript halted after this clause, $\geq t$ variables had to be queried.
- $(B_1, \ldots, B_r)$ is a list of (potentially empty) subsets of $[w]$ satisfying $|B_i| = s_i$ trivially by construction.
- $(\alpha_1, \ldots, \alpha_r)$ is a list of (potentially empty) bit strings satisfying $|\alpha_i| = s_i$ trivially by construction.

We can then easily see by construction of the tuple, it is indeed a t-witness for ρ. ◁

We note that the difference between Definition 4 in [18] and the one here is the relaxation to allow $(s_1, \ldots, s_r)$ to contain up to k zeros, rather than to all be positive. As evident in the proof, this is to handle cases the CDT encounters a clause that was already fixed to 1, which causes the corresponding s_i value to be 0. This wasn't recorded in Lyu's witness definition because in the case of CNFs, one satisfied clause determines the value of the circuit, and the CDT immediately halts. Why do we still record that the CDT didn't query any variables at a clause instead of just ignoring this behavior and moving on? It turns out if we don't include this piece of information, the witness searcher we create will not have enough information to reconstruct the whole witness (see the "balancing act" discussion in Section 1.2.1).

We now move on to define partial witnesses.

▶ **Definition 24.** *Let F be a circuit and ρ a restriction. We call (r, s_i, B_i, α_i) a partial t-witness for ρ if there exists $(\ell_1, \ldots, \ell_r)$ such that $(r, \ell_i, s_i, B_i, \alpha_i)$ is a t-witness for ρ.*

We note the following important claim.

▷ Claim 25. If P is a partial witness for ρ, then there exists exactly one list of integers (ℓ_i) such that (ℓ_i, P) is a witness for ρ.

Proof. By construction of Algorithm 1, ℓ_1 must be be the index of the first clause not fixed to 0 by ρ. But now, we notice ℓ_2 must be the index of the first clause after C_{ℓ_1} not fixed to 0 by $\rho \circ \alpha_1$. We then continue this induction to get our unique list (ℓ_i), Where ℓ_j will be forced to be the index of the first clause after $C_{\ell_{j-1}}$ that is not fixed to 0. ◁

Therefore, by Claims 23 and 25

$$\Pr_{\rho}[\mathsf{DT}(F|_\rho) \geq t] \leq \sum_{(\ell_i,P)} \Pr_{\rho}[(\ell_i, P) \text{ is a } t\text{-witness for } \rho]$$
$$\leq \sum_{P} \Pr_{\rho}[P \text{ is a partial } t\text{-witness for } \rho]] \tag{5}$$

where P ranges over all partial t-witness tuples.

Going back to our proof, we now define our *witness searcher* $\mathcal{S}$ as Algorithm 2.

Algorithm 2 Witness Searcher $\mathcal{S}$.

Input: (orlike $\mathsf{G}(k)) \circ \mathsf{AND}_w$ circuit $F(C_1, \ldots, C_m)$, ground assignment $z \in \{0,1\}^n$, partial witness $W = (r, s_i, B_i, \alpha_i)$.
initialize:
  $j^* \leftarrow 0$
  $x \leftarrow (\star)^n$
  $ctr \leftarrow 1$
while $ctr \leq r$ **do**
  while $j^* < m$ **do**
    Find the first $j > j^*$ such that $C_j(z) \equiv 1$. If no such j exists, exit the inner while loop.
    $\ell_{ctr} \leftarrow j$
    Query α_{B_j}.
    Set the B_{ctr} portion of z to be α_{ctr}.
    $ctr \leftarrow ctr + 1$
    $j^* \leftarrow j$
  end
end
return (ℓ_i, W)

We now prove the following essential property about $\mathcal{S}$, stating in a probabilistic way, it can use a partial witness to reconstruct a total witness for ρ.

▶ **Lemma 26.** *Let P be a partial witness, and let s be its size. Define a restriction ρ to be* good *for P if P is a partial t-witness for it.*

$$\Pr_{z}[\mathcal{S}(z, P) \text{ is a } t\text{-witness for } \rho(\Lambda, z) | \rho(\Lambda, z) \text{ is good for } P] = 2^{-s}$$

Furthermore this event is solely dependent on z_I, where I is the set of variable indices referred to by the unique completion of P with respect to ρ.

Proof. If $\rho = \rho(\Lambda, z)$ is good, then by Claim 25 we know there exists unique ℓ_i such that (ℓ_i, P) witnesses ρ. In particular, we know that Λ must contain all the indices I that (ℓ_i, B_i) identify. Let I_j be the index set identified by ℓ_j and B_j (so $I = I_1 \sqcup \cdots \sqcup I_r$). Now condition on a fixed ρ. This means all bits in z not covered by Λ are fixed. In particular, only source of randomness left are the bits covered by Λ, which is a superstring of z_I. We now claim every z_{I_j} is assigned the unique bit string such that $C_{\ell_i}|_{z_{I_j}} \not\equiv 0$ (not forced to be unsatisfied) iff $\mathcal{S}$ successfully outputs (ℓ_i, P). This consequently proves the lemma, since this has probability $2^{-|I|} = 2^{-s}$ of happening.

By construction of $T_{F|_\rho}$ we know that all clauses before C_{ℓ_1} was falsified by ρ. Upon inspection, we see $\mathcal{S}$ correctly skips past these clauses (as z is a completion of ρ). Now we note C_{ℓ_1} was not fixed to 0 ρ, causing $T_{F|_\rho}$ to query all unknowns in C_{ℓ_1} at the time (which

might be nothing if C_{ℓ_1} was fixed to 1), which is x_{I_1}. Inspecting $\mathcal{S}$, we see $\mathcal{S}$ will set the correct ℓ_1 iff $C_{\ell_1}(z)$ is satisfied iff z_{I_1} is assigned the unique string such that $C_{\ell_i}|_{z_{I_j}} \not\equiv 0$ (since all variables outside I_1 ocurring in C_{ℓ_1} is fixed by ρ). $\mathcal{S}$ then (importantly) replace z_{I_1} with α_{I_1} so that all variables encountered thus far are assigned exactly as $T_{F|_\rho}$ did.

We then repeat this argument r times, noting that due to $z \circ \alpha_{I_1} \circ \cdots \circ \alpha_{I_j}$ being a completion of $\rho \circ \alpha_{I_1} \circ \cdots \alpha_{I_j}$, $\mathcal{S}$ rightfully skips all clauses between C_{ℓ_j} and $C_{\ell_{j+1}}$. We then similarly argue that $\mathcal{S}$ will set ℓ_{j+1} to be the $j+1$st clause $T_{F|_\rho}$ queries iff $z_{I_{j+1}}$ is the unique string such that $C_{\ell_{j+1}}|_{z_{I_{j+1}}} \not\equiv 0$. ◀

Combining Equation (5) and Lemma 26, it follows that

$$\begin{aligned}\Pr_\rho[\mathsf{DT}(F|_\rho) \geq t] &\leq \sum_P \Pr_\rho[P \text{ is a partial } t\text{-witness for } \rho] \\ &\leq \sum_P \mathbb{E}_\Lambda \frac{\Pr_z[\mathcal{S}(z,P) \text{ is a } t\text{-witness for } \rho(\Lambda,z)]}{\Pr_z[\mathcal{S}(z,P) \text{ is a } t\text{-witness for } \rho(\Lambda,z) | \rho(\Lambda,z) \text{ is good}]} \\ &\leq \sum_P 2^{\mathsf{size}(P)} \mathbb{E}_z \Pr_\Lambda[\mathcal{S}(z,P) \text{ is a } t\text{-witness for } \rho(\Lambda,z)] \qquad (6)\end{aligned}$$

Notice a necessary condition for a restriction $\rho(\Lambda, z)$ to be t-witnessed by a size-s W is for Λ to cover the s variables that W recorded as the CDT needing to query, which happens with probability $\leq p^s$ (as $s \leq t + w$ and Λ is $(t+w)$-wise p-bounded). Hence, every term in the sum in (6) can be bounded by $p^{\mathsf{size}(P)}$ and it remains to find the number of partial t-witness tuples P.

For a fixed s, we can bound the number of potential partial witnesses naively by noting

- the number of choices of (r, s_i) can be bounded by the number of ways to write s as the sum of at most $s+k$ nonnegative integers, which is $\sum_{r=1}^{s+k} \binom{s+r}{r} \leq \sum_{r=1}^{s+k} \binom{2s+k}{r} \leq 2^{2s+k}$ (notice that we get a larger count here than the analogous quantity of 2^{2s} in Lyu's proof [18], which is a side effect of looking at a more complicated circuit class),
- the choices for (B_i) can be bounded by $\prod_i \binom{w}{s_i} \leq w^s$,
- and the choices for (α_i) can be bounded by 2^s,

giving a total count of $(8w)^s 2^k$. Combining this count with the previous paragraph's observation and (6), while remembering to sum over all sizes, we derive

$$\Pr_\rho[\mathsf{DT}(F|_\rho) \geq t] \leq \sum_{s=t}^{t+w-1} (2p)^s (8w)^s 2^k = \sum_{s=t}^{t+w-1} (16pw)^s 2^k \leq (20pw)^s 2^k. \quad ◀$$

▶ Remark 27. One may ask whether the failure probability of $(20pw)^t 2^k$ tight. We show that PAR_{kw} can be expressed as a $\mathsf{G}(k) \circ \mathsf{AND}_w$ cirucit and prove this saturates the above bound in the Appendix (Theorem 52).

After defining witnesses, partial witnesses, and witness searchers for $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}$ circuits, we notice that Lyu's proof of the multi-switching lemma directly goes through with these definitions with zero changes (even down to the exact algorithm of the canonical partial decision tree and global witness searcher). Due to this, we defer the proof of the multi-switching lemma to the appendix. However, we do highlight here what properties about the circuit class is needed in order to invoke Lyu's lift from a switching lemma to a multi-switching lemma. The key properties needed were that

- the number of partial witnesses for a depth t canonical decision tree needed to be small, and
- there needed to exist a witness searcher function $\mathcal{S}$ such that for all ρ and a partial witness for ρ, $\mathcal{S}$ recovers the full witness with decent probability over an advice string.

- given a complete witness, there needed to be a small chance that a random restriction witnessed it

▶ **Theorem 28.** *Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of $\mathsf{G}(k) \circ \mathsf{AND}_w$ circuits on $\{0,1\}^m$. Then*

$$\Pr_{\rho \sim R_p}[\mathcal{F}|_\rho \text{ does not have } r\text{-partial depth-}t\ \mathsf{DT}] \leq 4(64(2^k m)^{1/r} pw)^t$$

Proof. See Theorem 56. ◀

▶ Remark 29. At this point, we can observe an aspect of the expression that illuminates an important unifying flavor of the rest of the results. Notice that the only difference in the failure probability expression between the standard AC^0 multi-switching lemma in (lyu) and the above one for $\mathsf{G}(k)$ is that every occurrence of m is multiplied by a factor of 2^k. This means if constants in the exponent can be ignored, the multi-switching lemma asymptotically gives the same result as the AC^0 version if $2^k \approx m$. In particular, we can expect any result for size s AC^0 circuits to immediately extend to $\mathsf{GC}^0(\log s)$ circuits with no loss in parameters! This will be demonstrated in various settings in future sections.

With our multi-switching lemma in hand, we can simplify depth 2 circuits with high probability. To extend this to constant depth circuits, we also require a depth reduction lemma. In the case of AC^0, this was trivial enough to embed in the main proof, but in the case of $\mathsf{GC}^0(k)$, we need to be more delicate and use more specific properties of decision trees.

▶ **Lemma 30.** *Any depth 2 circuit of the form $\mathsf{G}(k) \circ \mathsf{DT}_w$ with top gate fan-in m can be expressed as a circuit in $G(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ of size $m2^w$.*

Proof. Say the circuit we start with is $F(D_1, \ldots, D_m)$, where D_i are the bottom layer depth w decision trees. Assume F is orlike (the andlike case is analogous). By enumerating over all 1-paths, expand out each D_i as an OR of ANDs, namely $C_1^i \vee C_2^i \vee \cdots \vee C_{2^w}^i$. Now define a function F' over $m2^w$ bits, where

$$F'(x_1^1, \ldots x_{2^w}^1, x_1^2, \ldots, x_{2^w}^m) = \begin{cases} F(1^m) & \sum_{i,j} x_j^i \geq k \\ F(\bigvee_{i=1}^{2^w} x_j^1, \ldots, \bigvee_{i=1}^{2^w} x_j^m) & \text{otherwise} \end{cases}$$

Clearly by construction, $F' \in \mathsf{G}(k)$. Therefore to prove the lemma, it suffices to show that over all input assignments, $F(D_1, \ldots, D_m) = F'(C_1^1, \ldots C_{2^w}^1, C_1^2, \ldots, C_{2^w}^m)$.

If $\geq k$ of the D_i are satisfied, we know since F is an orlike $\mathsf{G}(k)$ function, $F(D_1, \ldots, D_m) = F(1^m)$. This also clearly implies $\geq k$ of the C_j^i are satisfied. Therefore by construction of F', $F'(C_1^1, \ldots, C_{2^w}^m)$ also evaluates to $F(1^m)$.

If $< k$ of the D_i are satisfied, then we need to use the following observation. For any assignment of inputs, at most one of the clauses $C_1^i, \ldots, C_{2^m}^i$ can be satisfied for each i, since each assignment uniquely defines a path in a decision tree. In more conventional terms, the DNF created by the decision tree D_i is *unambiguous*. Therefore the amount of D_i satisfied is exactly equal to the number of C_j^i satisfied, and so $< k$ clauses C_i^j are satisfied. This forces us into the second case of the piecewise definition of F', and so

$$F'(C_1^1, \ldots, C_{2^w}^m) = F(\bigvee_{i=1}^{2^w} C_j^1, \ldots, \bigvee_{i=1}^{2^w} C_j^m) = F(D_1, \ldots, D_m)$$

as desired. ◀

We have finally built up the tools to prove our main result: a constant depth simplification lemma.

▶ **Theorem 31.** *Let G be any gate, and let F be a $G \circ \mathsf{GC}^0_d(k)$ circuit of size m. Then for $p = \frac{1}{128(m2^k)^{1/w}}(128w(m2^k)^{1/w})^{-d+1}$ and any $t \geq 1$,*

$$\Pr_{\rho \sim R_p}[F|_\rho \text{ is not computed by a } ((2^d-1)t, G \circ \mathsf{DT}_w)\text{-decision tree}] \leq 4d \cdot 2^{-t}$$

Proof. WLOG assume the circuit is layered (all paths down the circuit are of length exactly $d+1$). We first append an extra layer of $\{\mathsf{AND}, \mathsf{OR}\}_1$ gates to the bottom of the circuit so that the input level fan-in is 1. We then apply a random restriction $\rho_0 \sim R_{p_0}$ with $p_0 = \frac{1}{128(m2^k)^{1/w}}$ and use Theorem 28 on all the depth-2 subcircuits to deduce that

$$\Pr_{\rho_0 \sim R_{p_0}}[F|_\rho \text{ is not computed by } (t, G \circ \mathsf{GC}^0_{d-1}(k) \circ \mathsf{DT}_w)\text{-decision tree}] \leq 4 \cdot 2^{-t}.$$

Letting $F^{(0)}$ be a good tree from above which does simplify, we see that there are at most 2^t leaves of the partial decision tree, with each leaf containing a $G \circ \mathsf{GC}^0(k)_{d-1} \circ \mathsf{DT}_w$ circuit (which we will refer to as "leaf-circuits"). By Lemma 30, these circuits can be simplified to $G \circ \mathsf{GC}^0_{d-1}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuits. We apply Theorem 28 on the depth-2 subcircuits of a particular leaf-circuit with $p_1 = \frac{1}{128w(m2^k)^{1/w}}$, using $2t$ instead of t, union bound over all 2^t leaves, and then apply Lemma 30 to get that

$$\begin{aligned}\Pr_{\rho_1 \sim R_{p_1}}[F^{(0)}|_{\rho_1} \text{ is not a } (t+2t, G \circ \mathsf{GC}^0_{d-2}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w\text{-decision tree}] &\leq 4 \cdot 2^{-2t} \cdot 2^t \\ &= 4 \cdot 2^{-t}.\end{aligned}$$

Iterating this argument $d-2$ more times, where we apply Theorem 28 on the depth 2 subcircuits using $p_i = \frac{1}{128w(m2^k)^{1/w}}$ and $2^i t$ instead of t on the ith iteration, and then union bound over all $2^{(2^i-1)t}$ leaves, we get that on the ith iteration, our desired single depth simplification happens with probability $2 \cdot 2^{-t}$. If the desired simplifications happen on all iterations, we result in a $((2^d-1)t, G \circ \mathsf{DT}_w)$-decision tree with probability at most $\sum_{i=0}^{d-1} 4 \cdot 2^{-t} = 4d \cdot 2^{-t}$ (via a union bound over the d iterations) and with a restriction from R_p where $p = \prod p_i = \frac{1}{128(m2^k)^{1/w}} \cdot (128w(m2^k)^{1/w})^{-d+1}$. The conclusion follows. ◀

With this theorem, we can let G be a $\mathsf{G}(k)$ gate to get the following corollary.

▶ **Corollary 32.** *Let C be a $\mathsf{GC}^0_d(k)$ circuit of size m and let $p = \frac{1}{40(128(k+\log m))^{d-1}}$. Then*

$$\Pr_{\rho \sim R_p}[\mathsf{DT}(C|_\rho) \geq t] \leq 2 \cdot 2^{-\frac{t}{2^{d-1}}+k}$$

Proof. Applying Theorem 31 to C with $w = k + \log m$, it follows for $p_1 = \frac{1}{128(128(k+\log m))^{d-2}}$,

$$\Pr_{\rho \sim R_{p_1}}[C|_\rho \text{ is not computed by a } ((2^{d-1}-1)t, \mathsf{G}(k) \circ \mathsf{DT}_{k+\log m})\text{-decision tree}] \leq 4d \cdot 2^{-t}.$$

Fix a ρ such that C simplifies to such a tree, T. By Lemma 30, the leaf circuits simplify to $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_{k+\log m}$ circuits. Let ℓ be a leaf, and let C_ℓ be the associated leaf-circuit. By Theorem 20, we know that for $p_2 = 1/40w$, $\Pr_{\tau \sim R_{p_2}}[\mathsf{DT}(C_\ell|_\tau) \geq 2^{d-1}t] \leq 2^{-2^{d-1}t}2^k$. Union bounding over all $\leq 2^{(2^d-1)t}$ leaves ℓ, it follows that

$$\begin{aligned}\Pr_{\rho \sim R_{p_1}, \tau \sim R_{p_2}}&[C|_{\rho\circ\tau} \text{ is not computed by a } ((2^{d-1}-1)t, \mathsf{DT}_{2^{d-1}t})\text{-decision tree}] \\ &\leq 2^{(2^d-1)t} \cdot 2^{-2^{d-1}t}2^k + 4d \cdot 2^{-t} \\ &\leq 2 \cdot 2^{-t+k}. \qquad (7)\end{aligned}$$

Because $\rho \circ \tau \sim R_{p_1 p_2}$, $p := p_1 p_2 = \frac{1}{40(128(k+\log m))^{d-1}}$, and a $((2^{d-1}-1)t, \mathsf{DT}_{2^{d-1}t})$-tree is simply a DT_{2^d-1}, (7) implies

$$\Pr_{\rho \sim R_p}[\mathsf{DT}(C|_\rho) \geq (2^d-1)t] \leq 2 \cdot 2^{-t+k}.$$

The desired result then follows after a change of variables from $t \to t/(2^d-1)$. ◀

4 Applications of The $\mathsf{GC}^0(k)$ Simplification Theorem

4.1 Exponential Lower Bounds Against Parity

Given Theorem 20 and Corollary 32, we can establish correlation bounds of $\mathsf{GC}^0_d(k)$ circuits against PAR (parity).

▶ **Theorem 33.** *Let $C \in \mathsf{GC}^0_d(k)$ have size m and let PAR be the parity function. Then the correlation of C and PAR is*

$$\mathbb{E}_{x \sim \{0,1\}^n}[(-1)^{C(x)+\mathsf{PAR}(x)}] \leq 2^{-\Omega(n/(k+\log m)^{d-1})+k}.$$

Proof. The uniform distribution is equivalent to performing a fair random restriction (a random restriction where non-star variables are set to a uniform bit), and then filling in the $\star$s with uniform bits. We will show that under a fair random restriction, C will become constant with high probability while PAR becomes a parity over the live variables. Averaging over these live variables then gives a correlation of zero. The total correlation is then the probability C doesn't become constant.

Let $p = \frac{1}{40(128(k+\log m))^{d-1}}$. Applying Corollary 32, we see that

$$\Pr_{\rho \sim R_p}[\mathsf{DT}(C|_\rho) \geq pn/4] \geq 2 \cdot 2^{-\frac{pn}{4(2^d-1)}+k}.$$

By a Chernoff bound, we know ρ will have $\geq pn/2$ stars with $\geq 1 - 2^{-pn/8}$ probability. Let $\mathcal{E}$ be the event both of these events happen, and fix such a ρ. Consider performing a random walk down the depth $\leq pn/4$ decision tree (start at the root and iteratively pick which of the 2 children to travel to uniformly, effectively filling in $\leq pn/4$ of the variables with uniform bits), which induces a random restriction τ. No matter which path restriction τ was taken, $C|_{\rho\circ\tau}$ becomes constant, while $\mathsf{PAR}|_{\rho\circ\tau}$ becomes a parity over $\geq pn/2 - pn/4 = pn/4$ variables. The correlation of these two functions is trivially 0. Therefore,

$$\begin{aligned}
&\mathbb{E}_{x \sim \{0,1\}^n}[(-1)^{C(x)+\mathsf{PAR}(x)}] \\
&\qquad = |\mathbb{E}_\rho \mathbb{E}_x[(-1)^{C|_\rho(x)+\mathsf{PAR}|_\rho(x)}]| \\
&\qquad \leq \Pr[\neg\mathcal{E}] + \mathbb{E}_\rho[|\mathbb{E}_x[(-1)^{C|_\rho(x)+\mathsf{PAR}|_\rho(x)}]|\big|\mathcal{E}] \\
&\qquad \leq 2 \cdot 2^{-\frac{pn}{4(2^d-1)}+k} + 2^{-pn/8}\mathbb{E}_\rho[\mathbb{E}_\tau|\mathbb{E}_x[(-1)^{C|_{\rho\circ\tau}(x)+\mathsf{PAR}|_{\rho\circ\tau}(x)}]|\big|\mathcal{E}] \\
&\qquad \leq 2^{-\Omega(n/(k+\log m)^{d-1})+k}.
\end{aligned}$$

◀

As an application, we can observe that for $0 \leq k \leq .1n^{1/d}$ one can set $m = 2^{\Theta((n/k)^{\frac{1}{d-1}})}$ in the above lemma such that the correlation is $< 1/2$, yielding us the following corollary.

▶ **Corollary 34.** *For some absolute constant C, integer d, and $0 \leq k \leq .1n^{1/d}$, $\mathsf{GC}^0_d(k)$ circuits computing PAR_n requires size $2^{\Omega((n/k)^{\frac{1}{d-1}})}$.*

This is an interesting result in multiple ways. First, notice the dependence of the lower bound on the "k" parameter is extremely tight and the corollary becomes absurdly false if $k = n^{1/d}$. This is seen by the fact $\mathsf{PAR}_{n^{1/d}} \in \mathsf{G}(n^{1/d})$, and so one can create a size $O(n^{1-1/d})$ $\mathsf{GC}^0_d(n^{1/d})$ formula computing PAR_n simply by having the ith depth from the bottom have $n^{1-i/d}$ $\mathsf{PAR}_{n^{1/d}}$ gates, each of which takes in inputs from $n^{1/d}$ gates below it. Hence we observe a "sharp threshold" behavior where a difference in constants can change an exponential lower bound to a sublinear one.

We also observe that this lower bound almost matches the classic $2^{\Omega(n^{\frac{1}{d-1}})}$ construction for AC^0_d circuits calculating parity. Thus, augmenting AC^0 with unbounded fan-in gates which have the power to calculate the majority of polynomially many bits has *no effect* on its ability to calculate parity, even thought we know such gates require exponentially sized AC^0_d circuits. In fact, by an argument resembling Shannon's classic circuit lower bound, there exists gates in $\mathsf{G}(k)$ which require size $2^{\Omega(n^{1/2d})}$ AC^0_d circuits.

We can also show that the size lower bound in Corollary 34 and the correlation bound in Theorem 33 is tight. In particular, the gap between the $2^{\Omega(n^{\frac{1}{d-1}})}$ lower bound for AC^0 and $2^{\Omega(n^{\frac{1}{d}})}$ bound established for $\mathsf{GC}^0(.1n^{1/d})$ cannot be bridged. We defer the formal proofs to Appendix A.2.

4.2 Correlation Bounds for $\mathsf{GC}^0(k)$ Circuits With Few Arbitrary Threshold Gates

In this section, we prove that state of the art correlation bounds against AC^0 circuits [27] with a small number of threshold gates extends to if we instead start with $\mathsf{GC}^0(\log^2 n)$ circuits. We first give an overview of their proof. As in previous works studying this correlation [23, 33, 17, 27], the hard function we uncorrelate with is

$$\mathsf{RW}_{m,k,r}(x) = \bigoplus_{i=1}^{m} \bigwedge_{j=1}^{k} \bigoplus_{\ell=1}^{r} x_{ijk}$$

A uniform string can be sampled by performing a random restriction and then filling the $\star$s with uniform bits. Driven by this, the overlying strategy is to apply a random restriction, and show the circuit collapses while the RW function maintains integrity. It turns out because our multi-switching lemma gives no loss in parameters (up to constants), we can apply the exact same argument in [27], except replace the AC^0 simplification lemma (Corollary 3.2 of [27]) with our more general Theorem 31.

▶ **Theorem 35.** *Fix u. Let $v = .005 \log n$ and $q = \sqrt{n/(v+1)}$ There exists a function $\mathsf{RW}_{q,v,q} \in \mathsf{P}$ and small enough constant τ such that for all circuits $\mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{GC}^0_d(\Omega(\log^2 n))$ circuits F where each of the u $\mathsf{THR} \circ \mathsf{GC}^0_d$ subcircuits of F has size at most $s = n^{\tau \log n}$, we have*

$$|\mathbb{E}_{x \sim U_n}[(-1)^{\mathsf{RW}(x)+C(x)}] \leq 2^{-\Omega(n^{.499}/u)}$$

Proof. We immediately apply Theorem 31 to F with G being the top $\mathsf{ANY}_u \circ \mathsf{THR}$ circuit, $m = u \cdot 2^{(\varepsilon/100d)\log^2 n}, w = \varepsilon \log m, t = q/2$ and $k = (\varepsilon/100d)\log^2 n$ to get that for $\rho' \sim R_p$, where $p = n^{-\varepsilon/50}$,

$$\Pr[F|_{\rho'} \text{ is an } (m/2, \mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{DT}_w)\text{-decison tree}] \leq 1 - 4d \cdot 2^{-q/2}.$$

This computational model is now void of $\mathsf{G}(k)$ gates, and we can essentially port in the rest of [27] to finish. By the "Second Step" and "Third Step" under Section 2 of [27], one can compose ρ' with another restriction to get a final random restriction ρ that simplifies the tree further and prunes the fan-in to an $\mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{AND}_v$ circuit.

It was shown in Lemma 4.3 of [27] that the same random restriction ρ will have $\mathsf{RW}|_\rho$ equal(after restricting additional bits and negating input bits and/or the output) $\mathsf{GIP}_{q/2,v+1}$ except with probability $2^{-\tilde{\Omega}(pq)}$. Theorem 21 in [27] then states that an $\mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{AND}_v$ circuits can be calculated by a randomized NOF $(v+1)$-party protocol with error $\gamma = 2^{-q^{.99}/u}$ using $O(uv^3 \log n \log(n/\gamma)) = O(q^{.99}v^3 \log n)$ bits. Finally, by Theorem 14 in [27], we can conclude the correlation between $\mathsf{GIP}_{q/2,v+1}$ and $\mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{AND}_v$ is at most $2^{-\Omega(q^{.99}/u)}$. Hence the overall correlation can be bounded, via union bound, by the sum of the error probabilities and the correlation of $\mathsf{GIP}_{q/2,v+1}$ and $\mathsf{ANY}_u \circ \mathsf{THR} \circ \mathsf{AND}_v$, yielding

$$|\mathbb{E}_{x\sim U_n}[(-1)^{\mathsf{RW}(x)+C(x)}] \leq 4d \cdot 2^{-q/2} + 2^{-\tilde{\Omega}(pq)} + 2^{-\Omega(q^{.99}/u)} = 2^{-\Omega(n^{.49}/u)}$$

as desired. ◀

With this theorem, we can prove the actual correlation bound for $\mathsf{GC}^0(k)$ circuits with arbitrary gates.

▶ **Theorem 36.** *Let C be a $\mathsf{GC}^0(\Omega(\log^2 n))$ circuit, g of whose gates are arbitrary THR gates. Then*

$$\mathbb{E}[(-1)^{C(x)+\mathsf{RW}(x)}] \leq 2^{-\Omega(\frac{n^{.499}}{g}-g)}.$$

In particular, plugging in $g = \Theta(n^{.249})$ tells us

$$\mathbb{E}[(-1)^{C(x)+\mathsf{RW}(x)}] \leq 2^{-\Omega(n^{.249})}$$

Proof. This follows from Theorem 35 exactly like how Theorem 3 follows from Lemma 6 in [17]. ◀

▶ Remark 37. We note that an argument analogous to the above can be used to show $2^{-\Omega(n^{.499})}$ correlation bounds against $\mathsf{GC}^0(\Omega(\log^2 n))$ circuits with $n^{.499}$ gates, via the same argument presented in [27].

It is worth noting that if we had tried performing this argument by expanding the size $n^{\Omega(\log n)}$ $\mathsf{GC}^0(\log^2 n)$ circuit naively into an AC^0 circuit, not only would we get a loss in parameters, but the argument *will not go through*. The proof crucially relied on correlation bounds against $v = .005 \log n$ party protocols. Had we asymptotically increased the size of our circuit by writing it as an AC^0 circuit, then after applying random restrictions to prune the fan-in of our circuit, we will be left with trying to uncorrelate against arbitrary $\omega(\log n)$-party protocols, a longstanding open problem (Problem 6.21 in [14]).

4.3 Derandomizing the Multi-Switching Lemma and PRGs for $\mathsf{GC}^0(k)$

Using the same techniques appearing in [12, 18], we can completely derandomize our switching and multi-switching lemma. We defer the proof to the appendix (Theorem 62).

▶ **Theorem 38.** *Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of size m $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuits. Let (Λ, z) be a joint random variable such that*

- *Λ is a $(t+w)$-wise p-bounded subset of $[n]$*
- *Conditioned on any instance of Λ, z ε-fools CNF of size $\leq m^2$.*

Then

$$\Pr_{\Lambda,z}[\mathcal{F}|_{\rho(\Lambda,x)} \text{ has no } r\text{-partial depth-}t\ \mathsf{DT}] \leq 4(m2^k)^{t/r}(64pw)^t + (64wm)^{t+w}(2m)^{2kt/r} \cdot \varepsilon$$

Proof. See the Theorem 62 in the appendix. ◀

Using the derandomized multi-switching lemma, we can use the partition-based template in order to create PRGs for $\mathsf{GC}^0(k)$. The arument to reduce any constant depth to depth 2 will be a very similar argument. [18] simply uses a CNF PRG to tackle the base case of AC^0_2 circuits, but we cannot do so with a $\mathsf{G}(k) \circ \mathsf{AND}_w$ circuit unless we want to expand it out as a CNF and incur a multiplicative k loss in our seed length. We instead use the derandomized switching lemma one more time to simplify $\mathsf{GC}(k) \circ \mathsf{AND}_w$ to a $\mathsf{DT}_{\log m}$ and then fool this with an $(\varepsilon/m, \log m)$-wise independent seed. We quickly prove this latter statement in the following lemma.

▶ **Lemma 39** (PRG for Depth t Decision Trees). *There exists a ε-error PRG with $O(\log\log n + t + \log(1/\varepsilon))$ seed length for DT_t.*

Proof. Let D be a $(\varepsilon/2^t, t)$-wise independent distribution, samplable using $O(\log\log n + t + \log(2^t/\varepsilon)) = O(\log n + t + \log(1/\varepsilon))$ bits. For arbitrary $T \in \mathsf{DT}_t$, label the leaves $L_1, \ldots, L_{2^t}$, and let the value of leaf L_i be ℓ_i. Then

$$T(x) = \sum_{j=1}^{2^t} \ell_j \cdot \mathbb{1}(T(x) \text{ reaches } L_j).$$

Note that $\ell_j \cdot \mathbb{1}(T(x) \text{ reaches } L_j)$ depends on at most t bits, and so D will $\varepsilon/2^t$-fool it. Therefore,

$$\begin{aligned}
&|\mathbb{E}_{x\sim U_n}[T(x)] - \mathbb{E}_{x\sim D}[T(x)]| \\
&\quad\leq \sum_{j=1}^{2^t} |\mathbb{E}_{x\sim U}[\ell_j \cdot \mathbb{1}(T(x) \text{ reaches } L_j)] - \mathbb{E}_{x\sim D}[\ell_j \cdot \mathbb{1}(T(x) \text{ reaches } L_j)]| \\
&\quad\leq \sum_{j=1}^{2^t} \varepsilon/2^t \\
&\quad= \varepsilon.
\end{aligned}$$

◀

With this, we are now ready to prove our final PRG for $\mathsf{GC}^0(\log m)$.

▶ **Theorem 40.** *For $m, n \in \mathbb{N}$ and $w \leq \log m$, there is an ε-error PRG with $O((w\log^{d-1}(m) + \log^2(m))\log(m/\varepsilon)\log\log m)$ seed length for $\mathsf{GC}^0_d(\log m) \circ \mathsf{AND}_w$ circuits.*

Proof. Let $\ell = 512w$ and $t = 10\log(m/\varepsilon)$.

- Let $H : [n] \to [\ell]$ be a $2t$-wise independent hash function which needs $O(t\log n) = O(\log n \log(m/\varepsilon))$ bits. We will let H_i be an n-bit string such that $(H_i)_j = 1$ iff $H(j) = i$.
- Let $\varepsilon' = \varepsilon/(\ell \cdot 2^{t+1})$ and set $X_1, \ldots, X_\ell$ to be strings that ε'-fool $\mathsf{GC}^0_{d-1}(\log m) \circ \mathsf{AND}_{\log m}$ circuits of size $4m^2$ if $d \geq 2$, which by the inductive hypothesis uses

  $$O(\log^{d-1}(m)\log(m/\varepsilon)\log\log m)$$

  random bits per X_i. If $d = 1$, use the PRG from Lemma 39 giving a seed length, which needs $O(\log(m/\varepsilon))$ seed per X_i.

- Let Y be a string that $\varepsilon/((64mw)^{t+w+1}(2m)^{2t})$-fools CNF of size m^2, samplable using

  $$O(\log m \log((mw)^{t+w} m^t/\varepsilon) \log\log m) = O(\log(m/\varepsilon) \log^2 m \log\log m)$$

  bits.

  The PRG will sample the above strings and output the following computation

  $$Y \oplus (X_1 \wedge H_1) \oplus \cdots \oplus (X_\ell \wedge H_\ell)$$

where $\wedge$ and $\oplus$ are the bitwise AND and XOR operations, respectively. Therefore, we get a total seed length of

$$\begin{aligned} O(\log n \log(m/\varepsilon) &+ \ell \log(m/\varepsilon) + \log(m/\varepsilon) \log^2 m \log\log m) \\ &= O(\log(m/\varepsilon) \log^2 m \log\log m) \end{aligned}$$

if $d = 1$ and

$$\begin{aligned} O(\log n \log(m/\varepsilon) &+ \ell \log^{d-1}(m) \log(m/\varepsilon) \log\log m + \log(m/\varepsilon) \log^2 m \log\log m) \\ &= O((w \log^{d-1}(m) + \log^2(m)) \log(m/\varepsilon) \log\log m). \end{aligned}$$

Let C be an arbitrary $\mathsf{GC}^0_d(\log m) \circ \mathsf{AND}_w$ circuit, and let $U_1, \ldots U_\ell$ be independent and uniform n-bit strings. Like in [18] we use a hybrid argument to prove the theorem using the hybrid distributions

$$D_i = Y \oplus \bigoplus_{1 \le j \le i} (U_i \wedge H_i) \oplus \bigoplus_{i < j \le \ell} (X_i \wedge H_i)$$

for $0 \le i \le \ell$. Noting D_0 is the PRG output, while D_ℓ is a uniform string, it suffices to show

$$|\mathbb{E}_{x \sim D_{i-1}}[C(x)] - E_{x \sim D_i}[C(x)]| \le \varepsilon/\ell \tag{8}$$

for all $1 \le i \le \ell$, from which summing over all i and applying the triangle inequality gets the desired result.

Notice each H_i is $2t$-wise $\frac{1}{\ell}$-bounded. Conditioned on H, note that $Z_i := Y \oplus \bigoplus_{1 \le j < i}(U_i \wedge H_i) \oplus \bigoplus_{i < j \le \ell}(X_i \wedge H_i)$ $\varepsilon/((64mw)^{t+w+1}(2m)^{2t})$-fools CNF of size m^2 since Y does. Let $\mathcal{F}$ be the collection of all bottom depth-2 $\mathsf{G}(k) \circ \{\mathsf{AND}_w, \mathsf{OR}_w\}$ subcircuits of C. Therefore, if we let $\mathcal{E}$ be the event that $\mathcal{F}|_{\rho(H_i, Z_i)}$ has no $\log m$-partial depth-t DT, by Theorem 38 it follows

$$\begin{aligned} &\Pr_{H, Y, U_{<i}, X_{>i}}[\mathcal{E}] \\ &\quad \le 4(m 2^{\log m})^{t/\log m}(64w/\ell)^t + (64mw)^{t+w}(2m)^{2t \log m/\log m} \cdot \frac{\varepsilon}{(64mw)^{2t}(2m)^{2t}} \\ &\quad \le 4 \cdot 2^{2t}(1/16)^t + \frac{\varepsilon}{(64mw)^{40\log(m/\varepsilon) - \log m}} \\ &\quad \le 4(1/4)^t + \frac{\varepsilon}{4\ell} \\ &\quad \le \frac{\varepsilon}{2\ell} \end{aligned}$$

Conditioning on $\neg\mathcal{E}, H, Y, U_{<i}, X_{>i}$, we see upon replacing all depth 2 subcircuits with DT_ts and applying Lemma 30, $C|_{\rho(H_i, Z_i)}$ is computable by a depth-t DT where each leaf L_j is an $\mathsf{GC}^0_{d-1}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuit of size $\le m \cdot 2^{\log m} + m \le 2m^2$, and will become a

$\mathsf{DT}_{\log m}$ tree if $d = 1$. In either case, we see by construction that X_i will fool it. Formally, we note that conditioned on the good events above, we have $C_{\rho(H_i,Z_i)}(y) = \sum_{j=1}^{2^t} L_j(y) \cdot \mathbb{1}\{T(y) \text{ reaches } L_j\}$. By construction of X_i,

$$|\mathbb{E}_{X_i}[L_j(X_i + Z_i) \cdot \mathbb{1}\{T(X_i + Z_i) \text{ reaches } L_j\}] - \mathbb{E}_{U_i}[L_j(U_i + Z_i) \cdot \mathbb{1}\{T(U_i + Z_i) \text{ reaches } L_j\}]| \leq \frac{\varepsilon}{\ell \cdot 2^{t+1}}$$

Summing over all $1 \leq j \leq 2^t$, applying the Triangle Inequality, and using linearity of expectation, we see

$$\mathbb{E}_{X_i}[C_{\rho(H_i,Z_i)}(X_i + Z_i)] - \mathbb{E}_{U_i}[C_{\rho(H_i,Z_i)}(U_i + Z_i)]| \leq \frac{\varepsilon}{2\ell}.$$

Therefore,

$$|\mathbb{E}_{x\sim D_{i-1}}[C(x)] - \mathbb{E}_{x\sim D_i}[C(x)]| \leq \Pr[\mathcal{E}] + \Pr[\neg\mathcal{E}] \cdot \frac{\varepsilon}{2\ell} \leq \frac{\varepsilon}{\ell}$$

and (8) is proven. ◀

From this, we immediately get PRGs for size-m $\mathsf{GC}^0_d(\log m)$ circuits.

▶ **Theorem 41.** *For every m, n, d and $\varepsilon > 0$, there is an ε-PRG for size-m $\mathsf{GC}^0_d(\log m)$ with seed length $O((\log^{d-1}(m) + \log^2(m)) \log(m/\varepsilon) \log\log m)$*

Proof. Add trivial fan-in 1 gates to the bottom so that we effectively have a $\mathsf{GC}^0_d(\log m)\circ\mathsf{AND}_1$ circuit. By Theorem 40, we can fool this with seed length

$$O((\log^{d-1}(m) + \log^2(m)) \log(m/\varepsilon) \log\log m).$$ ◀

4.4 Fourier Spectrum Bounds for $\mathsf{GC}^0(k)$

Linial, Mansour, Nisan, and Tal showed that many notions of the Fourier spectrum of a function class is intimately related [16, 19, 30]. [30] writes out four key properties and conveniently describes the implications existing between them. We report a slightly altered version here.

▶ **Theorem 42** ([16, 19, 30])**.** *Say for a class of functions, $\mathcal{C}$ we have the following property.*

- ***ESFT**: Exponentially small Fourier tails. For all $f \in \mathcal{C}$,*

  $$W^{\geq k}[f] \leq Ce^{-\Omega(k/t)}.$$

  for some constant C.

Then, $\mathcal{C}$ also satisfies the following for some constant C'.

- ***SLPT**: Switching lemma type property. For all $f \in \mathcal{C}, d, p$,*

  $$\Pr_{\rho\sim R_p}[\deg(C|_\rho) \geq d] \leq C' \cdot O(pt)^d.$$

- ***InfK**: Bounded total degree-k influence. For all $f \in \mathcal{C}$,$0 \leq k \leq n$,*

  $$Inf^k[f] \leq C' \cdot O(t)^k.$$

- ***L1**: Bounded L_1 norm at the kth level. For all $f \in \mathcal{C}$, $0 \leq k \leq n$,*

  $$\sum_{|S|=k} |\widehat{f}(x)| \leq C' \cdot O(t)^k$$

- ***FMC**: Fourier mass concentration. For all $f \in \mathcal{C}$, f is ε-concentrated on $t^{O(t\log(1/\varepsilon))}$ coefficients.*

Due to the above unification result, it appears like we can bootstrap Corollary 32 to give us a plethora of information about the Fourier spectrum of $\mathsf{GC}^0(k)$. Unfortunately, upon closer inspection, the Corollary doesn't quite give the exact property of SLPT. We instead show that $\mathsf{GC}^0(k)$ has ESFT. Our proofs will use the following lemma.

▶ **Lemma 43** ([16]). *For $f : \{\pm 1\}^n \to \{\pm 1\}$, $0 \le \ell \le n$, and $p \in [0,1]$,*

$$W^{\ge \ell}[f] \le 2\mathbb{E}_{\rho \sim R_p} W^{\ge kp}[f|_\rho]$$

We first start off with depth 2 circuits.

▶ **Lemma 44.** *Let f be a $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$. Then*

$$W^{\ge \ell}[f] \le 2 \cdot 2^{-\ell/80w+k}$$

Proof. Let $p = 1/40w$ and $t = \ell/80w$. By Theorem 20, if $\rho \sim R_p$, $f|_\rho$ becomes a depth-t DT with $\ge 1 - (20w/40w)^t 2^k = 1 - 2^{-t+k}$ probability. Such trees have no Fourier mass above level t. Say ρ is good if $f|_\rho$ does indeed become a DT. Using Lemma 43 it follows

$$\begin{aligned} W^{\ge \ell}[f] &\le 2\mathbb{E}_{\rho \sim R_p}[W^{\ge p\ell}[f|_\rho]] \\ &\le 2\mathbb{E}_{\rho \sim R_p}[W^{\ge \ell/40w}[f|_\rho] | \rho \text{ is good}] + 2 \cdot 2^{-t+k} \\ &\le 2 \cdot 2^{-\ell/80w+k}. \end{aligned}$$

◀

We can now use this as a base case to prove ESFT for GC^0. We will need to utilize the following lemma.

▶ **Lemma 45** ([30]). *Let $f : \{\pm 1\}^n \to \{\pm 1\}$, $0 \le \ell \le n$, and let T be a depth d decision tree such that for any leaf ℓ and the corresponding restriction ρ_ℓ induced by the root-to-leaf path, we have $W^{\ge \ell}[f|_{\rho_\ell}] \le \varepsilon$. Then $W^{\ge \ell + d}[f] \le \varepsilon$.*

We now state and prove the theorem. Define the *effective size* of a Boolean circuit to be the number of gates in the circuit at distance 2 or more from the inputs.

▶ **Theorem 46.** *Let f be a $\mathsf{GC}^0_d(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuit with effective size m. Then*

$$W^{\ge \ell}[f] \le 4^d \cdot 2^{-\frac{\ell}{80w(128(k+\log m))^{d-1}} + k}$$

Proof. We apply induction. The base case of $d = 1$ is taken care of by Lemma 44.

We now prove the inductive step for depth d. Sample $\rho \sim R_p$ with

$$p = \frac{1}{128w(m2^k)^{1/(k+\log m)}} = \frac{1}{128w},$$

and let $t = p\ell/2 = \ell/256w$. By Theorem 28, all the bottom depth-2 $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ subcircuits of $f|_\rho$ can be calculated by a $(k + \log m)$-partial depth-t decision tree with probability $\ge 1 - 4 \cdot 2^{-t}$. By Lemma 30, this implies $f|_\rho$ becomes a $(t, \mathsf{GC}^0_{d-1}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_{k+\log m})$-tree T. Furthermore, each leaf circuit has effective size $\le m$. Call ρ *good* if $f|_\rho$ simplifies to such a tree. Then

$$W^{\ge \ell}[f] \le 2\mathbb{E}_{\rho \sim R_p}[W^{\ge p\ell}[f|_\rho]] \le 2\mathbb{E}_{\rho \sim R_p}[W^{\ge p\ell}[f|_\rho] | \rho \text{ is good}] + 8 \cdot 2^{-t}.$$

Fix a good ρ. For a leaf L of T, let τ_L be the restriction induced by the path to L in T. We know by Lemma (cite) that

$$W^{\ge p\ell}[f|_\rho] \le \max_{\text{leaf } L} W^{\ge p\ell - t}[f|_{\rho \circ \tau_L}] \le \max_{\text{leaf } L} W^{\ge p\ell/2}[f|_{\rho \circ \tau_L}].$$

As $f|_{\rho\circ\tau_L}$ is a $\mathsf{GC}^0_{d-1}(k)\circ\{\mathsf{AND},\mathsf{OR}\}_{k+\log m}$ circuit for every L we can then use the inductive hypothesis to bound

$$\max_{\text{leaf } L} W^{\geq p\ell/2}[f|_{\rho\circ\tau_L}] \leq 4^{d-1}\cdot 2^{-\frac{p\ell}{80(k+\log m)(128(k+\log m))^{d-2}}+k} = 4^{d-1}\cdot 2^{-\frac{\ell}{80w(128(k+\log m))^{d-1}}+k}.$$

Putting this all together, we get

$$\begin{aligned} W^{\geq\ell}[f] &\leq 2\mathbb{E}_{\rho\sim R_p}[W^{\geq p\ell}[f|_\rho]|\rho \text{ is good}] + 8\cdot 2^{-t} \\ &\leq 2\cdot 4^{d-1}\cdot 2^{-\frac{\ell}{80w(128(k+\log m))^{d-1}}+k} + 8\cdot 2^{-\ell/256w} \\ &\leq 4^d\cdot 2^{-\frac{\ell}{80w(128(k+\log m))^{d-1}}+k} \end{aligned}$$

◀

We can bootstrap Theorem 46 with Theorem 42 to yield the following properties about $\mathsf{GC}^0(k)$

▶ **Theorem 47.** *Let f be a size-m $\mathsf{GC}^0_d(k)$ circuit and define $t := (k+\log m)^{d-1}$. Then the following is true for some C*

1. *ESFT: $W^{\geq\ell}[f] \leq C\cdot 2^k\cdot 2^{-\Omega\left(\frac{\ell}{t}\right)}$.*
2. *SLTP: For all $0<p<1$, $\Pr_{\rho\sim R_p}[\deg(f|_\rho)\geq\ell] \leq C\cdot O(pkt)^\ell$.*
3. *InfK: $\mathrm{Inf}^\ell[f] \leq C\cdot O(kt)^\ell$.*
4. *L1: $\sum_{|S|=\ell}|\widehat{f}(x)| \leq C\cdot O(kt)^\ell$.*
5. *FMC: f is ε-concentrated on $2^{O((k+\log(1/\varepsilon))t\log t)}$ coefficients.*

where f and any hidden constants only depend on d.

Proof. Add a trivial $(d+1)$-st layer of AND_1 gates at the base of f and apply Theorem 46 to deduce that

$$W^{\geq\ell}[f] \leq 4^d\cdot 2^{-\frac{\ell}{80(128(k+\log m))^{d-1}}+k},$$

proving the first item. Now since we know $W^{\geq\ell}[C]\leq 1$ (by Parseval's) and $k\geq 1$, it follows that

$$W^{\geq\ell}[f] \leq (W^{\geq\ell}[C])^{1/k} \leq C_d\cdot 2^{-\Omega\left(\frac{\ell}{kt}\right)}.$$

Therefore, the second, third, and fourth items follow by applying Theorem 42 (as well as a version of the fifth item with weaker parameters). We now prove Item 5.

Notice that for $w := t\cdot O(k+\log(1/\varepsilon))$, we have by Item 1 that $W^{\geq w}[f]\leq\varepsilon/2$. Now by Item 4,

$$\sum_{|S|<w}|\widehat{f}(S)| \leq \sum_{i=0}^{w-1} O(kt)^i \leq (C'kt)^w. \tag{9}$$

Now let $\mathcal{F} = \{S : |S|<w \text{ and } |\widehat{f}(S)| \geq \frac{\varepsilon/2}{(C'kt)^w}\}$. Notice that

$$\begin{aligned} \sum_{S\in\mathcal{F}}\widehat{f}(S)^2 &= 1 - \sum_{|S|\geq w}\widehat{f}(S)^2 - \sum_{|S|<w, S\notin\mathcal{F}}\widehat{f}(S)^2 \\ &\geq 1-\varepsilon/2 - \frac{\varepsilon/2}{(C'kt)^w}\sum_{|S|<w}|\widehat{f}(S)| \\ &\geq 1-\varepsilon. \end{aligned}$$

By Equation (9), the maximum number of terms in $\mathcal{F}$ can be at most

$$(C'kt)^w / \left(\frac{\varepsilon/2}{(C'kt)^w} \right) = 2(C'kt)^{2w}/\varepsilon = 2^{O(w \log(kt) + \log(1/\varepsilon))} = 2^{O((k+\log(1/\varepsilon))t \log t)}$$

and thus Item 5 is proved. ◀

As a first application, the work of Kushilevitz and Mansour [13] allows us to translate FMC to learnability results.

▶ **Lemma 48** ([13]). *Let f be a Boolean function such that there exists a t-sparse multivariate polynomial g (over the Fourier basis) such that $\mathbb{E}_{x \sim U_n}[(f(x) - g(x))^2] \leq \varepsilon$. There exists a randomized algorithm, whose running time is polynomial in $t, n, 1/\varepsilon, \log(1/\delta)$ such that given blackbox access to f and $\delta > 0$, outputs a function h such that over the randomness of the algorithm,*

$$\Pr[\mathbb{E}_{x \sim U_n}[(f(x) - h(x))^2] \leq O(\varepsilon)] \geq 1 - \delta.$$

Using this lemma, we can derive a learning algorithm for $\mathsf{GC}^0(k)$.

▶ **Theorem 49.** *There exists an algorithm such that given blackbox access to any $C \in \mathsf{GC}^0_d(k)$ of size m and $\delta > 0$, outputs a function h such that over the randomness of the algorithm,*

$$\Pr[\mathbb{E}_{x \sim U_n}[(f - h)^2] \leq O(\varepsilon)] \geq 1 - \delta.$$

Furthermore, this algorithm runs in poly$(n, 2^{\tilde{O}((k+\log(1/\varepsilon))(k+\log m)^{d-1})}, 1/\varepsilon, \log(1/\delta))$

Proof. From Theorem 47, for any $C \in \mathsf{GC}^0_d(k)$, there exists g of sparsity

$$t = 2^{\tilde{O}((k+\log(1/\varepsilon))(k+\log m)^{d-1})},$$

created by taking the Fourier expansion of C and only keeping the ε-concentrated coefficients $\mathcal{S} \subset 2^{[n]}$, such that

$$\mathbb{E}_{x \sim \{\pm 1\}^n}[(C(x) - g(x))^2] \leq \mathbb{E}_{x \sim \{\pm 1\}^n}\left[\left(\sum_{S \notin \mathcal{S}} \widehat{C}(S) x^S \right)^2 \right] = \sum_{S \notin \mathcal{S}} \widehat{C}(S)^2 \leq \varepsilon.$$

The result then follows by Lemma 48. ◀

We also can prove a new correlation bound result with this Fourier spectrum. It is known that MAJ is a symmetric function that has $O_d(\log^{d-1}(m)/\sqrt{n})$ correlation against size-m $\mathsf{AC}^0_d[\oplus]$ circuits. A natural question to ask is whether Majority is special in this regard, or if a random symmetric function (use $n+1$ coin tosses to assign a bit to each Hamming level) will display $O_d(\log^{d-1}(m)/n^\alpha)$ correlation against size-m $\mathsf{AC}^0_d[\oplus]$ circuits for some α. Tal ([30], Theorem 6.1) used ESFT and L1 of AC^0 to prove that random symmetric functions (or more specifically balanced symmetric functions) display $O_d(\log^{d-1}(m)/\sqrt{n})$ correlation against size-m AC^0_d circuits, so it is natural to believe that this should similarly be true against $\mathsf{AC}^0[\oplus]$. Unfortunately, since PAR has all its Fourier weight at level n, this proof approach is doomed to fail for $\mathsf{AC}^0[\oplus]$ circuits, as the class doesn't demonstrate ESFT. However, we can now give partial progress towards this goal by showing a random symmetric function uncorrelates with $\mathsf{GC}^0(k)$ circuits, as this class contains gates which calculate parity as long as the Hamming weight of the input is at most k. This result can be seen as finding out how general of a circuit class we can stretch the Fourier argument before we reach the roadblock on this approach demonstrated by PAR.

▶ **Theorem 50.** *Let $f \in \mathsf{GC}^0_d(k)$, and let g be a symmetric function, both mapping $\{\pm 1\}^n \to \{\pm 1\}$, and let $(k+\log m)^{d-1} \le O\left(\frac{n}{k+\log n}\right)^{1/3}$. Then*

$$\mathsf{corr}(f,g) := \mathbb{E}_x[f(x)g(x)] \le |\widehat{g}(\emptyset)| + \frac{C_d k(k+\log m)^{d-1}}{\sqrt{n}}$$

Proof. We note for ℓ' to be picked later, we can decompose

$$\begin{aligned}\mathsf{corr}(f,g) &= |\mathbb{E}_x[f(x)g(x)]| \\ &= \left|\sum_{S\subset[n]} \widehat{f}(S)\widehat{g}(S)\right| \\ &\le |\widehat{g}(\emptyset)| + \sum_{|S|<\ell'} |\widehat{f}(S)\widehat{g}(S)| + \sum_{|S|\ge\ell} |\widehat{f}(S)\widehat{g}(S)|. \end{aligned} \tag{10}$$

We will bound the first summation using L1, and the second summation by ESFT. The second summation can be bounded as follows using Cauchy-Schwarz.

$$\sum_{|S|\ge\ell} |\widehat{f}(S)\widehat{g}(S)| \le \sqrt{W^{\ge\ell'}[f]\cdot W^{\ge\ell'}[g]} \le \sqrt{4^d \cdot 2^{-\frac{\ell'}{80(128(k+\log m))^{d-1}}+k}} \le 1/\sqrt{n} \tag{11}$$

if we set $\ell' = c_d(k+\log n)(k+\log m)^{d-1}$ for some constant c_d only depending on d. Now to bound the first summation, note since g is symmetric, $\widehat{g}(S)$ is constant over all S of same cardinality. Therefore,

$$|\widehat{g}(S)| = \sqrt{\widehat{g}(S)^2} = \sqrt{\frac{1}{\binom{n}{|S|}}\sum_{S';|S'|=|S|}\widehat{g}(S')^2} \le \sqrt{\frac{1}{\binom{n}{|S|}}}.$$

Hence using L1 from Theorem 47, we can bound

$$\sum_{|S|<\ell'} |\widehat{f}(S)\widehat{g}(S)| \le \sum_{1\le\ell<\ell'} \sqrt{\frac{1}{\binom{n}{\ell}}}\sum_{|S|=\ell}|\widehat{f}(S)| \le \sum_{1\le\ell<\ell'}\left(\frac{C_d k(k+\log m)^{d-1}}{\sqrt{n/\ell}}\right)^{\ell} \tag{12}$$

where C_d is some constant depending on d. We bound this sum by a geometric series of the same first term and with common ratio $1/2$. Indeed, we see that the ratio of consecutive terms will be

$$\frac{\left(\frac{C_d k(k+\log m)^{d-1}}{\sqrt{n/(\ell+1)}}\right)^{\ell+1}}{\left(\frac{C_d k(k+\log m)^{d-1}}{\sqrt{n/\ell}}\right)^{\ell}} = \frac{C_d k(k+\log m)^{d-1}}{\sqrt{n}}\sqrt{\frac{(\ell+1)^{\ell+1}}{\ell^\ell}} \le \frac{C_d k(k+\log m)^{d-1}}{\sqrt{n}}\sqrt{e\ell'} \le 1/2$$

where the last inequality follows from the assumption $(k+\log m)^{d-1} \le O\left(\frac{n}{k+\log n}\right)^{1/3}$. Hence the quantity in Equation (12) can be upper bounded by twice the first term, so

$$\sum_{|S|<\ell'} |\widehat{f}(S)\widehat{g}(S)| \le \frac{C_d k(k+\log m)^{d-1}}{\sqrt{n}}.$$

Hence from (10),

$$\mathsf{corr}(f,g) \le |\widehat{g}(\emptyset)| + \frac{1}{\sqrt{n}} + \frac{C_d k(k+\log m)^{d-1}}{\sqrt{n}}.$$

◀

5 Open Problems

We conclude with some directions for future research.

- Our tightness result in Theorem 52 uses a function in $\mathsf{G}(k) \circ \mathsf{AND}_w$, but it is not known whether a k-$\mathsf{OR} \circ \mathsf{AND}_w$ circuit can saturate the bound. In particular, is Theorem 20 tight for $\mathsf{AC}^0(k)$ or $\mathsf{TC}^0(k)$ circuits?
- It was already noted that Corollary 34 is tight in essentially every way possible for $\mathsf{GC}^0(k)$ circuits. However, all our tightness results (Theorem 52) use constructions that abuse the generality of $\mathsf{GC}(k)$. Are there constructions exhibiting tightness which are in $\mathsf{TC}^0(k)$ or $\mathsf{AC}^0(k)$? Alternatively, can we obtain stronger size lower bounds if we were only concerned with $\mathsf{AC}^0(k)$ or even $\mathsf{TC}^0(k)$ circuits? Either finding a pathological construction in $\mathsf{AC}^0(k)/\mathsf{TC}^0(k)$ or proving stronger lower bounds for these weaker circuit class would be interesting.
- We touched up on how a result of Allender and Koucký ([2], Theorem 3.8) states that there exists an absolute constant C_{AK} such that MAJ_n can be written as an $\mathsf{AC}^0(n^\varepsilon)$ circuit with depth $\leq C_{AK}/\varepsilon$ and size $O(n^{1+\varepsilon})$. [2] actually only use $\mathsf{AND}_2, \mathsf{OR}_2$, and $\mathsf{MAJ}_{n^\varepsilon}$ gates. If we were allowed all the gate classes in $\mathsf{AC}^0(n^\varepsilon)$, could we find a better construction (more specifically a lower depth blowup)? That way, we would get a stronger reduction from proving bounds on TC to $\mathsf{AC}^0(n^\varepsilon)$ where we only need to show size lower bounds on smaller depth $\mathsf{AC}^0(n^\varepsilon)$ circuits.
- Are there any other applications of the generalized switching lemma? Due to the versatility of this theorem, it can essentially be plugged in wherever the classical switching lemma was used to get more general results. Perhaps this can generalize other results or even push a switching lemma argument that initially wouldn't go through (the remark after Theorem 52 gives an example of the general switching lemma giving stronger bounds than the classical one).

References

1 Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1), February 2009. doi:10.1145/1490270.1490272.

2 Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3), March 2010. doi:10.1145/1706591.1706594.

3 N. Alon, O. Goldreich, J. Hastad, and R. Peralta. Simple construction of almost k-wise independent random variables. In *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*, pages 544–553 vol.2, 1990. doi:10.1109/FSCS.1990.89575.

4 Theodore Baker, John Gill, and Robert Solovay. Relativizations of the p?=np question. *SIAM Journal on Computing*, 4(4):431–442, 1975. doi:10.1137/0204037.

5 Norbert Blum. A boolean function requiring 3n network size. *Theor. Comput. Sci.*, 28:337–345, 1984. doi:10.1016/0304-3975(83)90029-4.

6 Lijie Chen and Roei Tell. Bootstrapping results for threshold circuits "just beyond" known lower bounds. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, pages 34–41, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3313276.3316333.

7 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM Journal on Computing*, 43(5):1699–1708, 2014. doi:10.1137/120897432.

8 Pooya Hatami, William M. Hoza, Avishay Tal, and Roei Tell. Fooling constant-depth threshold circuits (extended abstract). In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 104–115, 2022. doi:10.1109/FOCS52979.2021.00019.

9 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for ac0. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 961–972, USA, 2012. Society for Industrial and Applied Mathematics.

10 Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size–depth tradeoffs for threshold circuits. *SIAM Journal on Computing*, 26(3):693–707, 1997. doi:10.1137/S0097539792282965.

11 Daniel M. Kane and Ryan Williams. Super-linear gate and super-quadratic wire lower bounds for depth-two and depth-three threshold circuits. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, pages 633–643, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2897518.2897636.

12 Zander Kelley. An improved derandomization of the switching lemma. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 272–282. ACM, 2021. doi:10.1145/3406325.3451054.

13 Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993. doi:10.1137/0222080.

14 Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, 1996. doi:10.1017/CBO9780511574948.

15 Jiatu Li and Tianqi Yang. 3.1n - o(n) circuit lower bounds for explicit functions. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 1180–1193, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519935.3519976.

16 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, fourier transform, and learnability. *J. ACM*, 40(3):607–620, July 1993. doi:10.1145/174130.174138.

17 Shachar Lovett and Srikanth Srinivasan. Correlation bounds for poly-size ac0 circuits with n(1-o(1)) symmetric gates. In Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 640–651, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

18 Xin Lyu. Improved pseudorandom generators for ac0 circuits. In *Proceedings of the 37th Computational Complexity Conference*, CCC '22, Dagstuhl, DEU, 2022. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2022.34.

19 Yishay Mansour. An o(nlog log n) learning algorithm for dnf under the uniform distribution. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 53–61, New York, NY, USA, 1992. Association for Computing Machinery. doi:10.1145/130385.130391.

20 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014. doi:10.1017/CBO9781139814782.

21 Igor Carboni Oliveira, Rahul Santhanam, and Srikanth Srinivasan. Parity Helps to Compute Majority. In Amir Shpilka, editor, *34th Computational Complexity Conference (CCC 2019)*, volume 137 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 23:1–23:17, Dagstuhl, Germany, 2019. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2019.23.

22 Aaron Potechin. On the approximation resistance of balanced linear threshold functions. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, pages 430–441, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3313276.3316374.

23 Alexander Razborov and Avi Wigderson. n(log n) lower bounds on the size of depth-3 threshold cicuits with and gates at the bottom. *Information Processing Letters*, 45(6):303–307, 1993. doi:10.1016/0020-0190(93)90041-7.

24 Alexander A. Razborov. Lower bounds on the size of bounded depth circuits over a complete basis with logical addition. *Mathematical notes of the Academy of Sciences of the USSR*, 41:333–338, 1987.

25 Alexander A Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997. doi:10.1006/jcss.1997.1494.

26 Rocco A. Servedio. Every linear threshold function has a low-weight approximator. In *Proceedings of the 21st Annual IEEE Conference on Computational Complexity*, CCC '06, pages 18–32, USA, 2006. IEEE Computer Society. doi:10.1109/CCC.2006.18.

27 Rocco A. Servedio and Li-Yang Tan. Luby-Velickovic-Wigderson Revisited: Improved Correlation Bounds and Pseudorandom Generators for Depth-Two Circuits. In Eric Blais, Klaus Jansen, José D. P. Rolim, and David Steurer, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2018)*, volume 116 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 56:1–56:20, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.APPROX-RANDOM.2018.56.

28 Rocco A. Servedio and Li-Yang Tan. Improved pseudorandom generators from pseudorandom multi-switching lemmas. *Theory of Computing*, 18(4):1–46, 2022. doi:10.4086/toc.2022.v018a004.

29 R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, pages 77–82, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/28395.28404.

30 Avishay Tal. Tight Bounds on the Fourier Spectrum of AC0. In Ryan O'Donnell, editor, *32nd Computational Complexity Conference (CCC 2017)*, volume 79 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 15:1–15:31, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2017.15.

31 Luca Trevisan and Tongke Xue. A derandomized switching lemma and an improved derandomization of ac0. In *2013 IEEE Conference on Computational Complexity*, pages 242–247, 2013. doi:10.1109/CCC.2013.32.

32 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012. doi:10.1561/0400000010.

33 Emanuele Viola. Pseudorandom bits for constant-depth circuits with few arbitrary symmetric gates. *SIAM Journal on Computing*, 36(5):1387–1403, 2007. doi:10.1137/050640941.

A Deferred Proofs

A.1 Showing $\mathsf{TC}^0(k) \subset \mathsf{GC}^0(k)$

Here, we prove that circuits created by biased LTF gates are indeed contained in $\mathsf{GC}^0(k)$.

▶ **Theorem 51.** *Any* THR *gate* f *with balance* $\leq k$ *(see Definition 3) is, upon negating certain input bits, in* $\mathsf{G}(k)$.

Proof. Let $f : \{0,1\}^n \to \{\pm 1\}$ be defined as $f(x) = \operatorname{sgn}(\sum_{i=1}^{n} w_i(-1)^{x_i} - \theta)$. By negating input bits, we can assume each $w_i \geq 0$. Furthermore, since the definition of $\mathsf{G}(k)$ is symmetric (solely depends on the sum of input bits), we can assume WLOG that $0 \leq w_1 \leq \cdots \leq w_n$. Since f has balance $\leq k$, we know that $-\sum_{i\leq k} w_i + \sum_{i>k} w_i < \theta$. Assuming $-\sum_{i\leq k} w_i + \sum_{i>k} w_i < |\theta|$, we will show f is an orlike $\mathsf{G}(k)$ gate (an analogous proof will show the case for $-\theta$ and andlike).

Consider x such that $\sum x_i \geq k$. Let $S_x \subset [n]$ denote the set where $i \in S_x$ iff $x_i = 1$. It follows that

$$f(x) = \operatorname{sgn}\left(-\sum_{i\in S_x} w_i + \sum_{i\notin S_x} w_i - \theta\right)$$
$$= \operatorname{sgn}\left(-\sum_{i\leq k} w_i + \sum_{i>k} w_i - \theta + 2\sum_{i\in[k]\setminus S_x} w_i - 2\sum_{i\in[k+1,n]\cap S_x} w_i\right) \tag{13}$$

We know by assumption that $-\sum_{i\leq k} w_i + \sum_{i>k} w_i - \theta \leq 0$. Since $|S_x| \geq k$,

$$|[k] \setminus S_x| = k - |k \cap S_x| \leq |[k+1, n] \cap S_x|,$$

and each element in $[k] \setminus S_x$ is strictly smaller than each element in $[k+1, n] \cap S_x$. Combining these observations with the fact $w_1 \leq \cdots \leq w_n$, it follows

$$2\sum_{i\in[k]\setminus S_x} w_i - 2\sum_{i\in[k+1,n]\cap S_x} w_i \leq 0.$$

Therefore,

$$-\sum_{i\leq k} w_i + \sum_{i>k} w_i - \theta + 2\sum_{i\in[k]\setminus S_x} w_i - 2\sum_{i\in[k+1,n]\cap S_x} w_i \leq 0.$$

Combining this with (13) it follows $f(x) = -1$ for any x with $\sum x_i \geq k$. Hence f is an orlike $\mathsf{G}(k)$ gate as desired. ◀

A.2 Tightness of the $\mathsf{GC}^0(k)$ Switching Lemma and Correlation Bounds

In this section, we give constructions which show that various bounds we establish are indeed tight. We first show that the switching lemma we established is tight.

▶ **Theorem 52.** *Let p, w, t, k be parameters such that $pkw < 1/2$ and $t > k/2$. There exists a $\mathsf{G}(k) \circ \mathsf{AND}_w$ circuit C such that*

$$\Pr_{\rho\sim R_p}[\mathsf{DT}(C|_\rho) \geq t] \geq 2^{k/2}(.5pw)^t$$

Proof. We will take $C = \mathsf{PAR}_{kw}$. To see that this is computable in $\mathsf{G}(k) \circ \mathsf{AND}_w$, write

$$\mathsf{PAR}_{kw}(x) = \mathsf{PAR}_k(\mathsf{PAR}_w(x_1, \ldots, x_w), \ldots, \mathsf{PAR}_w(x_{(k-1)w+1}, \ldots, x_{kw})).$$

Now, write out each bottom layer PAR_w as a size $w2^{w-1}$ CNF which takes the ORs of the 2^{w-1} AND clauses corresponding to w-bit inputs with an odd number of ones. Notice that for any assignment,

- at most *one* of the 2^{w-1} clauses under each OR can simultaneously be satisfied,
- which implies at most k of the bottom layer clauses can be simultaneously satisfied.

By the first bullet point, we can turn the OR gates into PAR gates, turning C into a $\mathsf{PAR}_{k2^{w-1}} \circ \mathsf{AND}_w$ circuit. By the second bullet point, we can replace the top $\mathsf{PAR}_{k2^{w-1}}$ gate with the gate $\mathsf{G} \in \mathsf{G}(k)$ which calculates parity if at most k input bits are one, and outputs 0 otherwise. Consequently, C can be calculated by a $\mathsf{G}(k) \circ \mathsf{AND}_w$ circuit.

We can now directly calculate the simplification probability. Notice that if $\geq t$ variables are alive in ρ, then $C|_\rho$ must have decision tree depth $\geq t$ (since even if $d-1$ input bits are known, parity remains ambiguous). Hence we calculate

$$\begin{aligned}\Pr_{\rho\sim R_p}[\mathsf{DT}(C|_\rho)\geq t] &\geq \sum_{i=t}^{kw}\binom{kw}{i}p^i(1-p)^{kw-i}\\ &\geq (1-p)^{kw}\left(\frac{kw}{t}\right)^t\left(\frac{p}{1-p}\right)^t\\ &\geq (1-pkw)2^{k/2}\left(\frac{pw}{1-p}\right)^t\\ &\geq 2^{k/2}(.5pw)^t\end{aligned}$$

where the third inequality follows from the fact $(k/t)^t$ is increasing in t for $t > k/2$ ◀

▶ Remark 53. Notice that if PAR_n was written as a width n-CNF and the classical switching lemma was applied, we would get a "failure to simplify" probability bound of $\leq (5pn)^t$. However, if we rewrite PAR_n as a $\mathsf{G}(t\log(n/t))\circ\mathsf{AND}_{n/t\log(n/t)}$ circuit in a similar manner as above, and use Theorem 20, we get a bound of $\leq (20pn/t\log(n/t))^t 2^{t\log(n/t)} \leq (20pn^2/t^2)^t$, which is asymptotically stronger when $t=\omega(\sqrt{n})$. This shows that we can potentially obtain tighter parameters by expressing functions as a more compact $\mathsf{GC}^0(k)$ circuit and applying Theorem 20 rather than using the classical switching lemma on a larger AC^0 circuit computing the same function.

We will now show that the circuit lower bound established in Corollary 34 is tight. We thank an anonymous reviewer for pointing us to this construction.

▶ **Theorem 54.** *There exists a size-*$2^{O((n/k)^{\frac{1}{d-1}})}$ $\mathsf{GC}^0_d(k)$ *circuit which computes* PAR_n.

Proof. Split the input string into k blocks of size n/k bits each. We can compute the parity of each of the n/k-size blocks straightforwardly using a depth $d-1$ circuit made out of $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ gates; iteratively group the bits into blocks of $(n/k)^{\frac{1}{d-1}}$, and use a gate to take the parity of each block, thereby creating a depth $d-1$ tree of $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ gates. Finally, we can take a PAR_k of these k depth-$(d-1)$ circuits to get a depth d circuit which computes the parity of all n bits. We now focus on converting this parity-riddled circuit to a $\mathsf{GC}^0(k)$ one.

Consider the top depth-2 subcircuit, which is a $\mathsf{PAR}_k\circ\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ circuit. Notice that $\mathsf{PAR}_k\in\mathsf{G}(k)$ and $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ is trivially computable by a decision tree in $\mathsf{DT}_{(n/k)^{\frac{1}{d-1}}}$. Therefore by Lemma 30, this top subcircuit can be replaced by a size $2^{O((n/k)^{\frac{1}{d-1}})}$ $\mathsf{G}(k)\circ\mathsf{OR}_{(n/k)^{\frac{1}{d-1}}}$. Consequently, we have converted our original depth-d circuit into a new one where the first 2 layers are made from gates in $\mathsf{G}(k)$.

We now use the fact that any $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ can be expressed as a size $2^{O((n/k)^{\frac{1}{d-1}})}$ CNF or DNF to convert the remaining $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ gates to AND/OR gates while preserving the depth. Convert the third layer $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ gates to a DNF (OR of ANDs), and then collapse the 2nd and 3rd layer as they both consist solely of OR gates. The third layer now consist of AND gates, so replace the fourth layer $\mathsf{PAR}_{(n/k)^{\frac{1}{d-1}}}$ gates with a CNF (AND of ORs) to again induce a collapse of the consecutive AND layers. Repeat this procedure down to the bottom of the circuit.

Clearly after this procedure, all gates are in $\mathsf{G}(k)$ (in fact, all but the top gate are AND or OR). Notice that at each stage, we increased the circuit depth by 1 when plugging in the DNF/CNF, but then reduced the circuit depth when collapsing layers of the same gate type. Hence the final circuit is still of depth d. We already calculated the top depth-2 subcircuit is of size $2^{O((n/k)^{\frac{1}{d-1}})}$. Assuming we didn't collapse any gates, we would have replaced each of the $o(n^{1-\varepsilon})$ parity gates with a size-$2^{O((n/k)^{\frac{1}{d-1}})}$ circuit, so clearly the final circuit size is at most

$$2^{O((n/k)^{\frac{1}{d-1}})} + o(n/k) \cdot 2^{O((n/k)^{\frac{1}{d-1}})} \leq 2^{O((n/k)^{\frac{1}{d-1}})}.$$

Therefore at the end of this procedure, we get our desired circuit. ◀

We now show that not only is the size lower bound tight, but the average case correlation bound established in Theorem 33 as well.

▶ **Theorem 55.** *Assume $m \geq k^d$. There is a $\mathsf{GC}^0(k)$ circuit C of size $\leq m$ such that*

$$\mathbb{E}_x[(-1)^{C(x)+\mathsf{PAR}(x)}] \geq 2^{\Omega(k)} \cdot 2^{-O(n/(k+\log m)^{d-1})}$$

Proof. Let $M = \max\{k, c_d \log m\}$, where c_d is a constant such that the parity over $(c \log m)^{d-1}$ bits can be computed by an AC^0_d circuit of size $\leq m$. Split the input into $\lceil n/M^{d-1} \rceil$ blocks of size $\leq M^{d-1}$. If $M = c_d \log m$, each block can be calculated by an AC^0_d circuit of size m, and if $M = k$, each block can be calculated by a $\leq k^d \leq m$-size $\mathsf{GC}^0_{d-1}(k)$ circuit using a tree of PAR_ks. Now if $\lceil n/M^{d-1} \rceil = 1$, this circuit computes the parity of all n bits and we are done, so assume $\lceil n/M^{d-1} \rceil \geq 2$.

Join all the $\lceil n/M^{d-1} \rceil$ subcircuits by the gate G defined to compute parity if the Hamming weight of the input is at most k, and to equal 0 otherwise. Clearly $G \in \mathsf{G}(k)$. Therefore, if the subcircuits were constructed to be in AC^0_d, we can collapse the top two layers into one using Lemma 30 (similar to Theorem 54), giving us a $\mathsf{GC}^0_d(k)$ circuit. If the subcircuits were $\mathsf{GC}^0_{d-1}(k)$, we trivially get a $\mathsf{GC}^0_d(k)$ gate after adding G. In the case more than k of the $\lceil n/M^{d-1} \rceil$ input blocks have parity 1, our circuit will be constant, and thus will agree with parity about half the time. Let us crudely lower bound the correlation in this case to be 0. When $\leq k$ have parity 1, the top gate computes parity exactly. Therefore our correlation is simply the probability at most k of the input blocks have parity 1, which is simply

$$2^{-\lceil n/M^{d-1} \rceil} \sum_{i \leq k} \binom{\lceil n/M^{d-1} \rceil}{k} \geq 2^{-\lceil n/M^{d-1} \rceil} \cdot 2^{\Omega(k)} \geq 2^{\Omega(k)} \cdot 2^{-O(n/(k+\log m)^{d-1})}.$$

This gives our correlation lower bound as desired. ◀

A.3 Proof of the $\mathsf{GC}^0(k)$ Multi-Switching Lemma

We prove the multi-switching lemma here.

▶ **Theorem 56** (Proof of Theorem 28). *Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of $\mathsf{G}(k) \circ \mathsf{AND}_w$ circuits on $\{0,1\}^m$. Then*

$$\Pr_{\rho \sim R_p}[\mathcal{F}|_\rho \text{ does not have } r\text{-partial depth-}t \text{ } DT] \leq 4(64(2^k m)^{1/r} pw)^t$$

Proof. We follow the proof in [18] exactly, where the only difference is that the "Canonical Partial Decision Tree" (CPDT) will use the modified CDT we created in Algorithm 1, the definition of global witnesses (resp. global partial witnesses) will now use the witnesses (resp. partial witnesses) that we defined in Definition 22 (resp. Definition 24), and the "Global Witness Searcher" will run the modified witness searcher we created in Algorithm 2.

Consider the following CPDT procedure.

Algorithm 3 Canonical Partial Decision Tree.

Input: A list of $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuits $\mathcal{F} = \{F_1, \ldots, F_m\}$, black-box access to a string $\beta \in \{0,1\}^n$, and an auxiliary string $z \in \{0,1\}^n$.

initialize:
- $x \leftarrow (\star)^n$.
- $j \leftarrow 1$.
- counter $\leftarrow 0$.

while counter $< t$ **do**
- Find the smallest $i \geq j$ such that $\mathsf{DT}(F_i|_x) > w$. If no such i exists, exit the loop.
- $y \leftarrow (\star)^n$.
- $I \leftarrow \emptyset$.
- **while** $F_i|_{x \circ y}(\star)$ *is not constant and* counter $< t$ **do**
  - $C_{i,q} \leftarrow$ the term that $T_{F_i|_{x\circ y}}$ from Algorithm 1 will query.
  - $B_{i,q} \leftarrow$ the set of unknown variables in $C_{i,q}|_{x\circ y}$.
  - $y_{B_{i,q}} \leftarrow z_{B_{i,q}}$.
  - $I \leftarrow I \cup B_{i,q}$.
  - counter $\leftarrow$ counter $+ |B_{i,q}|$.
- **end**
- Query β_I, and set $x_I \leftarrow \beta_I$.
- $j \leftarrow i$.

end

return x

With this, we can define the following notion of a "global witness" to intuitively be a transcript on adversarially chosen inputs.

▶ **Definition 57.** *Let t, w be two integers. Consider a list of $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuits $\mathcal{F} = \{F_1, \ldots, F_m\}$. Suppose $\rho \in \{0,1,\star\}^n$ is a restriction. Let $(R, L_i, S_i, W_i, \beta_i)$ be a tuple, where*

- *$1 \leq R \leq \frac{t}{r}$ is an integer;*
- *$1 \leq L_1 \leq L_2 \leq \cdots \leq L_R \leq m$ is a list of R non-decreasing indices;*
- *$S_1 \ldots, S_R$ is a list of R integers such that $\sum_{i=1}^{R} S_i \in [t, t+w]$;*
- *$W_1, \ldots, W_R$ is a list of witnesses (as per Definition 22). For every $i \in [R]$, W_i has size S_i;*
- *$\beta_1, \ldots, \beta_R$ are R strings where $|\beta_i| = S_i$ for every $i \in [R]$.*

We call the tuple a (r,t)-global witness for ρ, if it satisfies the following.

1. *Set $\rho_1 = \rho$. W_1 is a S_1-witness for $F_{L_1}|_{\rho_1}$.*
2. *For every $i \geq 2$, let $I_{i-1} \subseteq [n]$ be the set of variables involved in W_{i-1}. Note that $|I_{i-1}| = S_{i-1}$ since the size of W_{i-1} is S_{i-1}. Identify β_{i-1} as a partial assignment in $\{0,1,\star\}^n$ where only the part $\beta_{i-1,I_{i-1}}$ is set and other coordinates are filled in with $\star$. Construct $\rho_i = \rho_{i-1} \circ \beta_{i-1}$. Then W_i is a S_i-witness for $F_{L_i}|_{\rho_i}$.*

The size of the global witness is defined as $\sum_{i=1}^{R} S_i$.

▶ **Lemma 58.** *Consider a list of* $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ *circuits* $\mathcal{F} = \{F_1, \ldots, F_m\}$*. Suppose* $\rho \in \{0,1,\star\}^n$ *is a restriction such that* $\mathcal{F}|_\rho$ *does not have* w*-partial depth-*t *decision tree. Then there exists an* (r,t)*-global witness for* ρ*.*

Proof. Same as the proof of Corollary 1 in [18]. ◀

We now define *partial global witnesses*, as there are far too many global witnesses to union bound over.

▶ **Definition 59.** *Let* t, w *be two integers. Consider a list of* $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ *circuits* $\mathcal{F} = \{F_1, \ldots, F_m\}$*. Suppose* $\rho \in \{0,1,\star\}^n$ *is a restriction. Let* $(R, L_i, S_i, P_i, \beta_i)$ *be a tuple, where*

- $1 \leq R \leq \frac{t}{r}$ *is an integer;*
- $1 \leq L_1 \leq L_2 \leq \cdots \leq L_R \leq m$ *is a list of* R *non-decreasing indices;*
- $S_1 \ldots, S_R$ *is a list of* R *integers such that* $\sum_{i=1}^{R} S_i \in [t, t+w]$*;*
- $P_1, \ldots, P_R$ *is a list of partial witnesses. For every* $i \in [R]$*,* P_i *has size* S_i*.*
- $\beta_1, \ldots, \beta_R$ *are* R *strings where* $|\beta_i| = S_i$ *for every* $i \in [R]$*.*

We call $(R, L_i, S_i, P_i, \beta_i)$ *a* (r,t)*-global partial witness for* ρ*, if we can complete* P_i *to get a witness* W_i *for every* $i \in [R]$*, such that* $(R, L_i, S_i, W_i, \beta_i)$ *is a global witness for* ρ*.*

By a simple induction, one can show the following claim.

▷ Claim 60. Given a global partial witness for ρ, there is exactly one way to complete it and get a global witness for ρ.

We now construct a global witness searcher that will reconstruct a global witness from a partial one using advice, and present it as Algorithm 4.

Algorithm 4 Global Witness Searcher.

Input: A list of DNFs $\mathcal{F} = \{F_1, \ldots, F_m\}$, a global partial witness $(R, L_i, S_i, P_i, \beta_i)$, and an advice $z \in \{0,1\}^n$.

initialize:

  $c \leftarrow 1$.

  $\rho^{(1)} \leftarrow \rho$.

while $c \leq R$ **do**

  Run Algorithm 2 on $(F_{L_c}, \rho^{(c)}, P_c, y)$. If it reports ERROR, report ERROR and terminate the procedure. Otherwise let W_c be the witness returned.

  $I_c \leftarrow$ the set of variables involved in W_c.

  Identify β_c as a partial assignment, where only β_{I_c} is fixed.

  $\rho^{(c+1)} \leftarrow \rho^{(c)} \circ \beta_c$.

  $c \leftarrow c + 1$.

end

return x

We note the following important lemma about the searcher, which we denote $\mathcal{S}$.

▶ **Lemma 61.** *Let* P *be a size* S *global partial witness. Say* ρ *is* good *if* P *is a global partial witness for* ρ*, then*

$$\Pr_z[\mathcal{S}(z, P) \text{ is a global witness for } \rho(\Lambda, z) | \rho \text{ is good}] = 2^{-S}$$

Proof. Same as proof of Lemma 8 in [18], but we instead appeal to Lemma 26 whenever Lyu's proof refers to Lemma 6. ◀

By this lemma, we have

$$\begin{aligned}\Pr_{\Lambda,z}[\rho(\Lambda,z) \text{ is good for } P] &= \mathbb{E}_\Lambda \frac{\Pr_z[\mathcal{S}(z,P) \text{ is a global witness for } \rho(\Lambda,z)]}{\Pr_z[\mathcal{S}(z,P) \text{ is a global witness for } \rho(\Lambda,z) | \rho \text{ is good}]} \\ &= 2^S \mathbb{E}_\Lambda \Pr_z[\mathcal{S}(z,P) \text{ is a global witness for } \rho(\Lambda,z)] \\ &\leq 2^S \mathbb{E}_z \Pr_\Lambda[\mathcal{S}(z,P) \text{ is a global witness for } \rho(\Lambda,z)] \\ &\leq (2p)^s \end{aligned} \tag{14}$$

where the last inequality follows from the fact that ρ needs to keep the S variables specified by the global witness alive in order to have any hope of being witnessed by it. By the $(t+w)$-wise p-boundedness of Λ, this happens with probability $\leq p^S$.

Finally, if we let N_S be the number of global witnesses of size S, we can see by using Lemma 58 and Claim 60, along with (14) that

$$\begin{aligned}\Pr_\rho[\mathcal{F}|_\rho \text{ has no } r\text{-partial depth-}t \ \mathsf{DT}] &\leq \sum_P \Pr_\rho[P \text{ is a global partial witness for } \rho] \\ &\leq \sum_{S=t}^{t+w} N_S (2p)^S \end{aligned} \tag{15}$$

We can upper bound N_S as follows.

- There are $\leq \frac{t}{r} \cdot \binom{m}{t/r} \leq 2m^{t/r}$ ways to pick (R, L_i)
- There are $\leq 2^S$ choices for (S_i) (since there are $\leq 2^{n-1}$ ways to write n as an ordered partition)
- From Theorem 20, we know that there are $(8w)^S 2^k$ partial witnesses of size S, giving a total amount of $\leq \prod_i (8w)^{S_i} 2^k = (8w)^S 2^{kR} \leq (8w)^S 2^{kt/r}$
- There are clearly $2^{\sum S_i} = 2^S$ possibilities for (β_i).

Combining all this tells us that $N_S \leq 2m^{t/r}(32w)^S 2^{kt/r}$. Hence, from (15), we deduce

$$\begin{aligned}\Pr_\rho[\mathcal{F}|_\rho \text{ has no } r\text{-partial depth-}t \ \mathsf{DT}] &\leq \sum_{S=t}^{t+w} N_S (2p)^S \\ &\leq m^{t/r} 2^{kt/r} \sum_{S=t}^{t+w} (32pw)^S \\ &\leq 4(64(2^k m)^{1/r} pw)^t \end{aligned}$$ ◀

▶ **Theorem 62** (Proof of Theorem 38). *Let $\mathcal{F} = \{F_1, \ldots, F_m\}$ be a list of size m $\mathsf{G}(k) \circ \{\mathsf{AND}, \mathsf{OR}\}_w$ circuits. Let (Λ, z) be a joint random variable such that*

- *Λ is a $(t+w)$-wise p-bounded subset of $[n]$*
- *Conditioned on any instance of Λ, z ε-fools CNF of size $\leq m^2$.*

Then

$$\Pr_{\Lambda,z}[\mathcal{F}|_{\rho(\Lambda,x)} \text{ has no } r\text{-partial depth-}t \ \mathsf{DT}] \leq 4(m2^k)^{t/r}(64pw)^t + (64wm)^{t+w}(2m)^{2kt/r} \cdot \varepsilon$$

Proof. All steps of the proof of Theorem 28 follow identically until we reach the step

$$\Pr_{\Lambda,z}[F|_{\rho(\Lambda,x)} \text{ has no } w\text{-partial depth-}t \text{ DT}]$$
$$= \Pr \sum_{(R,L,S_i,P_i,\beta_i)} \Pr_{\Lambda,z}[(R,L,S_i,P_i,\beta_i) \text{ is a global partial witness for } \rho(\Lambda,z)]$$

From Claim 60, we can deduce the event decomposes

$$\mathbb{1}\{(R,L_i,S_i,P_i,\beta_i) \text{ is a global partial witness for } \rho(\Lambda,z)\}$$
$$= \sum_{W_i:\text{completion of } P_i} \mathbb{1}\{(R,L_i,S_i,W_i,\beta_i) \text{ is a global witness for } \rho(\Lambda,z)\}.$$

For a fixed Λ and (R,L_i,S_i,W_i,β_i), we can let

$$h_\Lambda^{(R,L_i,S_i,W_i,\beta_i)}(z) = \mathbb{1}\{(R,L_i,S_i,W_i,\beta_i) \text{ is a global witness for } \rho(\Lambda,z)\}.$$

We will now show that h is a predicate computable by a small size CNF, so that z will fool it. h is true iff W_i is a witness for $F_{L_i}|_{\rho(\Lambda,z)\circ\beta_1\ldots\beta_{i-1}}$ for all $1 \le i \le R$. Now for each i, one can verify $W_i = (r', \ell_i, s_i, B_i, \alpha_i)$ is a witness by a size m CNF as follows.

- For all $j < \ell_1$, C_j is falsified by $\rho(\Lambda,z)$. This is true iff $(\neg C_1) \wedge (\neg C_2) \wedge \cdots \wedge (\neg C_{\ell_1})$. Notice $\neg C_i$ becomes an OR clause by De Morgan's Law
- C_{j_1} is satisfied, which is an AND of variables.
- C_j is falsified by $\rho(\Lambda,z)\circ\alpha_1$ for $\ell_1 < j < \ell_2$, which is true iff $(\neg C_{\ell_1+1}) \wedge \cdots \wedge (\neg C_{\ell_2-1})$. Each $\neg C_i$ is an OR clause
- and so on and so forth until we verify $C_{j_{r'}}$.

Each bullet points gives a disjunction of (maybe trivial) conjunctions, so for all bullet points to hold, we simply take the AND of all of them, resulting in a CNF whose size is bounded by m (since our CNF is essentially F_{L_i} but with some gates and negations changed. Hence, if we want to verify W_i simultaneously over all i, we take the AND of all $R \le m$ CNFs to get a size m^2 CNF. Hence, z ε-fools h. From Theorem 20, we know over a uniform string x,

$$\sum_{(R,L_i,S_i,W_i,\beta_i)} \mathbb{E}_\Lambda \mathbb{E}_x h_\Lambda^{(R,L_i,S_i,W_i,\beta_i)}(x) \le 4(m2^k)^{t/r}(64pw)^t$$

Therefore, over z, we have

$$\begin{aligned}\Pr_{\Lambda,z}[F|_{\rho(\Lambda,x)} \text{ has no } w\text{-partial depth-}t \text{ DT}] &= \sum_{(R,L_i,S_i,W_i,\beta_i)} \mathbb{E}_\Lambda \mathbb{E}_z h_\Lambda^{(R,L_i,S_i,W_i,\beta_i)}(z) \\ &\le \sum_{(R,L_i,S_i,W_i,\beta_i)} \mathbb{E}_\Lambda(\varepsilon + \mathbb{E}_x h_\Lambda^{(R,L_i,S_i,W_i,\beta_i)}(x)) \\ &\le 4(m2^k)^{t/r}(64pw)^t + \sum_{(R,L_i,S_i,W_i,\beta_i)} \varepsilon.\end{aligned}$$

The number of tuples (R,L_i,S_i,W_i,β_i) can be bounded as follows.

- From the proof of Theorem 56, we know there are $2m^{t/r}2^{kt/r}(32w)^S$ many global partial witnesses of size S.
- We now multiply by the number of (ℓ_i) possible for each partial P_i of size S_i, which is at most $\binom{m}{t+k} < m^{S_i+k}$. Hence, the total number of (W_i) over all $1 \le i \le R$ is

$$\prod_i m^{S_i+k} = m^{S+kR} \le m^{S+kt/r}$$

Hence the total number of size-S tuples is upper bounded by $2m^{t/r}2^{kt/r}(32w)^S m^{S+kt/r} \leq (32mw)^S(2m)^{2kt/r}$. Summing over all $t \leq S \leq t+w$ gives us a grand total of

$$\sum_{S=t}^{t+w}(32mw)^S(2m)^{2kt/r} \leq (64mw)^S(2m)^{2kt/r}.$$

Therefore,

$$\Pr_{\Lambda,z}[F|_{\rho(\Lambda,x)} \text{ has no } r\text{-partial depth-}t\ \mathsf{DT}] \leq 4(m2^k)^{t/r}(64pw)^t + (64mw)^{t+w}(2m)^{2kt/r} \cdot \varepsilon. \blacktriangleleft$$

Criticality of AC^0-Formulae

Prahladh Harsha
Tata Institute of Fundamental Research, Mumbai, India

Tulasimohan Molli
Tata Institute of Fundamental Research, Mumbai, India

Ashutosh Shankar
Tata Institute of Fundamental Research, Mumbai, India

Abstract

Rossman [In *Proc. 34th Comput. Complexity Conf.*, 2019] introduced the notion of *criticality*. The criticality of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is the minimum $\lambda \geq 1$ such that for all positive integers t and all $p \in [0,1]$,

$$\Pr_{\boldsymbol{\rho} \sim \mathcal{R}_p} [\mathsf{DT}_{\mathrm{depth}}(f|_{\boldsymbol{\rho}}) \geq t] \leq (p\lambda)^t,$$

where $\mathcal{R}_p$ refers to the distribution of p-random restrictions.

Håstad's celebrated switching lemma shows that the criticality of any k-DNF is at most $O(k)$. Subsequent improvements to correlation bounds of AC^0-circuits against parity showed that the criticality of any AC^0-*circuit* of size S and depth $d+1$ is at most $O(\log S)^d$ and any *regular* AC^0-*formula* of size S and depth $d+1$ is at most $O(\frac{1}{d} \cdot \log S)^d$. We strengthen these results by showing that the criticality of *any* AC^0-formula (not necessarily regular) of size S and depth $d+1$ is at most $O(\frac{\log S}{d})^d$, resolving a conjecture due to Rossman.

This result also implies Rossman's optimal lower bound on the size of any depth-d AC^0-formula computing parity [Comput. Complexity, 27(2):209–223, 2018.]. Our result implies tight correlation bounds against parity, tight Fourier concentration results and improved #SAT algorithm for AC^0-formulae.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases AC^0 circuits, AC^0 formulae, criticality, switching lemma, correlation bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.19

Related Version *arXiv Version*: https://arxiv.org/abs/2212.08397

Funding Research supported by the Department of Atomic Energy, Government of India, under project 12-R&D-TFR-5.01-0500.
Prahladh Harsha: Research supported in part by the Google India Research Award.
Tulasimohan Molli: Research partially supported through the MATRICS grant MTR/2019/001226 (PI: Jaikumar Radhakrishnan) of the Science and Engineering Research Board (SERB), Department of Science and Technology, Government of India.

Acknowledgements The first and the second authors spent several years thinking about this problem and we are indebted to several people along the way. First and foremost, we thank Jaikumar Radhakrishnan and Ramprasad Saptharishi for spending innumerable hours in the various stages of this project going over several failed attempts and potential proofs and giving us very helpful feedback along the way. We are grateful to Ben Rossman for discussions in the early stages of this project as well as pointing out an error in the previous version of this proof. We would also like to thank Srikanth Srinivasan, Siddharth Bhandari, Yuval Filmus and Mrinal Kumar for their comments and feedback.

1 Introduction

Understanding the power of various models of computation is the central goal of complexity theory. With respect to small-depth AND-OR circuits, the early works of Furst, Saxe and Sipser [4], Sipser [15], Ajtai [1], Yao [18] and Håstad [5] using *random restrictions* and Razborov [9] and Smolensky [16] using the *polynomial method* laid out a promising direction.

Furst, Saxe and Sipser [4] and Ajtai [1] independently proved that the parity function requires super-polynomial sized constant depth AND-OR circuits to compute it. This was then later improved by Yao [18] and Håstad [5] who proved that any depth-$(d+1)$, AND-OR circuit computing parity on n bits requires size $2^{n^{\Theta(1/d)}}$. The pièce de résistance of these results is the *switching lemma* method introduced by Furst, Saxe and Sipser [4]. Informally stated, it states that any k-DNF[1] reduces (aka *switches*) to a low-width CNF with high probability, when acted upon by a p-random restriction. Very soon (in Håstad's paper itself [5]), it was discovered that it was more convenient and useful to state the switching lemma in terms of the depth of decision trees. This leads us to Håstad's switching lemma, one of the most celebrated theorems in theoretical computer science. Let f be a k-DNF and $\mathcal{R}_p$ denote the distribution of p-random restrictions ($p \in [0,1]$) where each variable independently is left unrestricted with probability p and otherwise set uniformly to 0 or 1. Then,

$$\Pr_{\rho \sim \mathcal{R}_p}[\mathsf{DT}_{\text{depth}}(f|_\rho) \geq s] \leq (5pk)^s \ .$$

Despite the immense success of these methods in proving optimal lower bounds for small-depth AND-OR circuits and related models, they did not help in understanding limits of considerably stronger computational models. It was soon discovered that "other techniques" are needed to tackle these stronger models and this was made formal in the *natural proof* approach by Razborov and Rudich [11]. About a decade ago, interest in an *improved* switching lemma was revived while trying to understand optimal correlation bounds of small depth AND-OR circuits with the parity function. While the early results of Ajtai [1] were only able to show a correlation bound of $\exp(-\Omega(n^{1-\varepsilon}))$, Beame, Impagliazzo and Srinivasan [3] proved a considerably smaller correlation bound of $\exp(-\Omega(n/2^{2d(\log S)^{4/5}}))$ for depth-d AND-OR circuits of size S with the parity function over n bits. This was then improved by Impagliazzo, Matthews and Paturi [7] and Håstad [6] who proved the optimal correlation bound of $\exp(-\Omega(n/(\log S)^d))$ for depth-$(d+1)$ AND-OR circuits of size S with the n-bit parity function. Håstad proved this optimal correlation bound by proving the *multi-switching lemma*, a significant strengthening of his earlier switching lemma. The multi-switching lemma is best described in terms of *criticality*, a notion introduced subsequently by Rossman [14].

The *criticality* of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is the minimum $\lambda \geq 1$ such that

$$\Pr_{\rho \sim \mathcal{R}_p}[\mathsf{DT}_{\text{depth}}(f|_\rho) \geq s] \leq (p\lambda)^s.$$

Thus, Håstad's switching lemma in terms of criticality states that k-DNFs (and k-CNFs) have criticality $O(k)$. The Multi-switching lemma (stated using Rossman's reformulation in terms of criticality [12, 14]) states that a depth-$(d+1)$ AND-OR circuit of size S has criticality $O(\log S)^d$. Bounds on criticality besides implying optimal correlation bounds against parity yield improved decision tree size bounds, ℓ_2 and ℓ_1-Fourier concentration bounds (as noted by Rossman [14]).

[1] A k-DNF is a Boolean formula in disjunctive normal form (DNF) in which each term has at most k literals. A k-CNF is defined similarly.

AND-OR formulae

What is the criticality of depth-$(d+1)$ AND-OR formulae of size S? Recall that the difference between circuits and formulae is that gates are allowed to have fan-out more than 1 in circuits where each gate in a formula has fan-out at most 1. The standard transformation from depth-$(d+1)$ AND-OR circuits to depth-$(d+1)$ AND-OR formulae transforms size-S circuits to size-S^d formulae. If one believes that the "best" AND-OR formulae for parity is obtained via the above transformation, then the "expected" bounds for AND-OR formulae would be the ones for AND-OR circuits with the parameter S replaced by $S^{1/d}$. This expectation is substantiated by the following two results of Rossman. Rossman [13] showed that any depth-$(d+1)$ AND-OR formula (not necessarily regular) that computes the n-bit parity requires size at least $S = 2^{\Omega(d(n^{1/d}-1))}$ (corresponding to the analogous result for AND-OR circuits due to Håstad that $S = 2^{\Omega(n^{1/d})}$). In a different work, Rossman [14] showed that if one restricts to *regular* formulae, formulae in which all gates at the same height have equal fan-in, then depth $(d+1)$ *regular* AND-OR *formulae* of size S have criticality $O\left(\log S^{1/d}\right)^d = O\left(\frac{\log S}{d}\right)^d$ (corresponding to the analogous criticality bound of $O\left(\log S\right)^d$ for AND-OR circuits). Given these results, Rossman conjectured [14, Conjecture 1] that any (not necessarily regular) depth $(d+1)$ AND-OR formula of size S has criticality $O\left(\frac{\log S}{d}\right)^d$. This conjecture, if true, besides proving optimal criticality bounds for AND-OR formulae, would yield a common strengthening and unification of all the aforementioned results (criticality bounds for regular AND-OR formulae and arbitrary AND-OR circuits as well as tight AND-OR formula lower bounds against parity). The main result of this paper is a postive resolution of Rossman's conjecture that any (not necessarily regular) depth $(d+1)$ AND-OR formula of size S has criticality $O\left(\frac{\log S}{d}\right)^d$. More precisely,

▶ **Theorem 1.1.** *Let F be an AND-OR formula of depth $d+1$ and size at most S, then for any $p \in [0,1]$*

$$\Pr_{\rho\sim\mathcal{R}_p}\left[\mathsf{DT}_{depth}(F|_\rho) \geq s\right] \leq \left(p \cdot O\left(32^d\left(\frac{\log S}{d}+1\right)^d\right)\right)^s.$$

As an immediate corollary of the above criticality result and [14, Theorem 14], we get the following results for general AND-OR formulae of depth $(d+1)$. Rossman had proved similar results for regular AND-OR formulae of depth $(d+1)$ [14].

▶ **Corollary 1.2.** *Let $f\colon \{0,1\}^n \to \{0,1\}$ be computable by an AND-OR formula of depth $d+1$ and size at most S. Then*

1. *Decision tree size bounds:* $\mathsf{DT}_{size}(f) \leq O\left(2^{\left(1-1/O\left(\frac{1}{d}\log S\right)^d\right)n}\right)$,
2. *Correlation bound with parity:* $\mathrm{Cor}(f,\oplus_n) \leq O\left(2^{-n/O\left(\frac{1}{d}\log S\right)^d}\right)$,
3. *Degree bounds:* $\Pr_{\rho\sim\mathcal{R}_p}\left[\deg(F|_\rho) \geq s\right] \leq \left(p\cdot O\left(32^d\left(\frac{1}{d}\log S+1\right)^d\right)\right)^s$.
4. *ℓ_2-Fourier concentration (Linial-Mansour-Nisan [8]):* $\sum_{|S|\geq k}\widehat{f}(S)^2 \leq 2e\cdot e^{-k/O\left(\frac{1}{d}\log S\right)^d}$,
5. *ℓ_1-Fourier concentration (Tal [17]):* $\sum_{|S|=k}\left|\widehat{f}(S)\right| \leq O\left(\frac{1}{d}\log S\right)^{dk}$.

As indicated before, Theorem 1.1 unifies all previous results for AC^0 circuits and formulae in this context. It also yields satisfiability results for AC^0-formulae along the lines of the Impagliazzo, Matthews and Paturi result [7] (see Section 6 for more details).

Proof Overview

The proof of Theorem 1.1 is an adaptation of Rossman's proof in the regular case [14] to the general setting. Like in Rossman's proof in the regular setting, we construct a canonical decision tree (CDT) for the depth-d formula. This CDT under a restriction is constructed in an inductive fashion by progressively refining the restriction. The reason we get the right bound for AND-OR formulae is because our proof (in particular, the construction of the CDT) is top-down respecting the structure of the formula. This is also the case with Rossman's formula bound [13] and Rossman's proof in the regular setting [14]. Note that the CDT constructed from the multi-switching lemma is bottom-up on the other hand. Unlike Rossman's proof [14], our proof can handle *general* formulae and not just regular ones, due to the use of downward-closed sets. We explain this in more detail below.

The main theorem is proved via the following statement (see Lemma 5.6 for the exact statement), which we prove inductively. For any s-length bitstring $a \in \{0,1\}^s$,

$$\Pr_{\boldsymbol{\rho}}\left[\text{There exists path in } \mathrm{CDT}(F,\boldsymbol{\rho}) \text{ labelled by instruction-set } a \mid \boldsymbol{\rho} \in \mathcal{T}\right] \leq (p \cdot \lambda(F))^s,$$

where $\mathcal{T}$ is any family of downward-closed set of restrictions and $\lambda(F)$ is the desired criticality bound that we wish to prove. The crucial difference from Rossman's proof in the regular setting is we prove the above statement subject to $\boldsymbol{\rho}$ belonging to any downward-closed set. As in Rossman's proof, the event "There exists a path ...." is broken into several subevents, $\mathcal{E}_t$ for t ranging over a polynomially large set and each of the events $\mathcal{E}_t$ is typically a conjunction of 3 events $\mathcal{A}_t, \mathcal{B}_t$ and $\mathcal{C}_t$. The required probability can then bounded by an expression as follows:

$$\sum_t \Pr\left[\mathcal{A}_t \cap \mathcal{B}_t \cap \mathcal{C}_t \mid \mathcal{T}\right] = \sum_t \Pr\left[\mathcal{A}_t \mid \mathcal{T}\right] \cdot \Pr\left[\mathcal{B}_t \mid \mathcal{A}_t \cap \mathcal{T}\right] \cdot \Pr\left[\mathcal{C}_t \mid \mathcal{A}_t \cap \mathcal{B}_t \cap \mathcal{T}\right].$$

The advantage of using conditioning is that some of these intermediate probabilities (c.f., $\Pr\left[\mathcal{C}_t \mid \mathcal{A}_t \cap \mathcal{B}_t \cap \mathcal{T}\right]$) can be bound using the inductive assumption provided the conditioned events are themselves downward-closed. The events $\mathcal{A}_t$ and $\mathcal{B}_t$ are chosen such that this is indeed the case. The sum over t is then handled via convexity. The use of downward-closed sets to prove the inductive claim is inspired from Håstad's use of downward-closed sets in his proof of the multi-switching lemma [6]. However, the situation for depth-d formulae is considerably more involved than the DNF/CNF setting and both the choice of the events as well as bounding these conditional probabilities require considerable care and subtlety (see Section 4 and Claim 5.8). The use of downward-closed sets considerably simplifies the proof and yields an arguably simpler proof of the criticality bound, even in the regular setting [14]. While our theorem yields both the criticality result for AND-OR circuits [6] and Rossman's optimal AND-OR formula lower against parity [13], the proof is quite different from the corresponding proofs and more along the lines of Rossman's proof in the regular setting with the crucial additional element being the use of downward-closed sets.

Organization

The rest of the paper is organized as follows. We begin with some preliminaries in Section 2, where we recall the standard notions of restrictions, decision trees and introduce variants of these notions such as restriction trees etc., which will be of use later. We then define canonical decision trees for depth-d formulae in Section 3, identical to the corresponding notion in the regular setting due to Rossman [14]. We then demonstrate the downward-closedness of some properties related to CDTs in Section 4 and finally prove the main theorem in Section 5. In Section 6, we use the main lemma to give a randomized #SAT algorithm for arbitrary AC^0 formulae generalizing the corresponding algorithm due to Rossman in the regular setting [14].

2 Preliminaries

For a positive integer $n \in \mathbb{N}$, $[n]$ refers to the set $\{1, 2, \ldots, n\}$. All logarithms in this paper are to base 2.

While studying distributions D over some finite set Σ, we will use bold letters (i.e., $\boldsymbol{\sigma}$) to distinguish a random sample according to D from a fixed element $\sigma \in \Sigma$. Given any distribution D on a finite set Σ, we let $\mu_D \colon S \to [0,1]$ denote the corresponding probability mass function (i.e., $\mu_D(\sigma) = \Pr_{\boldsymbol{\sigma} \sim D}[\boldsymbol{\sigma} = \sigma]$). We will drop the subscript D from μ_D typically.

We begin by recalling the definition of an AC^0-formula.

▶ **Definition 2.1** (AC^0-formulae). *Let d be a non-negative integer. Let V be a set of variable indices. An* AC^0-formula *F of depth-d over variables V is (inductively) defined as follows: A depth-0 formula is the constant 0 or 1 or a literal x_v or $\neg x_v$ where v is a variable index. For $d \geq 1$, a depth-d AC^0-formula is either an OR-formula or an AND-formula which are defined below. A depth-d OR-formula is of the form $F_1 \vee F_2 \vee \cdots \vee F_m$ where the F_i's are either depth-d' AND-formulae for some $1 \leq d' < d$ or depth-0 formulae. A depth-d AND-formula $F = F_1 \wedge F_2 \cdots \wedge F_m$ is defined similarly.*

Depth-1 AND-formulae and OR-formulae are usually referred to as terms and clauses respectively, while depth-2 AND-formulae and OR-formulae are called DNFs and CNFs respectively.

The size of a formula is given by the number of depth-1 sub-formulae[2]. More precisely, $\mathrm{size}(F)$ *is inductively defined as*

$$\mathrm{size}(F) := \begin{cases} 0 & \text{if } \mathrm{depth}(F) = 0 \\ 1 & \text{if } \mathrm{depth}(F) = 1 \\ \sum_{i=1}^{m} \mathrm{size}(F_i) & \text{if } F = F_1 \vee F_2 \vee \cdots \vee F_m \text{ or } F_1 \wedge F_2 \cdots \wedge F_m. \end{cases}$$

The variable index set V of a given formula F unless otherwise specified is always assumed to be $[n]$.

We will sometimes identify a formula F with the Boolean function it computes. We say "$F \equiv 1$" if this Boolean function is a tautology and "$F \equiv 0$" if it is a contradiction.

2.1 Restrictions, Decision Trees and Restriction Trees

We will be chiefly concerned with restrictions.

▶ **Definition 2.2** (restriction). *Given a variable index set V, a* restriction *ρ is a function $\rho \colon V \to \{0, 1, *\}$ or equivalently a partial function from V to $\{0, 1\}$. We refer to the domain of this partial function as* $\mathrm{dom}(\rho)$ *and the remaining set of unrestricted variables, namely $V \setminus \mathrm{dom}(\rho)$, as* $\mathrm{stars}(\rho)$.

We say that two restrictions ρ_1 and ρ_2 are consistent if for every $v \in \mathrm{dom}(\rho_1) \cap \mathrm{dom}(\rho_2)$, we have $\rho(v_1) = \rho(v_2)$.

We can define a partial ordering among restrictions as follows: we say $\rho_1 \preccurlyeq \rho_2$ if (1) $\mathrm{stars}(\rho_1) \subseteq \mathrm{stars}(\rho_2)$ *and (2) ρ_1 and ρ_2 are consistent. In words, ρ_1 only "sets more variables" than ρ_2. Sometimes, we will only be interested in this order with respect to a particular subset T of the variable index set V. In this case, we say*

$$\rho_1 \preccurlyeq_T \rho_2 \text{ if (1) } \mathrm{stars}(\rho_1) \cap T \subset \mathrm{stars}(\rho_2) \cap T \text{ and (2) } \rho_1 \text{ and } \rho_2 \text{ are consistent.}$$

[2] Traditionally, the size is defined by the number of leaves or depth-0 formulas but for this paper, it would be more convenient to work with this definition.

Given a formula F and a restriction ρ, the restricted formula $F|_\rho$ refers to the formula obtained by relabeling literals involving variable indices in $\mathrm{dom}(\rho)$ *according to ρ (we make no further simplification to the formula). Given two consistent restrictions ρ_1, ρ_2, $F|_{\rho_1,\rho_2}$ refers to the formula $(F|_{\rho_1})|_{\rho_2}$ (which is identical to $(F|_{\rho_2})|_{\rho_1}$).*

It will sometimes be convenient to consider an ordering among the variables in the domain of a restriction, especially when studying restrictions arising from decision trees.

▶ **Definition 2.3** (ordered restriction)**.** *An* ordered restriction *on a variable set V is a sequence of the form $\alpha = (x_{v_1} \to b_1, \ldots, x_{v_t} \to b_t)$ where $t \in \mathbb{N}$, $b_i \in \{0,1\}$, and $v_1, \ldots, v_t$ are distinct elements of V. We will use* $\mathrm{dom}(\alpha)$ *to refer to the set $\{v_1, \ldots, v_t\}$. (We will typically use α or β for ordered restrictions.)*

Any ordered restriction inducesrestriction ρ with $\mathrm{dom}(\rho) = \{v_1, \ldots, v_t\}$*. Similarly, given a restriction ρ on V, and an ordering on* $\mathrm{dom}(\rho)$*, we have a natural representation of ρ as an ordered restriction on V.*

The following is the standard definition of a decision tree except that we allow the internal nodes of the tree to have (out-)degree either 1 or 2.

▶ **Definition 2.4** (decision tree)**.** *A* decision tree *is a a finite rooted binary tree where*

- *each internal node is labelled by a variable, has one or two children and the edges to its children have distinct labels from the set $\{0,1\}$,*
- *the leaves are labelled by 0 or 1, and*
- *the variables appearing in any root-to-leaf path are distinct.*

For each node v (including leaf node), the root-to-node path in the decision tree naturally corresponds to an ordered restriction, which we denote by α_v (this restriction is non-trivial for every non-root node).

The depth *of a decision tree Γ, denoted by* $\mathrm{depth}(\Gamma)$*, is defined as the maximum number of degree-2 nodes along any root-to-leaf path in T. Note that this may be shorter than the length of the corresponding ordered restriction, which includes the degree-1 nodes also.*

A decision tree is said to compute a Boolean function $F\colon \{0,1\}^V \to \{0,1\}$ under a restriction ρ if the following conditions hold

- *any internal vertex labelled by a variable index, say v, that is in* $\mathrm{dom}(\rho)$ *has degree one, with the edge to the only child labelled with $\rho(v)$,*
- *any internal vertex labelled by a variable index in* $\mathrm{stars}(\rho)$ *has degree two and*
- *for every leaf v, we have $F|_{\rho,\alpha_v} \equiv label(v)$.*

An "honest-to-god" decision tree (with all internal nodes having degree two) can be obtained from the above decision tree by contracting the degree-1 edges. However, we will find it convenient to keep this information about degree-1 nodes while constructing decision trees for functions under a restriction. Note that if a decision tree T computes a function F under the restriction ρ, then the contracted decision tree T' computes the function $F|_\rho$.

To prove the criticality bound for a given formula F, we construct a canonical decision tree (CDT) for F under a (random) restriction ρ. This CDT is constructed in an inductive fashion by constructing the CDT's for F's sub-formulae first and then using these CDT's to construct F's CDT. While doing so, we progressively refine the restriction so that the final restriction under which the CDT is constructed is the target restriction ρ. This naturally leads us to the notion of *restriction trees*, which is essentially a family of restrictions, one for each sub-formula of a given formula, such that the restrictions get refined as we move from child to parent in the formula tree.

▶ **Definition 2.5** (restriction tree). *Let F be a formula on the variable index set V and T_F the set of all sub-formulae of F. The elements of T_F have a natural bijection with the underlying formula tree of F. A restriction tree for F, denoted by $\tilde{\rho}$, associates a restriction with each node in T_F, formally $\tilde{\rho}\colon T_F \to \{0,1,*\}^V$, such that for $G, H \in T_F$ where G is a sub-formula of H, we have $\tilde{\rho}(H) \preccurlyeq \tilde{\rho}(G)$. In other words, the sequence of restrictions on any leaf-to-root path sets increasingly more variables as we approach the root.*

For any sub-formula G of F, we let $\tilde{\rho}|_G$ denote the restriction of $\tilde{\rho}$ to the set T_G of sub-formulae of G.

We will use the "tilde" notation to distinguish between restrictions ρ and restriction trees $\tilde{\rho}$. Observe that, by definition, every restriction ρ in a restriction tree $\tilde{\rho}$ corresponding to a formula F satisfies $\rho \preccurlyeq \tilde{\rho}(F)$ and are hence consistent with each other.

2.2 Representation of restrictions and restriction trees

Recall that a restriction ρ is a partial function from the set V of variables to $\{0,1\}$. Sometimes (especially when dealing with *random restrictions*), it will be convenient to work with a (redundant) representation of ρ given by the pair (σ, S) where $\sigma\colon V \to \{0,1\}$ is a *global* assignment consistent with ρ and $S = \mathrm{stars}(\rho)$. Note this is representation is redundant as we only need $\sigma|_{\overline{S}}$ to specify ρ. When sampling restrictions, it will be easier to sample the pair (σ, S) from some distribution and set $\rho := \rho_{(\sigma,S)}$ to be the restriction given by

$$\rho_{(\sigma,S)}(v) = \begin{cases} \sigma(v) & \text{if } v \notin S, \\ * & \text{if } v \in S. \end{cases}$$

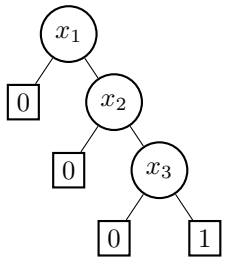
This representation naturally extends to restriction trees $\tilde{\rho}\colon T_F \to \{0,1,*\}$ which are given by a pair $(\sigma, \tilde{S})$ where $\sigma\colon V \to \{0,1\}$ is a global assignment consistent with all the restrictions in the restriction tree and $\tilde{S}\colon T_F \to 2^V$ is defined as $\tilde{S}(G) := \mathrm{stars}(\tilde{\rho}(G))$. Notice that any $\tilde{S}$ satisfies the monotonicity property that if G is a sub-formula of H in T_F, we have $\tilde{S}(H) \subseteq \tilde{S}(G)$. Given any such $\tilde{S}$ that satisfies the monotonicity property and a global assignment $\sigma\colon V \to \{0,1\}$, the corresponding restriction tree $\tilde{\rho}_{(\sigma,\tilde{S})}$ is given by $\tilde{\rho}_{(\sigma,\tilde{S})}(G) := \rho_{(\sigma,\tilde{S}(G))}$ for all $G \in T_F$.

3 Canonical decision tree

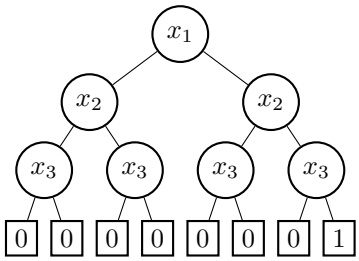
In this section, we construct a canonical decision tree (CDT) for a formula F. This definition is identical to Rossman's definition [14, Definition 19] (except that Rossman defines it completely in terms of ordered restrictions while we define it using decision trees which have both degree-1 and degree-2 internal nodes).

Let us first recall the CDT construction for DNFs in the proof of Håstad's classical switching lemma [2, 10, 6]. Let $F = T_1 \vee \cdots \vee T_m$ be a DNF and ρ a restriction on the variables of F. To construct $\mathrm{CDT}(F, \rho)$ we do the following:

1. Find the first term T (from left to right), not forced to 0 by ρ. If there is no such term, return the tree comprising of a single leaf node labelled 0.
2. If $T|_\rho \equiv 1$, return the tree comprising of a single leaf node labelled 1.
3. Let Y be the set of ρ-unrestricted variables in T. Let Γ be the $\mathrm{CDT}(T, \rho)$ constructed from the complete balanced binary tree of depth $|Y|$ indexed by the variables of Y and labelling the $2^{|Y|}$ leaves appropriately.
4. For each leaf v of Γ, inductively replace v with $\mathrm{CDT}(F|_{\alpha_v}, \rho)$ where α_v is the (ordered) restriction corresponding to leaf v.



(a) Optimal DT for $x_1 \wedge x_2 \wedge x_3$.



(b) Completed balanced DT for $x_1 \wedge x_2 \wedge x_3$.

Figure 1 Illustration of $\mathsf{DT}(x_1 \wedge x_2 \wedge x_3)$ used in the CDT construction in the proof of Håstad's Switching Lemma.

The construction of CDTs for depth-d formulae will be inspired by the above CDT construction for DNFs. Note that in Item 3, we used a complete binary tree instead of the best decision tree for the term T (see Figure 1). The rationale for doing this is because while proving the switching lemma, we wanted to attribute a 0-leaf in Γ to a 1-leaf which shares the same set of variables. We will need a similar property in our construction. To this end, we perform a balancing operation which ensures that every 0-leaf has a corresponding 1-leaf such that the two associated ordered restrictions share the same set of variables (this is the 0-balancing operation defined below. The 1-balancing operation is similar with the roles of 0 and 1 reversed).

3.1 0-Balancing and 1-Balancing

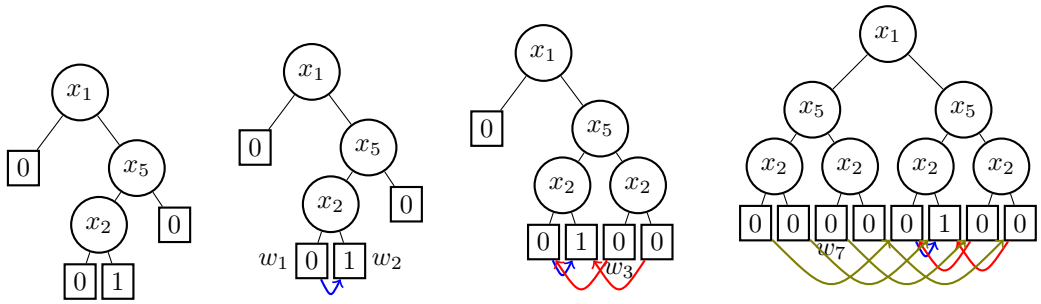
Given a decision tree Γ for a Boolean function F, the 0-balanced version Γ' is constructed as follows. We first pull-up the zeros, in other words, if there is any subtree all of whose leaves are labelled 0, we contract the entire subtree to a single leaf node labelled 0. The construction then proceeds in d rounds where d is the length of the longest root-to-leaf path in Γ (note this is not necessarily the depth of Γ due to the presence of degree-1 nodes). This process leaves the 1-leaves in Γ unaltered. As we proceed, we also construct a map assoc which associates each leaf (both 0 and 1 leaves) in Γ' with a 1-leaf in Γ'. To begin with, this map assoc associates each 1-leaf to itself (i.e, if u is a 1-leaf, then $\mathrm{assoc}(u) = u$).

In the i^{th} round, we consider all 0-leaves in Γ at distance $(d - i)$ from the root. Let u be one such 0-leaf and T_u the subtree rooted at the sibling of u. Observe that T_u necessarily has some leaf labelled 1, else the entire subtree rooted at the parent of u would have been contracted to a single leaf node labelled 0. We then mirror the entire subtree T_u at the leaf node u and relabel all the leaves of this mirrored subtree with 0. These are the 0-leaves of Γ'. For each such newly created 0-leaf w (in the mirrored subtree T_u), let w' be the corresponding leaf in the tree T_u. Set $\mathrm{assoc}(w) \leftarrow \mathrm{assoc}(w')$.

See Figure 2 for an illustration of the 0-balancing process. Observe that if we 0-balance the best decision tree for a term, we obtain the complete balanced tree (see Figure 1).

At the end of this process, observe that Γ is transformed into another decision tree Γ' such that the following hold.

- If Γ computes a function F under some restriction ρ, so does Γ'.
- The 1-leaves in Γ' are in 1-1 correspondence with the 1-leaves in Γ. Furthermore, the two 1-leaves (the one in Γ and its associated 1-leaf in Γ') correspond to identical ordered restrictions.
- Every 0-leaf w in Γ' has an associated 1-leaf in Γ' given by $\mathrm{assoc}(w)$. Furthermore, the corresponding ordered restrictions (namely $\alpha_w^{\Gamma'}$ and $\alpha_{\mathrm{assoc}(w)}^{\Gamma'}$) share the same set of variables which are queried in the same order along both these root-to-leaf paths.



(a) initial.

(b) first round $assoc(w_1) \leftarrow w_2$.

(c) second round $assoc(w_3) \leftarrow assoc(w_1) \leftarrow w_2$.

(d) third round $assoc(w_7) \leftarrow assoc(w_3) \leftarrow assoc(w_1) \leftarrow w_2$.

Figure 2 Illustration of 0-balancing process.

Let us now try to understand what are the 0-leaves constructed in the 0-balancing process. Let w be any 0-leaf in Γ' and $w' = \text{assoc}(w)$ be the corresponding 1-leaf. Furthermore, let $\alpha := \alpha_w^{\Gamma'} = (v_1 \mapsto c_1, \ldots, v_t \mapsto c_t)$ and $\beta := \alpha_{w'}^{\Gamma'} = (v_1 \mapsto d_1, \ldots, v_t \mapsto d_t)$. First, we must have that $\text{dom}(\alpha) = \text{dom}(\beta)$ and that the variables in this common domain must be queried in the same order. Furthermore, whenever α differs from β, the ordered restriction formed by following β up to the step prior to this particular point of disagreement and then taking a step according to α must cause the formula $F|_\rho$ to evaluate to 0. This occurs as the mirroring operation is performed only at such nodes. More precisely, let α_i denote the ordered restriction $(v_1 \mapsto d_1, v_2 \mapsto d_2, \ldots, v_{i-1} \mapsto d_{i-1}, v_i \mapsto c_i)$. Note, α_i is the ordered restriction of length i which is identical to β in the first $i-1$ variables and then is similar to α on the i^{th} variable. The ordered restrictions α and β satisfy the following:

$$\forall i \in [t], c_i \neq d_i \implies F|_{\rho,\alpha_i} \equiv 0.$$

Furthermore, the converse also holds. That is, let β correspond to an ordered restriction of some 1-leaf in Γ', then the ordered restriction α (with the same domain and same order of querying) corresponds to a 0-node in Γ' only if the above condition holds.

Since this is an important point, we summarize the above discussion in the following definition and claim.

▶ **Definition 3.1.** *Let F be a Boolean function, ρ a restriction and $\alpha = (v_1 \mapsto c_1, \ldots, v_t \mapsto c_t), \beta = (v_1 \mapsto d_1, \ldots, v_t \mapsto d_t)$ be two ordered restrictions (on the same domain and order of querying). We say $\alpha \in \text{ASSOC}_0(F, \rho, \beta)$ iff*

$$\forall i \in [t], c_i \neq d_i \implies F|_{\rho,\alpha_i} \equiv 0$$

where α_i refers to the ordered restriction $(v_1 \mapsto d_1, v_2 \mapsto d_2, \ldots, v_{i-1} \mapsto d_{i-1}, v_i \mapsto c_i)$.

▷ Claim 3.2. Let Γ compute the formula F under the restriction ρ and Γ' be the 0-balanced version of Γ. Let w be a 1-leaf in Γ (and hence also Γ') and β be the corresponding ordered restriction. Then α is an ordered restriction corresponding to a 0-leaf w' with $\text{assoc}(w') = w$ iff $\alpha \in \text{ASSOC}_0(F, \rho, \beta)$.

1-balancing is defined similarly with the roles of 0 and 1 reversed.

3.2 CDT Definition

We are now ready to define the canonical decision tree (CDT). As indicated before, this definition is identical to [14, Definition 19].

▶ **Definition 3.3.** *Given a formula F on variable set V and associated restriction tree $\tilde{\rho}\colon T_F \to \{0,1,*\}^V$, we define the canonical decision tree, denoted by $\mathrm{CDT}(F,\tilde{\rho})$, inductively (on depth and the number of variables) as follows:*

1. *If F is a constant 0 or 1, then $\mathrm{CDT}(F,\tilde{\rho})$ is the unique tree with a single leaf node labelled by the appropriate constant.*
2. *If F is a literal x or $\neg x$, then*
   - *if x is set by $\tilde{\rho}(F)$ to a constant, then $\mathrm{CDT}(F,\tilde{\rho})$ is the unique tree with a single node labelled by the appropriate constant.*
   - *Otherwise if x is unset by $\tilde{\rho}(F)$, then $\mathrm{CDT}(F,\tilde{\rho})$ is the tree with 3 nodes where the root is labelled by x and the two children are labelled appropriately by 0 or 1.*
3. *If $F = F_1 \vee \cdots \vee F_m$, then*
   - *If $F_1|_{\tilde{\rho}(F)} \equiv F_2|_{\tilde{\rho}(F)} \equiv \cdots \equiv F_m|_{\tilde{\rho}(F)} \equiv 0$, then $\mathrm{CDT}(F,\tilde{\rho})$ is the unique tree with a single leaf node labelled 0.*
   - *Else, there is some $1 \leq \ell \leq m$ such that $F_1|_{\tilde{\rho}(F)} \equiv \cdots \equiv F_{\ell-1}|_{\tilde{\rho}(F)} \equiv 0$ and $F_\ell|_{\tilde{\rho}(F)} \not\equiv 0$.*
   - *If $F_\ell|_{\tilde{\rho}(F)} \equiv 1$, then $\mathrm{CDT}(F,\tilde{\rho})$ is the unique tree node with a single leaf node labelled 1.*
   - *If $F_\ell|_{\tilde{\rho}(F)} \not\equiv$ constant, then do the following steps to construct $\mathrm{CDT}(F,\tilde{\rho})$*
     a. *Let Γ be $\mathrm{CDT}(F_\ell, \tilde{\rho}|_{F_\ell})$ constructed inductively (since $\mathrm{depth}(F_\ell) < \mathrm{depth}(F)$).*
     b. *Apply the restriction $\tilde{\rho}(F)$ to Γ to get Γ' and remove all the sub-trees which are inconsistent with $\tilde{\rho}(F)$[3].*
     c. *0-balance Γ' to get Γ''.*
     d. *For each 0-leaf u of Γ'', replace u by $\mathrm{CDT}(F|_{\alpha_u}, \tilde{\rho})$ where α_u is the ordered restriction corresponding to u in Γ''.*

   The case when $F = F_1 \wedge \cdots \wedge F_m$ is a conjunction of sub-formulas is handled similarly (with the roles of 0 and 1 reversed).

Given any s-long bit-string $a = (a_1, a_2, \ldots, a_s)$, we can walk along the CDT using a as an "instruction set". In other words, we walk from the root to a node of the tree by using a to make choices at the degree-2 nodes and otherwise following the degree one-edges. If this walks ends at a node w (possibly leaf node) of the CDT, we denote the corresponding ordered restriction α_w by $\mathrm{CDT}^{(a)}(F,\tilde{\rho})$, else $\mathrm{CDT}^{(a)}(F,\tilde{\rho})$ is undefined. When this node is a leaf node, then it is labelled either 0 or 1. In this case, we further enhance this definition as follows.

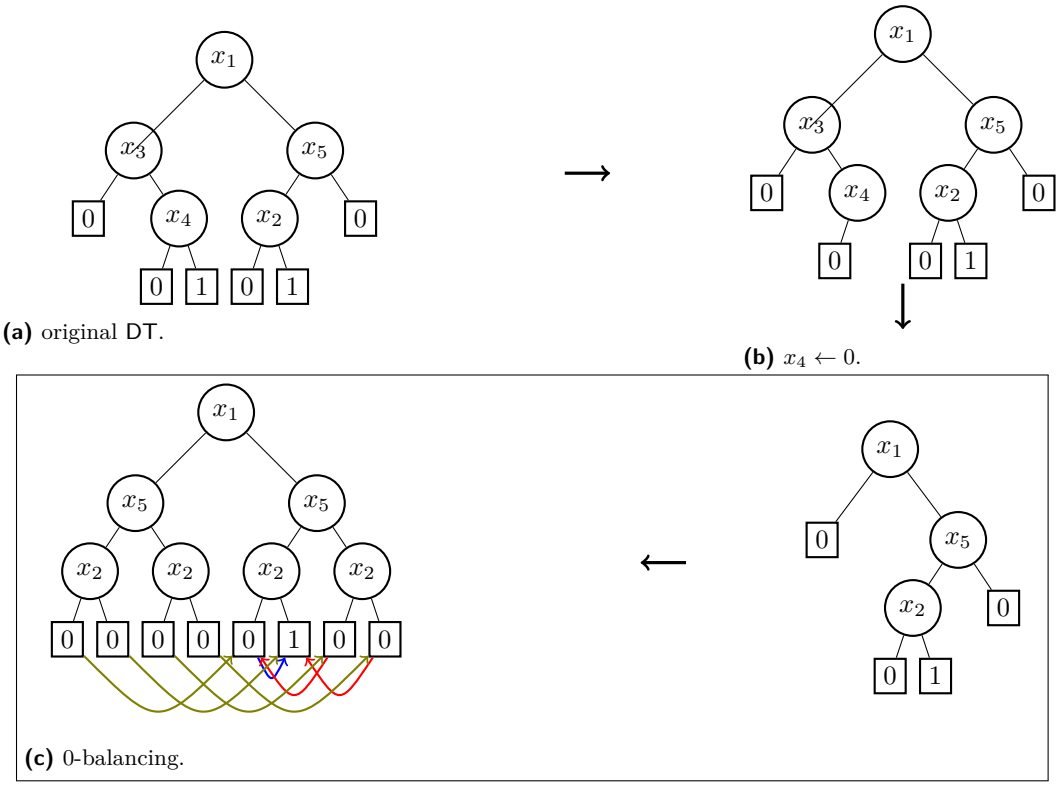
▶ **Definition 3.4.** *Let F be a formula and $\tilde{\rho}$ an associated restriction tree. For any bit-string $a = (a_1, \ldots, a_s)$ and $z \in \{0,1\}$, define*

$$\mathrm{CDT}_z^{(a)}(F,\tilde{\rho}) = \begin{cases} \alpha_w & \text{if walk according to instruction set "a" ends on leaf } w \text{ labelled } b, \\ \bot & \text{otherwise.} \end{cases}$$

3.3 Unpacking the CDT

Fix a formula F on n variables and an associated restriction tree $\tilde{\rho}\colon T_F \to \{0,1,*\}^n$. Let $a = (a_1, \ldots, a_s)$ be an s-bit-string with $s \geq 1$. Let us assume $F = F_1 \vee F_2 \vee \cdots \vee F_m$ is a disjunction. In this section, we try to understand when $\mathrm{CDT}_0^{(a)}(F,\tilde{\rho})$ exists.

[3] This step introduces degree 1 nodes in the decision tree.



(a) original DT.

(b) $x_4 \leftarrow 0$.

(c) 0-balancing.

Figure 3 Illustration of Items 3a–3c in CDT construction.

Suppose $\mathrm{CDT}_0^{(a)}(F, \tilde{\rho})$ exists and is the ordered restriction α. Then, the following must be true.

- There exists a unique $\ell \in [m]$ such that for all $\ell' < \ell$, we have $F_{\ell'}|_{\tilde{\rho}(F)} \equiv 0$ and $F_\ell|_{\tilde{\rho}(F)} \not\equiv$ constant.
- Let Γ be $\mathrm{CDT}(F_\ell, \tilde{\rho}|_{F_\ell})$. Let Γ' be the tree obtained by restricting Γ by $\tilde{\rho}(F)$ and Γ'' be the 0-balancing of Γ'. There must be some $1 \leq r \leq s$ such that, a walk to a leaf of Γ'' using instruction set $a_{\leq r}$ leads us to a leaf v' in Γ''. Let α' be the ordered restriction corresponding to leaf v' (which ought to be a prefix of α) .
- $\mathrm{CDT}_0^{(a_{>r})}(F|_{\alpha'}, \tilde{\rho})$ exists, and is α'' say. In that case, $\alpha = (\alpha', \alpha'')$.

Let us peer deeper into the balancing operation. Since Γ'' is the 0-balancing of Γ', we must have that $\mathrm{assoc}(v') = u$ for some 1-leaf u of Γ'' and this u is a 1-leaf of $\Gamma = \mathrm{CDT}(F_\ell, \tilde{\rho}|_{F_\ell})$ as well. Let β be the ordered restriction corresponding to u in Γ''. In fact, β must be consistent with $\tilde{\rho}(F)$ as this path in $\mathrm{CDT}(F_\ell, \tilde{\rho}|_{F_\ell}) = \Gamma$ survived in Γ' as well. Thus, there is some instruction set $b \in \{0,1\}^t$, for some $t \geq r$, such that $\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\rho}|_{F_\ell}) = \beta$.

Let us focus on the differences between the ordered restriction α' and β. We know there were t degree-2 nodes on the path to β in $\mathrm{CDT}(F_\ell, \tilde{\rho}_\ell)$, and there were r degree-2 nodes on the path to α' in Γ''. Thus, among the t degree-2 nodes on the path to β, we must have that $t - r$ of them belong to $\mathrm{dom}(\tilde{\rho}(F))$ (with β being consistent with $\tilde{\rho}(F)$) and the path to α' uses $a_{\leq r}$ as instructions for the other r nodes (instead of whatever route was taken by the path to β).

We summarize this discussion in the following lemma. We will be using this lemma for a *random* restriction tree $\tilde{\boldsymbol{\rho}}$ (chosen according to a suitable distribution). To distinguish the quantities that depend on this random variable from the rest, we use bold font to indicate all the quantities (including $\tilde{\boldsymbol{\rho}}$ itself) that are functions of $\tilde{\boldsymbol{\rho}}$.

▶ **Lemma 3.5** (Unpacking $\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}})$). *Let $F = F_1 \vee \cdots \vee F_m$ be a formula and $\tilde{\boldsymbol{\rho}} \colon T_F \to \{0,1,*\}^n$ an associated restriction tree. Let $s \geq 1$ and $a \in \{0,1\}^s$.*

Then $\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}})$ exists if and only if there exist

- *$\ell \in [m]$*
- *non-negative integer $r \in [s]$,*
- *non-negative integer $t \geq r$*
- *a bit-string $b \in \{0,1\}^t$ and*
- *$Q \in \binom{[t]}{r}$*

such that the following three conditions $\mathcal{A}, \mathcal{B}, \mathcal{C}$ are met.

$\mathcal{A}(\ell, t, b)$: **(i)** *$F_{\ell'}|_{\tilde{\boldsymbol{\rho}}(F)} \equiv 0$ for all $\ell' < \ell$,*

(ii) *$\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\boldsymbol{\rho}}|_{F_\ell})$ exists (and is $\boldsymbol{\beta}$ say).*

(iii) *$\boldsymbol{\beta}$ is consistent with $\tilde{\boldsymbol{\rho}}(F)$,*

$\mathcal{B}(\ell, t, b, r, Q, a_{\leq r})$: **(i)** *$Q$ identifies $\mathrm{stars}(\tilde{\boldsymbol{\rho}}(F))$ within $\mathrm{dom}(\boldsymbol{\beta}) \cap \mathrm{stars}(\tilde{\boldsymbol{\rho}}(F_\ell))$.*

(ii) *Let $\boldsymbol{\alpha'}$ is the ordered restriction obtained by modifying $\boldsymbol{\beta}$ by replacing the assignment of the r variables in $\mathrm{dom}(\boldsymbol{\beta}) \cap \mathrm{stars}(\tilde{\boldsymbol{\rho}}(F_\ell))$ identified by Q by $a_{\leq r}$. We denote this process as "$\boldsymbol{\alpha'} \xleftarrow[Q \leftarrow a_{\leq r}]{\mathrm{stars}\, \tilde{\boldsymbol{\rho}}(F_\ell)} \boldsymbol{\beta}$". Then $\boldsymbol{\alpha'} \in \mathrm{ASSOC}_0(F_\ell, \tilde{\boldsymbol{\rho}}(F), \boldsymbol{\beta})$*

$\mathcal{C}(\ell, t, b, r, Q, a)$: $\mathrm{CDT}_0^{(a_{>r})}(F|_{\boldsymbol{\alpha'}}, \tilde{\boldsymbol{\rho}})$ exists (is $\boldsymbol{\alpha''}$ say).

Furthermore, when $\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}})$ exists, we have $\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}) = (\boldsymbol{\alpha'}, \boldsymbol{\alpha''})$.

When clear from context, we will drop the arguments ℓ, t, b, r, Q, a from the properties $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$.

4 Downward closure property

Let ρ, ρ' be two restrictions on the variable set V and let $T \subseteq V$ any subset of the variables. Recall that we say that $\rho' \preccurlyeq_T \rho$ if (1) $\mathrm{stars}(\rho') \cap T \subseteq \mathrm{stars}(\rho) \cap T$ and (2) ρ_1 and ρ_2 are consistent. We say that a set $\mathcal{F}$ of restrictions is downward-closed with respect to the set of variables T if the following holds for any pair of restrictions ρ, ρ'

$$\rho \in \mathcal{F} \text{ and } \rho' \preccurlyeq_T \rho \implies \rho' \in \mathcal{F}.$$

We now extend this definition of downward-closed sets to restriction trees.

▶ **Definition 4.1** (downward-closed set of restriction trees). *Let F be a formula on the variable set V and $\tilde{\rho}, \tilde{\rho}' \colon T_F \to \{0,1,*\}^V$ be two associated restriction trees. Let $S \subseteq V$. We say $\tilde{\rho}' \preccurlyeq_S \tilde{\rho}$ iff for all $G \in T_F$, we have $\tilde{\rho}'(G) \preccurlyeq_S \tilde{\rho}(G)$.*

We call a set $\mathcal{T}$ of restriction trees downward-closed with respect to the variable set S *if the following holds for any pair of restriction trees*

$$\tilde{\rho} \in \mathcal{T} \text{ and } \tilde{\rho}' \preccurlyeq_S \tilde{\rho} \implies \tilde{\rho}' \in \mathcal{T}.$$

If $S = V$ (the full set of variables), then we drop the subscript S in the above definitions.

It is evident that if $\mathcal{T}$ and $\mathcal{T}'$ are two downward-closed set of restriction trees with respect to a variable set, so is their intersection. The key property that enables our proof of the main lemma is the following downward-closure property.

▶ **Lemma 4.2.** *Let $F = F_1 \vee F_2 \vee \cdots \vee F_m$ be a formula on variable set V and $\tilde{\rho}\colon T_F \to \{0,1,*\}^{|V|}$ be an associated restriction tree. Let $s \in \mathbb{Z}_{>0}$, $a \in \{0,1\}^s$ and α be an ordered restriction such that*

$$\mathrm{CDT}_0^{(a)}(F,\tilde{\rho}) = \alpha.$$

Suppose $\tilde{\rho}'\colon T_F \to \{0,1,*\}^{|V|}$ is another restriction tree satisfying

- *$\tilde{\rho}' \preccurlyeq \tilde{\rho}$ and*
- *$\tilde{\rho}'(G)|_{\mathrm{dom}(\alpha)} = \tilde{\rho}(G)|_{\mathrm{dom}(\alpha)}$ for all $G \in T_F$,*

then $\mathrm{CDT}_0^{(a)}(F,\tilde{\rho}') = \alpha$.

Similarly, when $F = F_1 \wedge F_2 \wedge \cdots \wedge F_m$, the same holds for "$\mathrm{CDT}_1^{(a)}(F,\tilde{\rho}) = \alpha$".

Note that the lemma implies that the set $\mathcal{T}_{F,a,\alpha} := \{\tilde{\rho}\colon \mathrm{CDT}_0^{(a)}(F,\tilde{\rho}) = \alpha\}$ is downward-closed with respect to the variable set $V \setminus \mathrm{dom}(\alpha)$.

Proof. We are given that $\tilde{\rho}'$ and $\tilde{\rho}'$ behave identically on $\mathrm{dom}(\alpha)$, and $\tilde{\rho}'$ only sets more variables (all of them outside of $\mathrm{dom}(\alpha)$) than $\tilde{\rho}$. The proof is by induction on the depth and number of variables in the formula.

Base case. The base case is when $F|_{\tilde{\rho}(F)}$ is a literal or a constant. The lemma is clearly true in this case as $\tilde{\rho}'$ only sets more variables than $\tilde{\rho}$ and does not change the variables in $\mathrm{dom}(\alpha)$.

Induction step. Let F be a formula of depth d on the variable set $[n]$. Assume the lemma is true for all formulae of either depth less than d or involving less than n variables.

By the Unpacking lemma (Lemma 3.5), we have that $\mathrm{CDT}_0^{(a)}(F,\tilde{\rho}) = \alpha$ if and only if there exist ℓ, r, t, b, Q and ordered restrictions α', α'', β such that the following are true.

- **(i)** $F_{\ell'}|_{\tilde{\rho}(F)} \equiv 0$ for all $\ell' < \ell$,
- **(ii)** $\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\rho}|_{F_\ell}) = \beta$,
- **(iii)** β is consistent with $\tilde{\rho}(F)$,
- **(iv)** Q identifies $\mathrm{stars}(\tilde{\rho}(F))$ within $\mathrm{dom}(\beta) \cap \mathrm{stars}(\tilde{\rho}(F_\ell))$.
- **(v)** $\alpha' \in \mathrm{ASSOC}_0(F_\ell, \tilde{\rho}(F), \beta)$ where $\alpha' \xleftarrow[Q \leftarrow a_{\le r}]{\mathrm{stars}\,\tilde{\rho}(F_\ell)} \beta$ (i.e., α' is the ordered restriction obtained by modifying β by replacing the assignment of the r variables in $\mathrm{dom}(\boldsymbol{\beta}) \cap \mathrm{stars}(\tilde{\boldsymbol{\rho}}(F_\ell))$ identified by Q by $a_{\le r}$). Note that $\alpha' \in \mathrm{ASSOC}_0(F_\ell, \tilde{\rho}(F), \beta)$ ensures that α' is a 0-path in the decision tree Γ'' where Γ'' is defined as follows:

$$\Gamma = \mathrm{CDT}(F_\ell, \tilde{\rho}_{F_\ell}) \overset{\text{Apply } \tilde{\rho}(F)}{\rightsquigarrow} \Gamma' \overset{\text{0-balance}}{\rightsquigarrow} \Gamma''.$$

- **(vi)** $\mathrm{CDT}_0^{(a_{>r})}(F|_{\alpha'}, \tilde{\rho}) = \alpha''$.
- **(vii)** $\alpha = (\alpha', \alpha'')$.

We will demonstrate that for the same ℓ, r, t, b, Q and ordered restrictions α', α'', β all the above conditions continue to hold good when $\tilde{\rho}$ is replaced by $\tilde{\rho}'$. This will prove that $\mathrm{CDT}_0^{(a)}(F,\tilde{\rho}') = \alpha$.

Item vii is trivially true as this is independent of $\tilde{\rho}$ or $\tilde{\rho}'$. The other conditions are met for the following reasons. We first observe that since $\alpha' \in \mathrm{ASSOC}_0(F_\ell, \tilde{\rho}(F), \beta)$, we have $\mathrm{dom}(\beta) = \mathrm{dom}(\alpha') \subseteq \mathrm{dom}(\alpha)$.

- Items i and iii continue to hold good when when more variables are set from $\tilde{\rho}$ to $\tilde{\rho}'$.

- Items ii and vi are true when $\tilde{\rho}$ is replaced by $\tilde{\rho}'$ due to the inductive assumption (since F_ℓ is a formula of smaller depth, $F|_{\alpha'}$ is a formula on fewer variables and $\tilde{\rho}'$ does not alter the variables in $\mathrm{dom}(\beta) = \mathrm{dom}(\alpha')$ or $\mathrm{dom}(\alpha'')$).
- Since the variables in $\mathrm{dom}(\beta) = \mathrm{dom}(\alpha') \subseteq \mathrm{dom}(\alpha)$ are unaltered by $\tilde{\rho}'$, we have

$$\mathrm{stars}(\tilde{\rho}(F)) \cap \mathrm{dom}(\beta) = \mathrm{stars}(\tilde{\rho}'(F)) \cap \mathrm{dom}(\beta),$$
$$\mathrm{stars}(\tilde{\rho}(F_\ell)) \cap \mathrm{dom}(\beta) = \mathrm{stars}(\tilde{\rho}'(F_\ell)) \cap \mathrm{dom}(\beta).$$

  Hence, if Q identifies the starred variables $\mathrm{stars}(\tilde{\rho}(F))$ within $\mathrm{dom}(\beta) \cap \mathrm{stars}(\tilde{\rho}(F_\ell))$, it also identifies $\mathrm{stars}(\tilde{\rho}'(F))$ within $\mathrm{dom}(\beta) \cap \mathrm{stars}(\tilde{\rho}'(F_\ell))$. Thus, Item iv holds.
- As for Item v, since $\mathrm{dom}(\beta) \cap \mathrm{stars}(\tilde{\rho}(F_\ell)) = \mathrm{dom}(\beta) \cap \mathrm{stars}(\tilde{\rho}'(F_\ell))$ and $\alpha' \xleftarrow[Q \leftarrow a_{\leq r}]{\mathrm{stars}\,\tilde{\rho}(F_\ell)} \beta$, we also have $\alpha' \xleftarrow[Q \leftarrow a_{\leq r}]{\mathrm{stars}\,\tilde{\rho}'(F_\ell)} \beta$. It is now easy to verify from the definition of ASSOC_0 (Definition 3.1), if $\alpha' \in \mathrm{ASSOC}_0(F_\ell, \tilde{\rho}(F), \beta)$, then we also have $\alpha' \in \mathrm{ASSOC}_0(F_\ell, \tilde{\rho}'(F), \beta)$ since we are only setting more variables. Thus Item v also holds.

Thus, we have proved the claim. ◀

All of the above works even when dealing with the representation of restrictions given by pairs (σ, S) (see Section 2.2). In this case, the notion of downward closure is the standard definition of downward closure of sets. Lemma 4.2 merely re-stated in this language is the following

▶ **Lemma 4.3.** *Let $F = F_1 \vee F_2 \vee \cdots \vee F_m$ be a formula on variable set V and $(\sigma, \tilde{S})$ be a representation of associated restriction tree $\tilde{\rho}\colon T_F \to \{0,1,*\}^{|V|}$ (i.e, $\rho = \rho_{(\sigma,\tilde{S})}$). Let $s \in \mathbb{Z}_{>0}$, $a \in \{0,1\}^s$ and α be an ordered restriction such that*

$$\mathrm{CDT}_0^{(a)}(F, \tilde{\rho}) = \alpha.$$

Suppose $(\sigma, \tilde{S}')$ is a representation of another restriction tree $\tilde{\rho}'\colon T_F \to \{0,1,*\}^{|V|}$ satisfying

- *$\tilde{S}'(G) \subseteq \tilde{S}(G)$ for all $G \in T_F$ and*
- *$\tilde{S}'(G) \cap \mathrm{dom}(\alpha) = \tilde{S}(G) \cap \mathrm{dom}(\alpha)$ for all $G \in T_F$,*

then $\mathrm{CDT}_0^{(a)}(F, \tilde{\rho}') = \alpha$.

Similarly, when $F = F_1 \wedge F_2 \wedge \cdots \wedge F_m$, the same holds for "$\mathrm{CDT}_1^{(a)}(F, \tilde{\rho}) = \alpha$".

We will complete this discussion by extending the definition of downward-close to this representation of restrictions.

▶ **Definition 4.4.** *Let T be any subset of the variable set V. For any pair of sets $S, S' \subseteq V$, we say that $S' \subseteq_T S$ if $S' \cap T \subseteq S \cap T$ and $S' \setminus T = S \setminus T$. Similarly, for any pair $\tilde{S}, \tilde{S}'\colon T_F \to 2^V$, we say that $\tilde{S}' \subseteq_T \tilde{S}$ if for $\forall G \in T_F$, $\tilde{S}'(G) \subseteq_T \tilde{S}(G)$.*

A set of restrictions $\mathcal{F} \subseteq \{0,1\}^V \times 2^V$ (given by their representations) is downward closed with respect to variable set T if the following holds for every pair of representations (σ, S) and (σ', S'):

$$(\sigma, S) \in \mathcal{F} \text{ and } S' \subseteq_T S \text{ and } \sigma|_{\overline{S}} \equiv \sigma'|_{\overline{S}} \implies (\sigma', S') \in \mathcal{F}.$$

Similarly, a set of restriction trees $\mathcal{T}$ (given by their representations) is downward closed with respect to the variable set T if the following holds for any pair of restriction trees $(\sigma, \tilde{S})$ and $(\sigma', \tilde{S}')$

$$(\sigma, \tilde{S}) \in \mathcal{T} \text{ and } \tilde{S}' \subseteq_T \tilde{S} \text{ and } \sigma|_{\overline{\tilde{S}(F)}} \equiv \sigma'|_{\overline{\tilde{S}(F)}} \implies (\sigma', \tilde{S}') \in \mathcal{T}.$$

Thus, Lemma 4.3 implies that the set $\mathcal{T}_{F,a,\alpha} := \{(\sigma, \tilde{S})\colon \mathrm{CDT}_0^{(a)}(F, \tilde{\rho}_{(\sigma,\tilde{S})}) = \alpha\}$ is downward-closed with respect to the variable set $V \setminus \mathrm{dom}(\alpha)$.

5 Bounds on criticality

In this section, we prove Theorem 1.1 (the criticality result for AC^0 formulae). To this end, we first define $\lambda(F)$, the bound on criticality that we eventually prove. We then define a sampling procedure to sample random restriction trees $\tilde{\boldsymbol{\rho}}$ for a given formula F such that the marginal distribution $\tilde{\boldsymbol{\rho}}(F)$ (i.e, the distribution of the restriction corresponding to the entire formula) is the standard p-random restriction. Finally, we state and prove the main inductive lemma (Lemma 5.6) that proves Theorem 1.1.

We begin by defining $\lambda(F)$ for any AC^0-formula.

▶ **Definition 5.1** (lambda). *For a positive integer $S \in \mathbb{Z}_{>0}$ and non-negative integer $d \in \mathbb{Z}_{\geq 0}$, define*

$$\lambda_{S,d} := 32^{d+1}\left(\frac{\log S}{d}+1\right)^d = 32^{d+1}\left(\frac{\log(2^d \cdot S)}{d}\right)^d.$$

Given an AC^0 formula F of depth $d+1$ and size S, define $\lambda(F) := \lambda_{S,d+1}$.

Note, that the above expression simplifies to 32 for depth-1 formulae (i.e., terms and clauses), where we have used the convention that $\frac{0}{0} = 1$.

▷ Claim 5.2. $8\lambda_{S,d} \leq \lambda_{S,d+1}$

Proof. $8\lambda_{S,d} \leq 8 \cdot 32^{d+1}\left(\frac{\log 2^{d+1}S}{d}\right)^d = \frac{\lambda_{S,d+1}}{4\cdot \log 2^{d+1}S}\frac{(d+1)^{d+1}}{d^d} \leq \frac{\lambda_{S,d+1}}{4}\frac{e(d+1)}{d+1+\log S} \leq \lambda_{S,d+1}.$ ◁

5.1 Sampling restriction trees

We begin by recalling the definition of the classical p-biased distribution and the p-random restriction $\mathcal{R}_p$ distribution over restrictions.

▶ **Definition 5.3** (p-biased distribution). *For $p \in [0,1]$ and variable set V, the p-biased distribution $\mu_p(V)$ is the distribution on the power set 2^V where a set $\boldsymbol{S} \in 2^V$ is sampled as follows:*

For each $v \in V$, independently set "$v \in \boldsymbol{S}$" with probability p.
We will express this succinctly as "$\boldsymbol{S} \leftarrow_p 2^V$".

▶ **Definition 5.4** (p-random restriction). *For $p \in [0,1]$ and a variable set V, $\mathcal{R}_p([n])$ is the distribution on representations of restrictions obtained by independently sampling a uniformly random string $\boldsymbol{\sigma} \leftarrow \{0,1\}^V$ and a set $\boldsymbol{S} \leftarrow_p 2^V$ and outputting the pair $(\boldsymbol{\sigma}, \boldsymbol{S})$. The corresponding random restriction $\boldsymbol{\rho}$ is given by $\boldsymbol{\rho} \leftarrow \rho_{(\boldsymbol{\sigma},\boldsymbol{S})}$.*

We now extend this definition to distribution over restriction trees. Given a formula F, we say that $\tilde{p}\colon T_F \to [0,1]$ is a valid set of probabilities if whenever G is a sub-formula of H, we have $\tilde{p}(G) \geq \tilde{p}(H)$.

▶ **Definition 5.5** ($\tilde{\mathcal{R}}_p$-distribution). *Let F be a formula on the variable set V and $\tilde{p}\colon T_F \to [0,1]$ be a valid set of probabilities. The distribution $\tilde{\mathcal{R}}_{\tilde{p}}(F)$ on representations of restriction trees is the the one obtained from the following sampling algorithm.*

1. *Choose a uniformly random string $\boldsymbol{\sigma} \leftarrow \{0,1\}^V$.*
2. *For each $G \in T_F$, choose independently a random $\boldsymbol{S}_G \leftarrow_{q_G} 2^V$ where*

$$q_G := \frac{\tilde{p}(G) - \tilde{p}(\mathit{parent}(G))}{1-\tilde{p}(\mathit{parent}(G))}.$$

(Here, we follow the convention that $\tilde{p}(\mathit{parent}(F)) = 0$).
Note $v \notin \boldsymbol{S}_G$ with probability $(1-\tilde{p}(G))/(1-\tilde{p}(\mathit{parent}(G)))$.

3. *For each $G \in T_F$, let $G_0 := G, G_1, \ldots, G_k := F$ be the sequence of formulae from G to the root F in the formula tree T_F. Set $\tilde{\boldsymbol{S}}(G) \leftarrow \boldsymbol{S}_{G_0} \cup \boldsymbol{S}_{G_1} \cup \cdots \cup \boldsymbol{S}_{G_k}$.*
4. *Output the pair $(\boldsymbol{\sigma}, \tilde{\boldsymbol{S}})$.*

The corresponding random restriction tree $\tilde{\boldsymbol{\rho}}$ is given by $\tilde{\boldsymbol{\rho}} \leftarrow \tilde{\rho}_{(\boldsymbol{\sigma}, \tilde{\boldsymbol{S}})}$.

For any $p \in [0, 1/\lambda(F)]$, let $\tilde{\mathcal{R}}_p(F)$ denote the distribution $\tilde{\mathcal{R}}_{\tilde{p}}(F)$ where $\tilde{p}$ is defined as follows $\tilde{p}(F) = p$ and for all $G \in T_F$ other than F, we have $\tilde{p}(G) = 1/8\lambda(G)$.[4]

It follows from the definition of $\tilde{\mathcal{R}}_{\tilde{p}}(F)$ that the marginal distribution $\tilde{\boldsymbol{\rho}}(G)$ on any sub-formula $G \in T_F$ is distributed exactly according to the distribution $\mathcal{R}_{\tilde{p}(G)}$.

5.2 Main Lemma

We are now ready to state the main lemma of the paper.

▶ **Lemma 5.6.** *Let $d \geq 0$ and $F = F_1 \vee F_2 \vee \cdots \vee F_m$ be an AC^0 formula of size S and depth $d+1$ on n variables. Let $\mathcal{T}$ be any set of downward-closed set of representations of restriction trees with respect to the variables of the formula F, then for all integers $s \geq 1$ and $a \in \{0,1\}^s$,*

$$\Pr_{(\boldsymbol{\sigma}, \tilde{\boldsymbol{S}}) \sim \tilde{\mathcal{R}}_p(F)} \left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma}, \tilde{\boldsymbol{S}})}) \text{ exists} \mid (\boldsymbol{\sigma}, \tilde{\boldsymbol{S}}) \in \mathcal{T} \right] \leq (p \cdot \lambda(F))^s.$$

The statement for conjunctions $F = F_1 \wedge F_2 \wedge \cdots \wedge F_m$ is identical with $\mathrm{CDT}_0^{(a)}$ replaced by $\mathrm{CDT}_1^{(a)}$.

Theorem 1.1 stated in the introduction clearly follows from the above lemma. The above lemma is stronger than what is needed for Theorem 1.1 as it proves the statement even when conditioned under any downward-closed set of restriction trees. This stronger statement is needed for the inductive proof to go through.

Proof. The proof is by induction on the depth d of the formula and the number of variables in the formula F.

Let us begin with the base case (depth-1 AC^0-formulae). The proof of this is similar to the proof of [6, Lemma 3.4]. The proof of the base case is written in a slightly more complicated fashion than it needs to be as it then serves as a warmup to one of the key claims (Claim 5.8) in the proof of the induction step (the base case).

Base case. The base case is when F is a depth-1 formula and we need to bound the probability by $(32p)^s$ since in this case $\lambda(F) = 32$. A depth-1 formula is a term or a clause. Without loss of generality let's assume that F is a clause of the form $x_1 \vee \cdots \vee x_m$, where the x_i's are distinct variables.

For a given $a \in \{0,1\}^s$, let $(\sigma, \tilde{S})$ be such that "$\mathrm{CDT}_0^{(a)}(F, \tilde{\rho}_{(\sigma,\tilde{S})})$ exists" and $(\sigma, \tilde{S}) \in \mathcal{T}$. Then there exists a unique subset of variables $Q^{(a)}_{(\sigma,\tilde{S})} \subset [m]$ of size s such that $\tilde{S}(F) \cap [m] = Q^{(a)}_{(\sigma,\tilde{S})}$ and for all variables $i \in [m] \setminus Q^{(a)}_{(\sigma,\tilde{S})}$, we have $\sigma(x_i) = 0$. For any $b \in \{0,1\}^s$, define $\sigma^{(b)}$ to be the global assignment that agrees with σ outside $Q^{(a)}_{(\sigma,\tilde{S})}$ and is equal to b within $Q^{(a)}_{(\sigma,\tilde{S})}$. Since $Q^{(a)}_{(\sigma,\tilde{S})} \subseteq \tilde{S}(F)$ and $\mathcal{T}$ is downward-closed, we have that these 2^s different

[4] For this to be well-defined, we need $\lambda(F) \geq 8\lambda(G)$ for any sub-formula G of F. This follows from Claim 5.2

representations $(\sigma^{(b)}, \tilde{S})$ are also in $\mathcal{T}$. Furthermore, since $\tilde{\rho}_{(\sigma,\tilde{S})} = \tilde{\rho}_{(\sigma^{(b)},\tilde{S})}$, we have that all these 2^s representations also satisfy "$\mathrm{CDT}_0^{(a)}(F, \tilde{\rho}_{(\sigma^{(b)},\tilde{S})})$ exists". We can hence conclude that

$$\begin{aligned}&\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\sim\tilde{\mathcal{R}}_p(F)}\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}) \text{ exists} \mid (\boldsymbol{\sigma},\tilde{\boldsymbol{S}}) \in \mathcal{T}\right]\\ &\leq 2^s \cdot \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\sim\tilde{\mathcal{R}}_p(F)}\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}) \text{ exists and } \forall i \in Q^{(a)}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}, \boldsymbol{\sigma}(x_i) = 1 \mid (\boldsymbol{\sigma},\tilde{\boldsymbol{S}}) \in \mathcal{T}\right].\end{aligned} \tag{1}$$

Let $\mathcal{E}_a = \left\{(\sigma,\tilde{S})\colon \mathrm{CDT}_0^{(a)}(F, \tilde{\rho}_{(\sigma,\tilde{S})}) \text{ exists and } \forall i \in Q^{(a)}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}, \boldsymbol{\sigma}(x_i) = 1\right\}$. We need to bound the quantity $\mu(\mathcal{E}_a \cap \mathcal{T})/\mu(\mathcal{T})$. For every $(\sigma,\tilde{S}) \in \mathcal{E}_a \cap \mathcal{T}$, define the set $N(\sigma,\tilde{S})$ as outlined below.

$$N(\sigma,\tilde{S}) := \left\{(\sigma,\tilde{S}')\colon \tilde{S}' \subseteq_{Q^{(a)}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}} \tilde{S}\right\}. \tag{2}$$

(Recall the definition of the notation "$\tilde{S}' \subseteq_T \tilde{S}$" from Definition 4.4).

It follows from the definition of $N(\sigma,\tilde{S})$ and the distribution $\tilde{\mathcal{R}}_p(F)$, that

$$\mu(\sigma,\tilde{S}) = p^s \cdot \mu(N(\sigma,\tilde{S})). \tag{3}$$

We now make the following observations about $N(\sigma,\tilde{S})$.

- Since $\mathcal{T}$ is downward closed and $(\sigma,\tilde{S}) \in \mathcal{T}$, we have $N(\sigma,\tilde{S}) \subseteq \mathcal{T}$.
- Exactly one element of $N(\sigma,\tilde{S})$, namely $(\sigma,\tilde{S})$, satisfies $\mathcal{E}_a$.
- For distinct $(\sigma,\tilde{S})$, the corresponding $N(\sigma,\tilde{S})$ are disjoint.

We can now bound $\Pr[\mathcal{E}_a \mid \mathcal{T}]$ using the above observations as follows:

$$\begin{aligned}\Pr[\mathcal{E}_a \mid \mathcal{T}] = \frac{\mu(\mathcal{E}_a \cap \mathcal{T})}{\mu(\mathcal{T})} &= \frac{\sum_{(\sigma,\tilde{S})\in\mathcal{E}_a\cap\mathcal{T}} \mu(\sigma,\tilde{S})}{\sum_{(\sigma,\tilde{S})\in\mathcal{E}_a\cap\mathcal{T}} \mu(N(\sigma,\tilde{S})) + \mu\left(\mathcal{T} \setminus \bigcup_{(\sigma,\tilde{S})\in\mathcal{E}_a\cap\mathcal{T}} N(\sigma,\tilde{S})\right)}\\ &\leq \frac{\sum_{(\sigma,\tilde{S})\in\mathcal{E}_a\cap\mathcal{T}} \mu(\sigma,\tilde{S})}{\sum_{(\sigma,\tilde{S})\in\mathcal{E}_a\cap\mathcal{T}} \mu(N(\sigma,\tilde{S}))} = p^s.\end{aligned}$$

Combining the above bound with (1), we thus have

$$\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\sim\tilde{\mathcal{R}}_p(F)}\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}) \text{ exists} \mid (\boldsymbol{\sigma},\tilde{\boldsymbol{S}}) \in \mathcal{T}\right] \leq (2p)^s,$$

concluding the base case of the induction.

Induction step. Let us assume without loss of generality that $F = F_1 \vee \cdots \vee F_m$ and the main lemma holds for all formulae of smaller depth (in particular the F_i's) and all formulae with smaller number of variables (in particular $F|_\beta$ for any non-trivial restriction β). By the Unpacking Lemma (Lemma 3.5) and a union bound we have that

$$\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}) \text{ exists} \mid (\boldsymbol{\sigma},\tilde{\boldsymbol{S}}) \in \mathcal{T}\right] \leq \sum_{r,\ell,t,Q,b} \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}[\mathcal{A} \cap \mathcal{B} \cap \mathcal{C} \mid \mathcal{T}], \tag{4}$$

where $\mathcal{A}, \mathcal{B}$ and $\mathcal{C}$ are as defined in Lemma 3.5.

For each fixing of ℓ, r, t, b, Q and a, we will bound the summand $\Pr[\mathcal{A} \cap \mathcal{B} \cap \mathcal{C} \mid \mathcal{T}]$ in the above expression. Consider a $(\sigma,\tilde{S}) \in \mathcal{A} \cap \mathcal{B} \cap \mathcal{C} \cap \mathcal{T}$. Let $\beta = \mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\rho}_{(\sigma,\tilde{S})|_{F_\ell}})$ which is guaranteed to exist as $(\sigma,\tilde{S}) \in \mathcal{A}$. By property $\mathcal{B}$, we have that there exist exactly r variables

in $\mathrm{dom}(\beta) \cap \mathrm{stars}(\tilde{\rho}_{(\sigma,\tilde{S})}|_{F_\ell}) = \mathrm{dom}(\beta) \cap \tilde{S}(F_\ell)$ which belong to $\mathrm{stars}(\tilde{\rho}_{(\sigma,\tilde{S})}) = \tilde{S}(F)$. Let us refer to this set of variables as $Q_{(\sigma,\tilde{S})}$. This part of the proof is similar to the base case. For any $b \in \{0,1\}^s$, define $\sigma^{(b)}$ to be the global assignment that agrees with σ outside $Q_{(\sigma,\tilde{S})}$ and is equal to b within $Q_{(\sigma,\tilde{S})}$. Since $Q_{(\sigma,\tilde{S})} \subseteq \tilde{S}(F)$ and $\mathcal{T}$ is downward-closed, we have that these 2^r different representations $(\sigma^{(b)}, \tilde{S})$ are also in $\mathcal{T}$. Furthermore, since $\tilde{\rho}_{(\sigma,\tilde{S})} = \tilde{\rho}_{(\sigma^{(b)},\tilde{S})}$, we have that all these 2^r representations also satisfy $\mathcal{A} \cap \mathcal{B} \cap \mathcal{C}$. We can hence conclude that

$$\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}[\mathcal{A} \cap \mathcal{B} \cap \mathcal{C} \mid \mathcal{T}] \leq 2^r \cdot \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}[\mathcal{A}' \cap \mathcal{B} \cap \mathcal{C} \mid \mathcal{T}], \tag{5}$$

where $\mathcal{A}'(\ell,t,b)$ is a modification of $\mathcal{A}$ (with respect to Item iii) as follows:

$\mathcal{A}'(\ell,t,b)$: **(i)** $F_{\ell'}|_{\tilde{\boldsymbol{\rho}}(F)} \equiv 0$ for all $\ell' < \ell$,

(ii) $\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\boldsymbol{\rho}}|_{F_\ell})$ exists (and is $\boldsymbol{\beta}$ say).

(iii) $\boldsymbol{\beta}$ is consistent with $\boldsymbol{\sigma}$,

We thus, have

$$\Pr\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}) \text{ exists} \mid (\boldsymbol{\sigma}, \tilde{\boldsymbol{S}}) \in \mathcal{T}\right] \leq \sum_{r,\ell,t,Q,b} 2^r \cdot \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}[\mathcal{A}' \cap \mathcal{B} \cap \mathcal{C} \mid \mathcal{T}]. \tag{6}$$

For each fixed choice r, ℓ, t, b, Q, the summand in the above expression can be factorized as

$$\Pr[\mathcal{A}' \mid \mathcal{T}] \cdot \Pr[\mathcal{B} \mid \mathcal{A}' \cap \mathcal{T}] \cdot \Pr[\mathcal{C} \mid \mathcal{A}' \cap \mathcal{B} \cap \mathcal{T}]. \tag{7}$$

The following three claims bound each of the terms in the above product.

▷ Claim 5.7. For a fixed a, ℓ, r, t, b and Q, we have

$$\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}\left[\mathrm{CDT}_0^{(a_{>r})}(F|_{\boldsymbol{\alpha}'}, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}) \text{ exists} \mid \mathcal{A}'(\ell,t,b) \cap \mathcal{B}(\ell,t,b,r,Q,a_{\leq r}) \cap \mathcal{T}\right] \leq (p \cdot \lambda(F))^{s-r}.$$

Proof. We first note that the formula being considered in the above expression, namely $F|_{\boldsymbol{\alpha}'}$, is itself random since the ordered restriction $\boldsymbol{\alpha}'$ is random. To deal with this, we prove the above bound for each fixing of $\boldsymbol{\alpha}'$. More precisely, we rewrite the above expression as follows (here we not only fix $\boldsymbol{\alpha}'$, but also $\boldsymbol{\beta}$).

$$\mathbb{E}_{\alpha',\beta}\left[\underbrace{\Pr_{\tilde{\boldsymbol{\rho}}}\left[\mathrm{CDT}_0^{(a_{>r})}(F|_{\alpha'}, \tilde{\boldsymbol{\rho}}) \text{ exists} \mid \mathcal{A}' \cap \mathcal{B} \cap \mathcal{T} \cap \mathcal{E}_{\alpha',\beta}\right]}\right],$$

where the expectation over α' and β is over the appropriate marginal distribution and $\mathcal{E}_{\alpha',\beta}$ is the set of representations of restriction trees $(\boldsymbol{\sigma}, \tilde{\boldsymbol{S}})$ that satisfy $\boldsymbol{\alpha}' = \alpha'$ and $\boldsymbol{\beta} = \beta$. We will prove that for any fixing of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, the indicated quantity in the above expression is at most $(p \cdot \lambda(F))^{s-r}$, which would imply the claim.

Consider any fixing (α', β) of $(\boldsymbol{\alpha}', \boldsymbol{\beta})$. We first observe that since α' is a non-trivial ordered restriction (which is true since $r \geq 1$), the variable set of the formula $F|_{\alpha'}$ is less than that of F and hence we can apply the inductive assumption provided the set $\mathcal{A}' \cap \mathcal{B} \cap \mathcal{T} \cap \mathcal{E}_{\alpha',\beta}$ is downward closed with respect to the variables of $F|_{\alpha'}$. Below, we verify that this is indeed the case.

$\mathcal{A}'(\ell,t,b) \cap \mathcal{E}_{\alpha',\beta}$: We will show that each of the 3 items of $\mathcal{A}' \cap \mathcal{E}_{\alpha',\beta}$ are downward-closed.

(i). $\mathcal{A}'$i and $\mathcal{A}'$iii are clearly downward-closed.

(ii). $\mathcal{A}'$ii $\cap\ \mathcal{E}_{\alpha',\beta}$ is the event that "$\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}|_{F_\ell}) = \beta$". This is downward-closed on the variable set $[n] \setminus \mathrm{dom}(\beta) = [n] \setminus \mathrm{dom}(\alpha')$ by Lemma 4.3.

$\mathcal{B}(\ell, t, b, r, Q.a_{\leq r}) \cap \mathcal{E}_{\alpha',\beta}$: $\mathcal{B}$i and $\mathcal{B}$ii: Both these conditions continue to hold good as long as the variables in $\mathrm{dom}(\beta) = \mathrm{dom}(\alpha')$ are unaltered. Since we care only about downward-closure on the variable set $[n] \setminus \mathrm{dom}(\alpha')$, we are fine (note this is not necessarily downward-closed on the entire set of variables).

Combined with the fact that $\mathcal{T}$ is downward-closed, we have $\mathcal{A}' \cap \mathcal{B} \cap \mathcal{T} \cap \mathcal{E}_{\alpha',\beta}$ is downward-closed on $[n] \setminus \mathrm{dom}(\alpha')$ and hence by the inductive assumption, we have the required bound. ◁

▷ Claim 5.8. For fixed a, ℓ, r, t, b and Q, we have $\Pr[\mathcal{B} \mid \mathcal{A}' \cap \mathcal{T}] \leq (8 \cdot p \cdot \lambda(F_\ell))^r$.

As indicated earlier, the proof of this claim is similar in spirit to the proof of the base case, which in turn is similar to the proof of [6, Lemma 3.4]. Things are however considerably more involved here and one has to do a careful conditioning argument to obtain the bound.

Proof. It suffices if for each fixed choice of a, ℓ, r, t, b and Q, we prove

$$\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}}) \sim \tilde{\mathcal{R}}_p(F)} \left[Q \text{ identifies } \tilde{\boldsymbol{S}}(F) \text{ within } \mathrm{dom}(\boldsymbol{\beta}) \cap \tilde{\boldsymbol{S}}(F_\ell) \mid \mathcal{A}'(\ell, t, b) \cap \mathcal{T} \right] \leq q^r,$$

where $q := (p/(1/8\lambda(F_\ell))) = (8 \cdot p \cdot \lambda(F_\ell))$.

Consider any $(\sigma, \tilde{S})$ that satisfies the 3 properties (1) Q identifies $\tilde{S}(F)$ within $\mathrm{dom}(\beta) \cap \tilde{S}(F_\ell)$, (2) $\mathcal{A}'(\ell, t, b)$ and (3) $\mathcal{T}$. As before let $Q_{(\sigma,\tilde{S})}$ be the set of r variables in $\tilde{S}(F_\ell) \cap \mathrm{dom}(\beta)$ which belong to $\tilde{S}(F)$. For every such $(\sigma, \tilde{S})$, we define the set $N(\sigma, \tilde{S})$ of representations of restrictions trees as follows.

$$N(\sigma, \tilde{S}) := \Big\{ (\sigma, \tilde{S}') \colon \tilde{S}'(G) \subseteq_{Q_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}} \tilde{S}(G) \text{ for every } G \in T_F \setminus T_{F_\ell} \text{ and } \tilde{S}'(H) = \tilde{S}(H) \text{ for every } H \in T_{F_\ell} \Big\}. \tag{8}$$

It follows from the definition of $N(\sigma, \tilde{S})$ and the distribution $\tilde{\mathcal{R}}_p(F)$, that

$$\mu(\sigma, \tilde{S}) = q^r \cdot \mu(N(\sigma, \tilde{S})). \tag{9}$$

We now make the following observations about $N(\sigma, \tilde{S})$.

- Exactly one element of $N(\sigma, \tilde{S})$, namely $(\sigma, \tilde{S})$, satisfies property (1).
- For distinct $(\sigma, \tilde{S})$, the corresponding $N(\sigma, \tilde{S})$ are disjoint.
- Since $\mathcal{T}$ is downward closed and $(\sigma, \tilde{S}) \in \mathcal{A}' \cap \mathcal{T}$, we have $N(\sigma, \tilde{S}) \subseteq \mathcal{A}' \cap \mathcal{T}$.

Putting these facts together, we have the following bound on the probability that we wish to bound.

$$\Pr_{(\boldsymbol{\sigma},\tilde{S})} \left[Q \text{ identifies } \tilde{\boldsymbol{S}}(F) \text{ within } \mathrm{dom}(\boldsymbol{\beta}) \cap \tilde{\boldsymbol{S}}(F_\ell) \mid \mathcal{A}'(\ell, t, b) \cap \mathcal{T} \right] \leq \frac{\sum_{(\sigma,\tilde{S})} \mu(\sigma, \tilde{S})}{\sum_{(\sigma,\tilde{S})} \mu(N(\sigma, \tilde{S}))} = q^r,$$

where the summation (in both the numerator and denominator) in the second step above is over all $(\sigma, \tilde{S})$ that satisfy all three properties. This completes the proof of the claim. ◁

▷ Claim 5.9. For fixed a, ℓ, t and b, we have $\eta(\ell, t, b) := \Pr[\mathcal{A}'(\ell, t, b) \mid \mathcal{T}] \leq \left(\frac{1}{8}\right)^t$.

Proof.

$$\begin{aligned}\eta(\ell,t,b) &= \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\sim\tilde{\mathcal{R}}_p(F)}\left[\mathcal{A}'(\ell,t,b) \mid \mathcal{T}\right] \leq \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\sim\tilde{\mathcal{R}}_p(F)}\left[\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})|_{F_\ell}}) \text{ exists} \mid \mathcal{T}\right] \\ &= \Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}}_{\boldsymbol{\ell}})\sim\tilde{\mathcal{R}}_{\tilde{p}(F_\ell)}(F_\ell)}\left[\mathrm{CDT}_1^{(b)}(F_\ell, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}}_{\boldsymbol{\ell}})}) \text{ exists} \mid \mathcal{T}\right] \\ &\leq (\tilde{p}(F_\ell)\cdot\lambda(F_\ell))^t = \left(\frac{1}{8\lambda(F_\ell)}\cdot\lambda(F_\ell)\right)^t = \left(\frac{1}{8}\right)^t.\end{aligned}$$

The last inequality follows from the induction assumption since F_ℓ has depth strictly smaller than that of F. ◁

Plugging the results of these claims back into the the expression in (7), we have

$$\eta(\ell,t,b)\cdot(16\cdot p\cdot\lambda(F_\ell))^r\cdot(p\cdot\lambda(F))^{s-r} \leq \left(\frac{1}{8}\right)^t\cdot(8\cdot p\cdot\lambda(F_\ell))^r\cdot(p\cdot\lambda(F))^{s-r}$$

We need to bound the sum of this expression when summed over all r,ℓ,t,b,Q as given by (6). However, even if we just over all possible ℓ the sum turns out to be prohibitively expensive. To keep this sum over ℓ (and also r,t,b,Q) under control, we further observe that the events $\mathcal{A}'(\ell,t,b)$ are mutually disjoint over disjoint ℓ,t,b. This lets us conclude the following claim.

▷ Claim 5.10. $\sum_{\ell,t,b}\eta(\ell,t,b) = \sum_{\ell,t,b}\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\sim\tilde{\mathcal{R}}_p(F)}\left[\mathcal{A}'(\ell,t,b) \mid \mathcal{T}\right] \leq 1.$

Proof. We observe that given a $(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})$, there is at most one ℓ such that $F_{\ell'}|_{\tilde{\boldsymbol{\rho}}(F)} \equiv 0$ and $F_\ell|_{\tilde{\boldsymbol{\rho}}(F)} \not\equiv 0$. Fix such an ℓ (if one exists). Given this ℓ, there is exactly one root-to-leaf path in $\mathrm{CDT}(F_\ell, \tilde{\boldsymbol{\rho}}|_{F_\ell})$ that is consistent with $\boldsymbol{\sigma}$. Let this be $\boldsymbol{\beta}$ (if one exists). Let $t := |\operatorname{dom}(\beta)\cap\operatorname{stars}(\tilde{\boldsymbol{\rho}}(F_\ell))|$ and $b\in\{0,1\}^t$ the assignment to the t degree-2 variables along the ordered restriction $\boldsymbol{\beta}$. Thus, $(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})$ uniquely determines (ℓ,t,b) such that $\mathcal{A}'(\ell,t,b)$ hold. Hence, $\eta(\ell,t,b)$ is a sub-distribution. ◁

We now have all the ingredients to bound the quantity of concern. The rest of the proof is a roller-coaster ride along the Jensen highway. We now bound the quantity in (6) as follows:

$$\begin{aligned}&\Pr_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}_{(\boldsymbol{\sigma},\tilde{\boldsymbol{S}})}) \text{ exists } \mid (\boldsymbol{\sigma},\tilde{\boldsymbol{S}})\in\mathcal{T}\right] \\ &\leq \sum_{r,\ell,t,b,Q} 2^r\cdot\eta(\ell,t,b)\cdot(8\cdot p\cdot\lambda(F_\ell))^r\cdot(p\cdot\lambda(F))^{s-r} \\ &\leq \sum_{r,\ell,t,b}\eta(\ell,t,b)\cdot(16\cdot p\cdot\lambda(F_\ell))^r\cdot(p\cdot\lambda(F))^{s-r}\cdot t^r \\ &= (p\cdot\lambda(F))^s\cdot\sum_{r,\ell}\left[\left(\frac{16\cdot\lambda(F_\ell)}{\lambda(F)}\right)^r\cdot\nu(\ell)\cdot\underbrace{\sum_{t,b}\frac{\eta(\ell,t,b)}{\nu(\ell)}\cdot t^r}\right] \qquad (10)\end{aligned}$$

where $\nu(\ell) := \sum_{t,b}\eta(\ell,t,b)$. We only sum over those ℓ that satisfy $\nu(\ell)>0$. Observe that $\sum_\ell\nu(\ell) = \sum_{\ell,t,b}\eta(\ell,t,b) \leq 1$. We first bound the quantity indicated (using underbraces) in the above expression using Jensen's inequality and Claim 5.9 as follows.

▷ Subclaim 5.11. $\sum_{t,b}\frac{\eta(\ell,t,b)}{\nu(\ell)}\cdot t^r \leq \left(\log\left(\frac{1}{\nu(\ell)}\right)\right)^r.$

Proof. Rewriting t^r as $(\log 2^t)^r$, the lefthand side can be written as $\sum_{t,b} \frac{\eta(\ell,t,b)}{\nu(\ell)} \cdot (\log 2^t)^r$. Since $\sum_{t,b} \frac{\eta(\ell,t,b)}{\nu(\ell)} = 1$, we can apply Jensen's inequality to the concave function $x \mapsto (\log x)^r$ to obtain

$$\begin{aligned}
&\sum_{t,b} \frac{\eta(\ell,t,b)}{\nu(\ell)} \cdot \left[\log 2^t\right]^r \\
&\leq \left[\log\left(\sum_{t,b} \frac{\eta(\ell,t,b)}{\nu(\ell)} \cdot 2^t\right)\right]^r \\
&\leq \left[\log\left(\sum_{t,b} \frac{1}{\nu(\ell)\cdot 4^t}\right)\right]^r && \text{[Since } \eta(\ell,t,b) \leq 8^{-t} \text{ from Claim 5.9]} \\
&\leq \left[\log\left(\frac{1}{\nu(\ell)} \sum_t \frac{1}{2^t}\right)\right]^r && \text{[Since there are at most } 2^t \text{ } b\text{'s]} \\
&\leq \left[\log \frac{1}{\nu(\ell)}\right]^r
\end{aligned}$$

◁

Substituting this bound back into the expression (10) above, we obtain

$$\Pr\left[\mathrm{CDT}_0^{(a)}(F, \tilde{\boldsymbol{\rho}}) \text{ exists} \mid \mathcal{T}\right] \leq (p \cdot \lambda(F))^s \cdot \sum_r \left[\left(\frac{16}{\lambda(F)}\right)^r \cdot \sum_\ell \nu(\ell) \cdot \underbrace{\left(\lambda(F_\ell) \cdot \log\left(\frac{1}{\nu(\ell)}\right)\right)^r}\right]$$

We now apply AM-GM inequality and the definition of $\lambda(F_\ell)$ to bound the indicated quantity.

▷ Subclaim 5.12. Let $S_\ell := \mathrm{size}(F_\ell)$. Then, $\lambda(F_\ell) \cdot \log\left(\frac{1}{\nu(\ell)}\right) \leq 32^d \left[\frac{\log\left(2^{d-1} \cdot S_\ell/\nu(\ell)\right)}{d}\right]^d$.

Proof. If $d = 1$, then $S_\ell = 1$ (recall the "size" defined in Definition 2.1), $\lambda(F_\ell) = 32$ (recall Definition 5.1). Thus, both sides of the above claim simplify to $32 \cdot \log(1/\nu(\ell))$.

For larger d,

$$\begin{aligned}
&\lambda(F_\ell) \cdot \log\left(\frac{1}{\nu(\ell)}\right) \\
&= 32^d \left(\frac{\log 2^{d-1} \cdot S_\ell}{d-1}\right)^{d-1} \cdot \log\left(\frac{1}{\nu(\ell)}\right) && \text{[Since } F_\ell \in \mathsf{AC}^0[S_\ell, d]\text{]} \\
&\leq 32^d \left[\frac{\log(2^{d-1} \cdot S_\ell) + \log\left(\frac{1}{\nu(\ell)}\right)}{d}\right]^d && \text{[Applying AM-GM inequality]} \\
&= 32^d \left[\frac{\log\left(2^{d-1} \cdot S_\ell/\nu(\ell)\right)}{d}\right]^d.
\end{aligned}$$

◁

Plugging this bound back into our expression, we have

$$\Pr\left[\mathrm{CDT}_0^{(a)}(F,\tilde{\boldsymbol{\rho}}) \text{ exists} \mid \mathcal{T}\right] \leq (p\cdot\lambda(F))^s \cdot \sum_r \left[\left(\frac{16\cdot 32^d}{\lambda(F)}\right)^r \cdot \underbrace{\sum_\ell \nu(\ell)\cdot\left(\frac{\log\left(\frac{2^{d-1}\cdot S_\ell}{\nu(\ell)}\right)}{d}\right)}^{dr}\right]$$

We bound the indicated quantity using yet another application of Jensen's inequality using the fact that $\sum_\ell \nu(\ell) \leq 1$ as follows.

▷ Subclaim 5.13. $\sum_\ell \nu(\ell)\cdot\left(\frac{\log\left(\frac{2^{d-1}\cdot S_\ell}{\nu(\ell)}\right)}{d}\right)^{dr} \leq \left[\frac{\log\left(2^d\cdot S\right)}{d}\right]^{dr}$.

Proof. Recall that $\sum_\ell \nu(\ell) \leq 1$. Consider the random variable Y defined as follows:

$$\boldsymbol{Y} \leftarrow \begin{cases} \frac{2^{d-1}\cdot S_\ell}{\nu(\ell)} & \text{with probability } \nu(\ell) \text{ for each } \ell \text{ such that } \nu(\ell)\neq 0,\\ 1 & \text{with probability } 1-\sum_\ell \nu(\ell).\end{cases}$$

and the concave function $x \xmapsto{f} \left(\frac{\log x}{d}\right)^{dr}$. Applying Jensen's inequality, we obtain

$$\begin{aligned}\mathbb{E}[f(\boldsymbol{Y})] \leq f(\mathbb{E}\,\boldsymbol{Y}) &= \left[\frac{\log\left(\left(\sum_\ell 2^{d-1}\cdot S_\ell\right)+\left(1-\sum_\ell \nu(\ell)\right)\right)}{d}\right]^{dr}\\ &\leq \left[\frac{\log\left(2^{d-1}\cdot S+1\right)}{d}\right]^{dr} && \left[\text{Since } S=\sum_\ell S_\ell\right]\\ &\leq \left[\frac{\log\left(2^d\cdot S\right)}{d}\right]^{dr} && [\text{Since } S\geq 1] \quad \triangleleft\end{aligned}$$

Plugging this bound back into our expression and recalling that $\lambda(F) = 32^{d+1}\cdot\left(\frac{\log 2^d\cdot S}{d}\right)^d$, we obtain

$$\Pr\left[\mathrm{CDT}_0^{(a)}(F,\tilde{\boldsymbol{\rho}}) \text{ exists} \mid \tilde{\boldsymbol{\rho}}\in\mathcal{T}\right] \leq (p\cdot\lambda(F))^s\cdot\sum_r \frac{1}{2^r} \leq (p\cdot\lambda(F))^s,$$

which completes the proof of our main lemma. ◀

6 Satisfiablity algorithms

In this section, we a give a randomized $\#SAT$ algorithm for general AC^0 formulae, matching the Impagliazzo-Matthews-Paturi result for AC^0 circuits. Rossman [14] had obtained a similar result for regular formulae. The proof below is a verbatim adaptation of Rossman's corresponding result [14, Theorem 30] for regular formulae to the general setting.

▶ **Theorem 6.1.** *There is a randomized, zero-error algorithm which, given an AC^0 formula F of depth $d+1$ and size S on n variables, outputs a decision tree for F of size $O\left(Sn\cdot 2^{(1-\varepsilon)n}\right)$ where $\varepsilon = 1/O\left(\left(\frac{1}{d}\log S\right)^d\right)$. This algorithm also solves the $\#SAT$ problem, that is, it counts the number of satisfying assignments for F.*

Proof. Given any depth d formula, and restriction tree $\tilde{\rho}$, the decision tree algorithm from Definition 3.3 computes $\mathrm{CDT}(F, \tilde{\rho})$ in time $O(n) \cdot \sum_{G \in T_F} \mathrm{size}(\mathrm{CDT}(G, \tilde{\rho}|_G))$. Given an AC^0 formula, consider the following tree of subsets $\tilde{D} \colon T_F \to [n]$ such that for each $G, H \in T_F$, such that G is a parent of H, $\tilde{D}(H) \subseteq \tilde{D}(G)$. For each such $\tilde{D}$, we get a decision tree for F as follows: We first construct a decision tree Γ by querying all the variables in $\tilde{D}(F)$ and labelling each leaf with the corresponding restriction on $\tilde{D}(F)$. For each such leaf σ (i.e, for each choice $\sigma \colon \tilde{D}(F) \to \{0,1\}$), we get a corresponding restriction tree $\tilde{\rho}_{\tilde{D},\sigma}$ in the natural manner. For each such σ, construct $\mathrm{CDT}(F, \tilde{\rho}_{\tilde{D},\sigma})$ and plug it in instead of the leaf corresponding to σ in the complete binary tree Γ. Clearly, this resultant tree $\Gamma_{\tilde{D}}$ is a decision tree for F.

We construct a (random) $\boldsymbol{\Gamma}_{\tilde{\boldsymbol{D}}}$ by sampling a $\tilde{\boldsymbol{D}}$ as follows: randomly choose a $\boldsymbol{\tau} \in_R [0,1]^n$ and set $\tilde{\boldsymbol{D}}(G) := \{i \colon \boldsymbol{\tau}_i \leq 1 - 1/8\lambda(G)\}$ for each $G \in T_F$. Therefore the expected running time of the algorithm which computes the decision tree for F is

$$O(n) \cdot \sum_{G \in T_F} \mathbb{E}_{\boldsymbol{\tau}} \left[\sum_{\sigma \colon \tilde{\boldsymbol{D}}(F) \to \{0,1\}} \mathrm{size}(\mathrm{CDT}(G, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\sigma}|_G)) \right]$$

while the expected size of the decision tree is $\mathbb{E}_{\boldsymbol{\tau}} \left[\sum_{\sigma \colon \tilde{\boldsymbol{D}}(F) \to \{0,1\}} \mathrm{size}(\mathrm{CDT}(F, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\sigma})) \right]$.

We bound these expression as follows. For each $G \in T_F$, size of the decision tree $\mathrm{CDT}(G, \tilde{\rho})$ is given by the expression,

$$\begin{aligned}
&\mathbb{E}_{\boldsymbol{\tau}} \left[\sum_{\sigma \colon \{0,1\}^{\tilde{\boldsymbol{D}}(F)}} \mathrm{size}(\mathrm{CDT}(F, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\sigma})) \right] = \mathbb{E}_{\boldsymbol{\tau}} \left[2^{|\tilde{\boldsymbol{D}}(F)|} \cdot \mathbb{E}_{\boldsymbol{\sigma}} \left[\mathrm{size}(\mathrm{CDT}(F, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\boldsymbol{\sigma}})) \right] \right] \\
&\leq 2^{n(1-1/16\lambda(F))} \cdot \underbrace{\mathbb{E}_{\boldsymbol{\tau},\boldsymbol{\sigma}} \left[\mathrm{size}(\mathrm{CDT}(F, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\boldsymbol{\sigma}})) \right]}_{} + 2^n \cdot e^{-(\frac{1}{2})^2 \cdot \frac{1}{2} \cdot \frac{n}{8\lambda(G)}}
\end{aligned}$$

where in the last expression, we have used the Chernoff Bound $\Pr[\sum X_i \leq (1-\delta)\mu] \leq e^{-\delta^2\mu/2}$ to bound the probability $\Pr[|[n] \setminus \tilde{\boldsymbol{D}}(F)| \leq (1 - 1/2)\mu]$ where $\mu = \mathbb{E}[|[n] \setminus \tilde{\boldsymbol{D}}(F)|] = n/8\lambda(F)$. We can now further simplify the expression indicated in the underbraces as follows:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\tau},\boldsymbol{\sigma}} \left[\mathrm{size}(\mathrm{CDT}(F, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\boldsymbol{\sigma}})) \right] &= \sum_{t \geq 0} \sum_{a \in \{0,1\}^t} \sum_{b \in \{0,1\}} \Pr_{\tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\boldsymbol{\sigma}}} \left[\mathrm{CDT}_b^{(a)}(F, \tilde{\boldsymbol{\rho}}_{\tilde{\boldsymbol{D}},\boldsymbol{\sigma}}) \ \text{ exists} \right] \\
&\leq 1 + \sum_{t=1}^{\infty} 2^t \left(\frac{1}{8} \right)^t = \frac{4}{3}.
\end{aligned}$$

We thus conclude that the expected size of the decision tree is at most $2^{n(1-\frac{1}{C\lambda})}$ for a suitably large constant C. ◀

References

1 Miklós Ajtai. Σ_1^1-formulae on finite structures. *Ann. Pure Appl. Logic*, 24(1):1–48, July 1983. doi:10.1016/0168-0072(83)90038-6.

2 Paul Beame. A switching lemma primer. Technical Report UW-CSE-95-07-01, University of Washington, 1994. URL: http://www.cs.washington.edu/homes/beame/papers/primer.ps.

3 Paul Beame, Russell Impagliazzo, and Srikanth Srinivasan. Approximating AC^0 by small height decision trees and a deterministic algorithm for $\#AC^0$-SAT. In Venkatesan Guruswami, editor, *Proc. 27th IEEE Conf. on Comput. Complexity*, pages 117–125, 2012. doi:10.1109/CCC.2012.40.

4 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984. (Preliminary version in 22*nd FOCS*, 1981). doi:10.1007/BF01744431.

5 Johan Håstad. Almost optimal lower bounds for small depth circuits. In Silvio Micali, editor, *Randomness and Computation*, volume 5 of *Advances in Computing Research*, pages 143–170. JAI Press, Greenwich, Connecticut, 1989. (Preliminary version in *18th STOC* 1986). URL: http://www.csc.kth.se/~johanh/largesmalldepth.pdf.

6 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM J. Comput.*, 43(5):1699–1708, 2014. doi:10.1137/120897432.

7 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In Yuval Rabani, editor, *Proc.* 23*rd Annual ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 961–972, 2012. doi:10.1137/1.9781611973099.77.

8 Nathan Linial, Yishay Mansour, and Noam Nisan. Constant depth circuits, Fourier transform, and learnability. *J. ACM*, 40(3):607–620, 1993. (Preliminary version in *30th FOCS*, 1989). doi:10.1145/174130.174138.

9 Alexander A. Razborov. Нжние оценки размера схем ограниченной глубины в полном базисе, содержащем функцию логического сложения (Russian) [Lower bounds on the size of bounded depth circuits over a complete basis with logical addition]. *Mathematicheskie Zametki*, 41(4):598–607, 1987. (English translation in *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987). doi:10.1007/BF01137685.

10 Alexander A. Razborov. Bounded arithmetic and lower bounds in Boolean complexity. In Peter Clote and Jeffrey B. Remmel, editors, *Feasible Mathematics II*, volume 13 of *Progress in Computer Science and Applied Logic*, pages 344–387. Birkhäuser, 1995. doi:10.1007/978-1-4612-2566-9_12.

11 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. (Preliminary version in *26th STOC*, 1991). doi:10.1006/jcss.1997.1494.

12 Benjamin Rossman. An entropy proof of the switching lemma and tight bounds on the decision-tree size of AC^0. (manuscript), 2017. URL: https://users.cs.duke.edu/~br148/logsize.pdf.

13 Benjamin Rossman. The average sensitivity of bounded-depth formulas. *Comput. Complexity*, 27(2):209–223, 2018. (Preliminary version in *56th FOCS*, 2015). doi:10.1007/s00037-017-0156-0.

14 Benjamin Rossman. Criticality of regular formulas. In Amir Shpilka, editor, *Proc.* 34*th Comput. Complexity Conf.*, volume 137 of *LIPIcs*, pages 1:1–1:28. Schloss Dagstuhl, 2019. doi:10.4230/LIPIcs.CCC.2019.1.

15 Michael Sipser. Borel sets and circuit complexity. In David S. Johnson, Ronald Fagin, Michael L. Fredman, David Harel, Richard M. Karp, Nancy A. Lynch, Christos H. Papadimitriou, Ronald L. Rivest, Walter L. Ruzzo, and Joel I. Seiferas, editors, *Proc.* 15*th ACM Symp. on Theory of Computing (STOC)*, pages 61–69, 1983. doi:10.1145/800061.808733.

16 Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In Alfred V. Aho, editor, *Proc.* 19*th ACM Symp. on Theory of Computing (STOC)*, pages 77–82, 1987. doi:10.1145/28395.28404.

17 Avishay Tal. Tight bounds on the Fourier Spectrum of AC^0. In Ryan O'Donnell, editor, *Proc.* 32*nd Comput. Complexity Conf.*, volume 79 of *LIPIcs*, pages 15:1–15:31. Schloss Dagstuhl, 2017. doi:10.4230/LIPIcs.CCC.2017.15.

18 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In Manuel Blum, John Hopcroft, Jeff Lagarias, Tom Leighton, Charles Rackoff, Larry Ruzzo, Larry Stockmeyer, Robert Tarjan, and Frances Yao, editors, *Proc.* 26*th IEEE Symp. on Foundations of Comp. Science (FOCS)*, pages 1–10, 1985. doi:10.1109/SFCS.1985.49.

Radical Sylvester-Gallai Theorem for Tuples of Quadratics

Abhibhav Garg
Cheriton School of Computer Science, University of Waterloo, Canada

Rafael Oliveira[1]
Cheriton School of Computer Science, University of Waterloo, Canada

Shir Peleg
Blavatnik School of Computer Science, Tel Aviv University, Israel

Akash Kumar Sengupta
Department of Mathematics, Columbia University, New York, NY, USA

Abstract

We prove a higher codimensional radical Sylvester-Gallai type theorem for quadratic polynomials, simultaneously generalizing [20, 36]. Hansen's theorem is a high-dimensional version of the classical Sylvester-Gallai theorem in which the incidence condition is given by high-dimensional flats instead of lines. We generalize Hansen's theorem to the setting of quadratic forms in a polynomial ring, where the incidence condition is given by radical membership in a high-codimensional ideal. Our main theorem is also a generalization of the quadratic Sylvester–Gallai Theorem of [36].

Our work is the first to prove a radical Sylvester–Gallai type theorem for arbitrary codimension $k \geq 2$, whereas previous works [36, 29, 30, 28] considered the case of codimension 2 ideals. Our techniques combine algebraic geometric and combinatorial arguments. A key ingredient is a structural result for ideals generated by a constant number of quadratics, showing that such ideals must be radical whenever the quadratic forms are far apart. Using the wide algebras defined in [28], combined with results about integral ring extensions and dimension theory, we develop new techniques for studying such ideals generated by quadratic forms. One advantage of our approach is that it does not need the finer classification theorems for codimension 2 complete intersection of quadratics proved in [36, 16].

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory; Theory of computation → Computational geometry

Keywords and phrases Sylvester-Gallai theorem, arrangements of hypersurfaces, algebraic complexity, polynomial identity testing, algebraic geometry, commutative algebra

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.20

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2023/074/

Acknowledgements The authors would like to thank Amir Shpilka for several useful discussions throughout the course of this work and for an anonymous reviewer for very helpful comments which helped improve the presentation, as well as to give an alternative proof of Lemma 64, which we give in Appendix A.

1 Introduction

Let $v_1, \ldots, v_m$ be a set of points in $\mathbb{R}^n$ with the property that the line joining any two points passes through a third point. The Sylvester–Gallai theorem states that $v_1, \ldots, v_m$ must all be collinear. This result was conjectured by Sylvester [39], and proved independently by Melchior [27] and Gallai [15].

[1] corresponding author

The inflection points of a cubic curve are a set of nine points in $\mathbb{C}^2$ such that the line joining any two of them passes through a third ([9]). However, these points are not collinear. In fact, Kelly [24] suggested that this was the motivation behind Sylvester's conjecture, to check if all inflection points can have real coordinates. In the same paper, Kelly observed that Hirzebruch's work on line arrangements [21] directly implies that every configuration of points in $\mathbb{C}^n$ satisfying the Sylvester–Gallai condition must be coplanar, and thereby answered a question of Serre [35]. This shows that the Sylvester–Gallai theorem crucially depends on the underlying field. If the underlying field is finite, then such configurations no longer have finite dimension. In light of these results, we fix our underlying field to $\mathbb{C}$ in this work, though our results hold for algebraically closed fields of characteristic zero.

A number of variations and generalisations of the Sylvester–Gallai theorem have been studied in combinatorial geometry such as a robust version [5], colored version [13], higher dimensional flats [20, 5] and many more. The main underlying theme in all such results is that the *local* linear relations between the points in a Sylvester–Gallai configuration must imply that such configurations can only happen in *low dimension*, which is a *global condition* on the configuration. Once one translates such geometric relations into algebraic terms, one sees that the study of Sylvester-Gallai configurations is a study about *cancellations* in algebraic geometry. In summary, Sylvester-Gallai type questions ask the following: given a set of algebraic geometric objects (e.g. vectors, linear forms or polynomials), whether "many" *local cancellations or syzygies* (such as the SG incidence conditions) imply *global constraints* on the configuration (such as being low-dimensional or dependence on a low number of variables).

Many results in algebraic and boolean complexity, as well as in cryptography, show that cancellations are very powerful in computation [32, 33, 41, 17, 42, 22, 26, 6]. Therefore, it is no surprise that proofs of Sylvester-Gallai theorems, which deal with limitations on the power of cancellations, have required sophisticated tools.

The variations alluded to above have applications in several areas of theoretical computer science, such as algebraic complexity (Polynomial Identity Testing and Reconstruction) and coding theory (Locally Correctable Codes). We now discuss some of these variations and their connections to TCS, and direct readers to [7] for more on classical Sylvester–Gallai problems.

Robust Sylvester-Gallai theorems. In this variation, one is given a constant $0 < \delta < 1$, and one requires the points $v_1, \ldots, v_m \in \mathbb{C}^n$ to satisfy the following condition: for every v_i, there are δm many points v_j such that the line joining v_i, v_j contains a third point in the set. The robust Sylvester-Gallai theorem states that such configurations lie on a constant dimensional subspace.

These configurations were first studied in [40], where the above theorem was proved for all values of δ that are close to 1. Subsequently, in [5], the authors proved the theorem for all values of δ, and showed that such configurations have dimension $O(1/\delta^2)$. In [12], this result was further improved, and the authors showed that such configurations have dimension $\mathcal{O}(1/\delta)$.

These configurations are useful in the study of locally correctable codes [5] and circuit reconstruction [37].

High dimensional Sylvester-Gallai theorems. Another variation of the Sylvester-Gallai theorem involves considering higher dimensional linear spaces instead of lines. For example, suppose now that for any v_i, v_j, v_k that are not collinear, we require the 2-dimensional

affine space spanned by v_i, v_j, v_k to contain a fourth point in the configuration. The higher dimensional Sylvester-Gallai theorem states that such configurations also lie in a constant dimensional affine subspace.

These configurations were first studied in [20], who proved the above theorem for affine spaces of all dimensions (the above is the case of dimension two). Further, in [4] the authors proved a robust version of the high dimensional Sylvester–Gallai theorem of [20].

These configurations have application in polynomial identity testing of depth three circuits ([23, 34]). The authors show that the linear forms in any depth three circuit computing the zero polynomial satisfy a version of this Sylvester-Gallai theorem, and therefore have low rank.

Higher degree generalisations and PIT. Motivated by the application of Syvester-Gallai theorems for depth three PIT, Gupta [19] introduced non-linear Sylvester-Gallai configurations and proposed Conjecture 1 below, generalizing the classical SG theorems to polynomials of higher degree, where the incidence condition is given by radical membership. [19] shows that a positive solution to Conjecture 1 yields deterministic poly-time PIT algorithms for depth four circuits with bounded top and bottom fan-in (circuits of the form $\Sigma^k\Pi\Sigma\Pi^d$).

Gupta divides nonzero $\Sigma^k\Pi\Sigma\Pi^d$ circuits into two classes, namely non-SG circuits and SG circuits. Informally, non-SG circuits are those where there is not much cancellation among the low degree polynomials computed at the bottom addition gate. These circuits form the easy case for their PIT algorithm, and the author gives an unconditional polynomial time algorithm to test if such circuits are nonzero. The analysis for non-SG circuits was recently simplified in [18].

The hard case for PIT is when there are non-trivial cancellations among the low-degree polynomials computed at the bottom addition gate. The author conjectures that such cancellations can only occur if this set of polynomials have constant transcendence degree. If this conjecture is true, then the Jacobian based method of [1] gives a poly-time deterministic PIT algorithm.

We now state the main conjecture of [19]:

▶ **Conjecture 1** (Conjecture 1, [19]). *Let $k, d, c \in \mathbb{N}^*$ be parameters, and let $\mathcal{F}_1, \ldots, \mathcal{F}_k$ be finite sets of irreducible polynomials of degree at most d satisfying*

- $\cap_i \mathcal{F}_i = \emptyset$,
- *for every $Q_1, \ldots, Q_{k-1}$, where each Q_j is from a distinct set $\mathcal{F}_{i_j}$, there are polynomials $P_1, \ldots, P_c$ in the remaining set such that $\prod P_i \in \operatorname{rad}(Q_1, \ldots, Q_{k-1})$.*

Then the transcendence degree of $\cup_i \mathcal{F}_i$ is a function of k, d, c, independent of the number of variables or the size of the sets $\mathcal{F}_i$.

In Conjecture 1, the division into k sets and the fact that the product of the forms in the remaining set are in the radical are both artefacts of the fact that the goal of the work was to solve $\Sigma^k\Pi\Sigma\Pi^d$ PIT. Since the conjecture above is a far-reaching non-linear generalization of Sylvester's conjecture, it is important to study simpler versions of this conjecture which are still wide open, just as was done in the linear case. With this in mind, towards the above conjecture, Gupta lists a series of conjectures regarding configurations that more closely resemble linear Sylvester-Gallai configurations, the first of which is the following.

▶ **Conjecture 2** (Conjecture 2, [19]). *Let $Q_1, \ldots, Q_m \in \mathbb{C}[x_1, \ldots, x_n]$ be irreducible, homogeneous, and of degree at most d such that for every pair Q_i, Q_j there is $k \neq i, j$ such that $Q_k \in \operatorname{rad}(Q_i, Q_j)$. Then the transcendence degree of $Q_1, \ldots, Q_m$ is $O_d(1)$ (where the constant depends on the degree d).*

Conjecture 2 is a beautiful mathematical generalization of the classical SG theorem as well as a stepping stone towards a full resolution of the PIT problem. So far Conjecture 2 is known for degrees 2 and 3 [36, 28] and it is open in general.

Since Conjecture 1 deals with radical ideals generated by $k-1$ polynomials (and hence of potentially higher codimension), it is important to generalize Conjecture 2 to a conjecture about radical ideals generated by k elements. Just as in the linear case (see [20]), some care must be taken when defining higher-codimensional Sylvester-Gallai configurations, and we address this formally in Section 3. Now, we present an informal version of the higher-codimensional SG conjecture, which will be the main focus of this work.[2]

▶ **Conjecture 3** (Higher-codimensional SG conjecture). *Let $\mathcal{F} \subset \mathbb{C}[x_1, \ldots, x_n]$ be a finite set of irreducible homogeneous forms of degree at most d. Suppose for every $F_1, \ldots, F_{k+1} \in \mathcal{F}$, either $F_{k+1} \in \operatorname{rad}(F_1, \ldots, F_k)$ or there exists $R \in \mathcal{F}$ such that $R \in \operatorname{rad}(F_1, \ldots, F_{k+1}) \setminus (\operatorname{rad}(F_1, \ldots, F_k) \cup (F_{k+1}))$. Then $\dim \operatorname{span}_{\mathbb{C}}\{\mathcal{F}\} = O_{d,k}(1)$ (where the constant depends on the degree d and the codimension parameter k).*

Note that the Sylvester-Gallai conditions in the above conjectures look different from the previous ones: we talk about membership in radical ideals as opposed to containment in affine spans. A discussion on why this is an appropriate generalisation of the linear Sylvester-Gallai condition can be found in [19].

Our main result, a proof of Conjecture 3 in the case where $d = 2$, is a step towards Conjecture 1 for the parameters $(k, d, c) = (k, 2, c)$ for any choice of $k, c \in \mathbb{N}$.

1.1 Main Result & Technical Contributions

In this subsection we informally state our main result, the higher codimensional analogue of the radical Sylvester–Gallai theorem. As is the case with the higher codimensional linear setting, the formal statement (Theorem 37) requires some additional definitions and is given in Section 3.[3]

▶ **Theorem 4** (Main theorem, informal). *Let $\mathcal{F} \subset \mathbb{C}[x_1, \cdots, x_n]$ be a finite set of irreducible forms of degree at most 2. Suppose for every $F_1, \ldots, F_{k+1} \in \mathcal{F}$, either $F_{k+1} \in \operatorname{rad}(F_1, \ldots, F_k)$ or there exists $R \in \mathcal{F}$ such that $R \in \operatorname{rad}(F_1, \ldots, F_{k+1}) \setminus (\operatorname{rad}(F_1, \ldots, F_k) \cup (F_{k+1}))$. Then $\dim \operatorname{span}_{\mathbb{C}}\{\mathcal{F}\} = O_k(1)$.*

▶ Remark 5. Note that our theorem, with $k = 1$, recovers the main theorem in [36].

Geometrically, the above statement says that the algebraic set defined by every set of $k+1$ forms in the configuration lies in the variety defined by another form. Since such algebraic sets have codimension at most $k+1$, we call our configurations higher codimension Sylvester-Gallai configurations.

In previous works [36, 29, 30, 31, 16, 28], which deal with (variants of) the case where $k = 1$, the approach used to prove a theorem of the above type required a structure theorem that would categorize ideals of the form (F_1, F_2) where each F_i is either a quadratic or a cubic form. These structure theorems used two main facts about ideals of the form (F_1, F_2):

1. they are complete intersections, and therefore Cohen-Macaulay (which implies unmixed).
2. they have small degree (four in the quadratic case and nine in the cubic case).

[2] The conjecture stated here is implied by our formal conjecture in Section 3.
[3] Theorem 37 in fact implies the result that we are stating in this page.

These two facts, along with properties of Hilbert-Samuel multiplicity, yield a list of special minimal primes and multiplicities such ideals can have, whenever they are not radical. Combined with existing literature and some new results on prime (and primary) ideals of codimension 2, the structure theorems are derived, and then used in the proof of their main theorem.

In our setting, neither of the above facts hold in general. The ideals we consider are generated by k quadratics, and therefore can have degree up to 2^k. Further, these ideals may no longer be complete intersections, and therefore can have embedded primes and even minimal primes of any codimension between 2 and k. This rules out the feasibility of using very fine-grained structure theorems as was done in previous works.

In a recent breakthrough, [2] proved that if one has quadratics $F_1, \dots, F_k$ which are "far enough apart," then the ideal $(F_1, \dots, F_k)$ is a complete intersection and prime (and hence radical). However, as discussed above, in our case this result alone is not enough for us to prove all we need: in many cases of interest, the forms in our configuration will not be far enough apart and the result from [2] will not apply.

To handle the remaining cases, we build on the techniques of [28] and prove a more general structural result on ideals generated by k quadratic forms. Our structural result (Lemma 64) states that given certain conditions on the quadratic forms $F_1, \dots, F_k$, even though they may not be far enough apart, one can still prove that the ideal $(F_1, \dots, F_k)$ is radical and has well-behaved minimal primes. The precise conditions of Lemma 64 are somewhat technical, and are developed in Section 6.1 with the definition of *integral sequences* of forms. An easier version of our structural lemma can be stated as follows:

▶ **Lemma 6** (Basic Lemma 64). *Let $F_1, \dots, F_k \in \mathbb{C}[x_1, \dots, x_n, y_1, \dots, y_k]$ be irreducible quadratic forms such that $F_i \in \mathbb{C}[x_1, \dots, x_n, y_i]$ is monic in y_i. Then, the ideal $I := (F_1, \dots, F_k)$ is radical and for any minimal prime $\mathfrak{p} \supset I$, we have $\mathfrak{p} \cap \mathbb{C}[x_1, \dots, x_n] = (0)$.*

Lemma 64, and the more basic version above, can be seen as general structural results, which say that either a given ideal is radical, or the generators are "related" (i.e. the "extra variables" $y_1, \dots, y_k$ must be related). This is a weaker structural result than the ones in the previous works, but holds in a more general setting, and is likely to generalise to higher degree configurations.

The proof of Lemma 64 involves tools from dimension theory, as well as the discriminant lemma, and the transfer principles from [28]. All of these concepts can be found in Section 4.

1.2 High level proof ideas

Our high level strategy is the that in order to bound $\dim \operatorname{span}_{\mathbb{C}} \{\mathcal{F}\}$, it is enough to prove that $\mathcal{F}$ is contained in a small graded algebra. To deal with the issues raised in the previous subsection, our strategy will be to prove that any such SG configuration $\mathcal{F}$ must be contained in a special ideal, which satisfies two properties:

1. the ideal is generated by a vector space $V := V_1 + V_2$ with $\dim V = O_k(1)$, where V_1 is a vector space of linear forms and V_2 is a vector space of quadratics
2. Any nonzero quadratic in V_2 is of very high rank (relative to $\dim V$).

With this result, we reduce the radical Sylvester-Gallai question to a linear, high-codimensional Sylvester-Gallai question, and apply the theorems from [5, 12, 11] to obtain that $\mathcal{F}$ must be contained in a small algebra. This is done in Section 7.3.

To prove that such special ideals exist, we proceed in two steps, each guided by a different conceptual principle. In the first step, we construct a small graded vector space W such that all forms in $\mathcal{F}$ are "close to" the algebra $\mathbb{C}[W]$. That is, there exists a constant B such that for each form $F \in \mathcal{F}$, there exist constantly many linear forms $y_1, \dots, y_r$, where $r \leq B$, such

that $F \in \mathbb{C}[W, y_1, \ldots, y_r]$. Note that both the linear forms y_i and the constant r depend on the form F, and the point here is to obtain a global upper bound on the values that r can take. We name such algebra $\mathbb{C}[W]$ *core algebra* (see Section 7.1).

In the second step, given $\mathcal{F}$ and a core algebra, we want to construct the special ideal V satisfying properties 1 and 2 above such that $\mathcal{F} \subset (V)$. To do this, we use Lemma 64 to show that for any sequence $F_1, \ldots, F_k$ such that $(F_1, \ldots, F_k)$ is not a radical ideal, it must be the case that the "extra variables" of the forms $F_1, \ldots, F_k$ must be (very) dependent. Thus we get a win-win type of result here: either the ideal $(F_1, \ldots, F_k)$ is radical (which gives us some linear dependencies amongst the forms of $\mathcal{F}$), or the linear forms coming from the extra variables must have very strong linear dependencies (and hence we can control their total dimension).

We now give an overview of each step.

Step 1 – constructing core algebras (Section 7.1). given a quadratic form Q and a vector space W, we say that Q is B-close to $\mathbb{C}[W]$ if there is a vector space Y of linear forms with $\dim Y \leq B$ such that $Q \in \mathbb{C}[W, Y]$.[4] That is, Q is a polynomial in few (linear) variables whenever we are allowed to have coefficients in $\mathbb{C}[W]$. We say that $\mathcal{F}$ is B-close to $\mathbb{C}[W]$ if every form in $\mathcal{F}$ is B-close to $\mathbb{C}[W]$. A *core algebra* is an algebra $\mathbb{C}[W]$ such that $\mathcal{F}$ is B-close to $\mathbb{C}[W]$ for some constant B.

The key inspiration for constructing such core algebras comes from the work [2], where the authors prove that if the quadratic forms $F_1, \ldots, F_{k+1}$ are "sufficiently far apart," then they form a *prime sequence* (which is a much stronger condition than complete intersection). Thus, either a given set of quadratic forms is a prime sequence, or one of the quadratics is "close" (that is, of low rank) to the vector space generated by the other quadratics.

One consequence of being a prime sequence is that the ideal $(F_1, \ldots, F_{k+1})$ will be a prime ideal (hence radical) and a complete intersection. If we have too many quadratic forms which are far apart, then the radical SG condition will imply that dependencies among the quadratics are linear dependencies, and therefore we can apply [5, 12] and construct our core algebra.

Here we get our first win-win: either many forms are far apart, in which case we will get linear dependencies (and thereby a vector space of low dimension) or we can construct a small vector space W such that $\mathcal{F}$ is close to $\mathbb{C}[W]$.

Since we want to control the quadratic forms of high rank (which we call strong forms), the proof of the construction of W requires an auxiliary SG configuration, dealing only with dependencies of high rank quadratics. We term these *strong SG configurations* (see Section 6.2 for details) and our proof is via a careful induction on the codimension of such configurations. Due to the fact that we are now dealing with both linear and quadratic forms, and our condition is a radical membership condition, the proof of this step is more involved and more delicate than the inductive approach used in [5, Section 5].

The technical reason why this step is more delicate than the induction on codimension done in [5, Section 5], is due to the fact that quotienting by a quadratic form will lead us to working with rings which are not necessarily polynomial rings, as well as the fact that we still have to handle non-linear radical dependencies and quadratic forms of low rank.

Step 2 – from core algebras to special ideals (Section 7.2). once we have constructed our core algebra $\mathbb{C}[W]$, we now have a global constant bound B such that all forms in $\mathcal{F}$ are B-close to $\mathbb{C}[W]$. In this setting, our structural lemma (Lemma 64) applies and we are able

[4] We extend this definition to linear forms by saying that any linear form is 1-close to any algebra.

to prove that either the quadratic forms are a linear Sylvester-Gallai configuration (which happens if many ideals $(F_1, \ldots, F_{k+1})$ are radical), or the extra variables of the quadratic forms must be (very) dependent. The proof of the aforementioned fact (in Section 7.2) is done by an iterative process to construct our special ideal. We couple Lemma 64 with two potential functions to prove termination of the iterative process providing the special ideal, in a similar way that [36, 16] use their potential functions.

Wide algebras

Both steps 1 and 2 use the notion of forms being close to an algebra. In Section 5, we make this notion clear, and establish what properties are needed from such algebras to make sure that we preserve the geometric properties of polynomial rings. Since we are dealing with quadratic forms, we need a slightly simpler version of the wide algebras introduced in [28].

1.3 Related work

As stated above, the main motivation for studying higher degree versions of the Sylvester-Gallai theorem comes from the relation established to depth four PIT in [19]. The $d = 2$ case of Conjecture 2 was proved in [36], which also kick started this line of work. Subsequently, in [29], the authors prove a product version of Conjecture 2 where the radical of the ideal generated by every pair of quadratics contains the product of all other quadratics. In [30], the authors strengthen this further, and prove Conjecture 1 in the case when $k = 3, d = 2, c = 4$. This also implies polynomial time PIT for $\Sigma^3\Pi\Sigma\Pi^2$ circuits. In [16] and [31] the authors independently proved a robust version of Conjecture 2 in the case when $d = 2$.

In [28], the authors prove Conjecture 2 in the case when $d = 3$. Our current work develops techniques building upon the intermediate results proved in [28]. In particular, the wide vector spaces we use are special cases of the wide vector spaces used in [28]. Further, our "structure theorems" are proved using the discriminant lemma from [28].

Progress on depth four PIT. There has been some recent progress on the PIT problem for depth four circuits with bounded top and bottom fan-in, the same model that is the focus on [19]. In [10], the authors give a quasipolynomial time PIT algorithm for such circuits. The authors use the Jacobian method of [1] to find a variable reduction map that preserves the algebraic independence of the inputs to the top addition gate. They are able to construct this map explicitly by first massaging the input circuits to change them to easier models, and then showing that the Jacobian can be computed by a read once oblivious arithmetic branching program (ROABP), for which hitting sets are known. Their methods are analytic in nature, and rely on the logarithmic derivative and its power series expansion.

In [25], the authors combine their lower bounds for bounded depth circuits with the methods of [8] to obtain subexponential time PIT algorithms for the same circuit families. Note that the methods of [8] cannot give a polynomial time PIT algorithm no matter how strong the lower bound assumptions are. Even getting a quasipolyomial time PIT from these methods for depth four circuits requires much stronger lower bounds than are currently known. However, these methods are more general, and work for all constant depth circuits.

The Sylvester–Gallai approach to PIT is the only one so far that can yield a deterministic poly–time algorithm. In both the works above, the methods used are quite distinct from the methods based on the Sylvester-Gallai theorem. In particular, they avoid dealing with cancellations, and therefore are unable to exploit the global structure that many local cancellations give rise to.

2 Preliminaries

In this section we establish notation and preliminary facts we will need for the rest of the paper. Let $S = \mathbb{C}[x_1, \ldots, x_N]$ denote the polynomial ring, graded by degree $S = \bigoplus_{i \geq 0} S_i$. Given a vector space $V \subset S$, we use V_i to denote the degree i piece, that is, $V_i = V \cap S_i$. We say that a vector space is graded if $V = \oplus V_i$.

We use *form* to refer to a homogeneous polynomial. Given two forms A, B we say that A, B are non-associate if $A \notin (B)$ and $B \notin (A)$. If A, B are of the same degree, this is equivalent to them being linearly independent.

2.1 Rank and linear spaces of quadratic forms

We now define a notion of the rank of quadratic forms, in accordance to [36].

▶ **Definition 7** (Rank of a quadratic form). *Let Q be a quadratic form. The rank of Q, denoted $\operatorname{rank} Q$, is the smallest $s \in \mathbb{N}$ such that we can write $Q = \sum_{i=1}^{s} a_i b_i$ with $a_i, b_i \in S_1$. If $\operatorname{rank} Q = s$, then a decomposition $Q = \sum_{i=1}^{s} a_i b_i$ with $a_i, b_i \in S_1$ is called a minimal representation of Q.*

▶ **Proposition 8.** *If $\phi : S_1 \to S_1$ is an invertible linear map and $\psi : S \to S$ is the map extending ϕ, then for any $Q \in S_2$ we have $\operatorname{rank} Q = \operatorname{rank} \psi(Q)$. If $U \subseteq S_1$ is a vector space of dimension k, and $\overline{Q}$ is the image of Q in $S/(U)$, then $\operatorname{rank} \overline{Q} \geq \operatorname{rank} Q - k$.*

Proof. Suppose $\operatorname{rank} Q = r$ and $Q = \sum_{i=1}^{r} a_i b_i$. We have $\psi(Q) = \sum_{i=1}^{r} \psi(a_i)\psi(b_i)$ therefore $\operatorname{rank} \psi(Q) \leq r$. If $\operatorname{rank} \psi(Q) = r'$ and $\psi(Q) = \sum_{i=1}^{r'} c_i d_i$ then $Q = \sum_{i=1}^{r'} \psi^{-1}(c_i)\psi^{-1}(d_i)$, which shows that $\operatorname{rank} Q = \operatorname{rank} \psi(Q)$.

Suppose $u_1, \ldots, u_k$ is a basis for U, and suppose $\overline{Q} = \sum_{i=1}^{r'} \overline{a_i}\,\overline{b_i}$. Then $Q = \sum_{i=1}^{r} a_i b_i + \sum_{j=1}^{k} u_i v_i$ for some $v_i \in S_1$. Therefore $\operatorname{rank} Q \leq \operatorname{rank} \overline{Q} + k$. ◀

▶ Remark 9. Let $Q = \sum_i a_{ii} x_i^2 + \sum_{i<j} 2a_{ij} x_i x_j$ be a quadratic form in S. Recall that there is an one-to-one correspondence between quadratic forms $Q \in S_2$ and symmetric bilinear forms. Let M be the symmetric matrix corresponding to the symmetric bilinear form of Q. Note that the (i, j)-the entry of M is given by a_{ij}. If M is of rank r, then after a suitable linear change of variables, we can write $Q = x_1^2 + \cdots + x_r^2$. Since the rank of a quadratic form is invariant under a linear change of variables(Proposition 8), we have $\operatorname{rank}(Q) = \lceil r/2 \rceil$, if M is of rank r.

In the next sections, we will need to use the following notion of a vector space of a quadratic form, which is a slight modification on the definition first given in [36]. The only modification that we make is that we preserve the quadratic form if its rank is high enough.

▶ **Definition 10** (Vector space of a quadratic form). *Let Q be a quadratic form of rank s, so that $Q = \sum_{i=1}^{s} a_i b_i$. Define the vector space $\operatorname{Lin}(Q) := \operatorname{span}_{\mathbb{C}} \{a_1, \ldots, a_s, b_1, \ldots, b_s\}$. Define $\mathbb{L}(Q)$ as:*

$$\mathbb{L}(Q) = \begin{cases} \operatorname{span}_{\mathbb{C}} \{Q\}, & \text{if } s \geq 5 \\ \operatorname{Lin}(Q), & \text{otherwise.} \end{cases}$$

We also extend the definition of Lin to linear forms in the natural way as follows.

▶ **Definition 11.** *For a linear form $\ell \in S_1$ define $\mathbb{L}(\ell) := \operatorname{span}_{\mathbb{C}} \{\ell\}$.*

Note that $\mathbb{L}(Q)$ is always a vector space of $\mathcal{O}(1)$ dimension (in fact, it is of dimension at most 10), while $\mathrm{Lin}(Q)$ can have non constant dimension. While a minimal representation $Q = \sum_{i=1}^{s} a_i b_i$ is not unique, the vector space $\mathrm{Lin}(Q)$ is unique and hence well-defined. The following lemma, which appears in [29, Fact 2.15] characterizes $\mathrm{Lin}(Q)$ as the smallest vector space of linear forms defining the algebras that contain Q.

▶ **Lemma 12.** *If* $Q = \sum_{i=1}^{r} x_i y_i$ *with* $x_i, y_i \in S_1$ *then* $\mathrm{Lin}(Q) \subseteq \mathrm{span}_{\mathbb{C}}\{x_i, y_j | i, j \in [r]\}$.

▶ Remark 13. The space $\mathrm{Lin}(Q)$ can also be defined as the space of first order partial derivatives of Q (see Lemma 16). However, we decided to not state this definition in this manner as this definition does not generalize well to forms of higher degree, as it is done in the works [2, 28].

We now state some useful results related to the rank and linear spaces of quadratics, some of which appear in [29, 16].

▶ **Lemma 14.** *Suppose* $Q \in S_2$ *is such that* $\mathrm{rank}\, Q = r$. *Then* $\dim \mathrm{Lin}(Q) = 2r$ *or* $\dim \mathrm{Lin}(Q) = 2r - 1$. *In the second case, we can write* $Q = a_r^2 + \sum_{i=1}^{r-1} a_i b_i$.

Proof. Suppose $v_1, \ldots, v_d$ is a basis for $\mathrm{Lin}(Q)$ for some $d \leq 2r$. We then have $Q \in \mathbb{C}[v_1, \ldots, v_d]$. By Remark 9, we can write $Q = \sum_{i=1}^{d'} u_i^2$ for some $d' \leq d$, where each $u_i \in \mathrm{span}_{\mathbb{C}}\{v_1, \ldots, v_d\}$. By Lemma 12 we have $\mathrm{Lin}(Q) \subseteq \mathrm{span}_{\mathbb{C}}\{u_1, \ldots, u_{d'}\}$ whence $d' = d$. If d is even then we get $d/2 \geq r$. Since we also have $d \leq 2r$ we get $d = 2r$. If d is odd, we must have $(d-1)/2 + 1 \geq r$. Since we also have $d \leq 2r$ we get $d = 2r - 1$. In this case, $u_d^2 + \sum_{j=1}^{d/2-1}(u_{2j-1} + u_{2j})(u_{2j-1} - u_{2j})$ is a minimal representation of Q, proving the last statement. ◀

▶ Remark 15. By the above lemma, given any $Q \in S_2$ such that $\mathrm{rank}\, Q = r$ we can write $Q = \sum_{i=1}^{r} a_i b_i$ such that $a_1, \ldots, a_r, b_1, \ldots, b_{r-1}$ are linearly independent, and either $b_r = a_r$ or b_r is independent of $a_1, \ldots, a_r, b_1, \ldots, b_{r-1}$.

▶ **Lemma 16.** *Let* $Q \in S = \mathbb{C}[x_1, \ldots, x_N]$ *be a quadratic form. Then* $\mathrm{Lin}(Q) = \mathrm{span}_{\mathbb{C}}\left\{\frac{\partial Q}{\partial x_1}, \cdots, \frac{\partial Q}{\partial x_N}\right\}$ *is the space of all first order partial derivatives of* Q.

Proof. Suppose $\mathrm{rank}\, Q = r$ and $\sum_{i=1}^{r} a_i b_i$ be a decomposition of Q as in Remark 15. Then note that $\frac{\partial Q}{\partial a_i} = b_i$ and $\frac{\partial Q}{\partial b_i} = a_i$ for all $i \leq r-1$. If $b_r = a_r$, then $\frac{\partial Q}{\partial a_r} = 2a_r$, and otherwise we have $\frac{\partial Q}{\partial a_r} = b_r$ and $\frac{\partial Q}{\partial b_r} = a_r$. Therefore $\mathrm{Lin}(Q) \subset \mathrm{span}_{\mathbb{C}}\left\{\frac{\partial Q}{\partial x_1}, \cdots, \frac{\partial Q}{\partial x_N}\right\}$. Since $Q = \sum_{i=1}^{r} a_i b_i$, we have $\frac{\partial Q}{\partial x_j} \in \mathrm{Lin}(Q)$ for all $j \in [N]$. ◀

The following lemma from [29] shows that adding a product of new variables increases the rank of a quadratic. In Lemma 18, we extend this to sums of quadratics in distinct variables.

▶ **Lemma 17** ([29, Claim 2.7])**.** *Suppose* $Q \in \mathbb{C}[x_1, \ldots, x_m]$ *is a polynomial of rank* r. *If* y, z *are new variables then* $\mathrm{rank}(Q + yz) = r + 1$. *In particular,* $\mathrm{Lin}(Q + yz) = \mathrm{Lin}(Q) + \mathrm{span}_{\mathbb{C}}\{y, z\}$.

▶ **Lemma 18.** *Suppose* $P \in \mathbb{C}[x_1, \ldots, x_m]$ *and* $Q \in \mathbb{C}[y_1, \ldots, y_n]$ *are two quadratics in distinct variables. Then* $\mathrm{Lin}(P + Q) = \mathrm{Lin}(P) + \mathrm{Lin}(Q)$.

Proof. Note that we have $\frac{\partial(P+Q)}{\partial x_i} = \frac{\partial P}{\partial x_i}$ and $\frac{\partial(P+Q)}{\partial y_j} = \frac{\partial Q}{\partial y_j}$ for all $i \in [m]$ and $j \in [n]$. Therefore, by Lemma 16, we have that $\mathrm{Lin}(P + Q) = \mathrm{Lin}(P) + \mathrm{Lin}(Q)$. ◀

▶ **Lemma 19.** *Let $W \subseteq S_1$ be a vector space. Suppose $Q \in S_2$ is such that* $\operatorname{rank} Q = r$ *in S and* $\operatorname{rank} \overline{Q} = r' < r$ *where $\overline{Q}$ is the image of Q in $S/(W)$. Then $W \cap \operatorname{Lin}(Q) \neq \{0\}$. In particular if $Q \in (W)$ then $W \cap \operatorname{Lin}(Q) \neq 0$.*

Proof. Suppose $Q = \sum_{i=1}^{r} a_i b_i$ is the minimal representation guaranteed by Remark 15. Assume towards a contradiction that $\operatorname{Lin}(Q) \cap W = \{0\}$. Since $a_1, \ldots, a_r, b_1, \ldots, b_{r-1}$ are independent in S, and either $b_r = a_r$ or b_r is independent of $a_1, \ldots, a_r, b_1, \ldots, b_{r-1}$, by assumption the same holds in $S/(W)$. We can now repeatedly apply Lemma 17 to deduce that $\operatorname{rank}(\overline{a_r}\overline{b_r} + \sum_{i=1}^{r-1} \overline{a_i}\overline{b_i}) = r$, contradicting assumption. ◀

2.2 General Projections

We now recall the definition and properties of projection maps from [36, 29, 28].

▶ **Definition 20** (Projection maps)**.** *Let $S = \mathbb{C}[x_1, \cdots, x_n]$ be a polynomial ring. Let $W \subset S_1$ be a subspace of linear forms and $y_1, \cdots, y_t$ be a basis of W. Let $y_1, \cdots, y_n$ be a basis of S_1 that extends the basis $y_1, \cdots, y_t$ of W. Let z be a formal variable not in $\{y_1, \cdots, y_n\}$. For $\alpha = (\alpha_1, \cdots, \alpha_t) \in \mathbb{C}^t$, we define the projection map $\varphi_{\alpha,W}$ as the $\mathbb{C}$-algebra homomorphism $\varphi_{\alpha,W} : S \to \mathbb{C}[z, y_{t+1}, \cdots, y_n] = S[z]/(W)$ defined by*

$$y_i \mapsto \begin{cases} \alpha_i z, & \text{if } 0 \leq i \leq t \\ y_i, & \text{otherwise} \end{cases}$$

For simplicity we will often drop the subscripts W or α, and write φ_α or φ for a projection map when there is no ambiguity about the vector space W or the vector α.

General projections. Fix a vector space $W \subset S_1$ as in Definition 20. We will say that a property holds for a *general projection* φ_α, if there exists a non-empty open subset $U \subset \mathbb{C}^t$ such that the property holds for all φ_α with $\alpha \in U$. Here $U \subset \mathbb{C}^t$ is open with respect to the Zariski topology, hence U is the complement of the zero set of finitely many polynomial functions on $\mathbb{C}^t$. The general choice of the element α defining a general projection φ_α allows us to say that such projection maps will avoid any finite set of polynomial constraints. As shown in [36, 29], general projection maps preserve several important properties of polynomials.

▶ **Proposition 21** ([28, Proposition 2.6])**.** *Let $F \in S$ be a polynomial and $W \subset S_1$ be a vector space of linear forms.*

(a) *If $F \notin \mathbb{C}[W]$, then $\varphi(F) \notin \mathbb{C}[z]$ for a general projection $\varphi : S \to S[z]/(W)$.*
(b) *If $F \neq 0$, then $\varphi(F) \neq 0$ for a general projection.*
(c) *Suppose F is a form which does not have any multiple factors and $F \in (W)$. If $\varphi(F) = z^k G$ where $G \notin (z)$, then G does not have any mulitple factors.*

The next proposition is from [29, Claim 2.23].

▶ **Proposition 22.** *Let $F, G \in S$ be two polynomials which have no common factor and $W \subset S_1$ a subspace of linear forms. For a general projection $\varphi : S \to S[z]/(W)$, we have* $\gcd(\varphi(F), \varphi(G)) \in \mathbb{C}[z]$. *In particular, if F, G are homogeneous then* $\gcd(\varphi(F), \varphi(G)) = z^k$ *for some $k \in \mathbb{N}$.*

The following result shows that general projections preserve linear independence for polynomials outside the algebra generated by W .

▶ **Corollary 23** ([28, Corollary 2.8])**.** *Let $F, G \in S$ be linearly independent irreducible forms and $W \subset S_1$ be a vector space of linear forms. If $F, G \notin \mathbb{C}[W]$ then $\varphi(F), \varphi(G)$ are linearly independent, for a general projection $\varphi : S \to S[z]/(W)$.*

The next proposition follows from [29, Claim 2.26].

▶ **Proposition 24.** *Let $W \subset S_1$ be a vector space of linear forms. Let $\mathcal{F} \subset S_2$ be a finite set of quadratic forms. Suppose there is an integer $D > 0$ such that $\dim \operatorname{span}_{\mathbb{C}} \left\{\bigcup_{F\in\mathcal{F}} \mathbb{L}\left(\varphi(F)\right)\right\} \leq D$ for a general projection $\varphi : S \to S[z]/(W)$. Then $\dim \operatorname{span}_{\mathbb{C}} \left\{\bigcup_{F\in\mathcal{F}} \mathbb{L}\left(F\right)\right\} \leq (D+1)\cdot \dim W$.*

The proposition above can be sharpened if we have extra information about the linear forms in $\mathcal{F}$. We state this sharpening in the next proposition

▶ **Proposition 25.** *Let $W \subset S_1$ be a vector space of linear forms and $\mathcal{F} \subset S_2$ be a finite set of quadratic forms such that $\mathcal{F} \cap (W)$ and $s(F) < s$ for each $F \in \mathcal{F}$. Suppose there is an integer $D > 0$ such that $\dim \operatorname{span}_{\mathbb{C}} \left\{\bigcup_{F\in\mathcal{F}} \mathbb{L}\left(\varphi(F)\right)\right\} \leq D$ for a general projection $\varphi : S \to S[z]/(W)$. Then we have $\dim \operatorname{span}_{\mathbb{C}} \left\{\bigcup_{F\in\mathcal{F}} \mathbb{L}\left(F\right)\right\} \leq (D+1)\cdot s$.*

3 Sylvester–Gallai configurations

We now formally define the Sylvester-Gallai configurations that we deal with in this work. Before we do this, we state the current known bounds on dimensions of linear Sylvester–Gallai configurations, these will be useful in our proofs.

3.1 Linear Sylvester–Gallai configurations

For this subsection, we let $\mathcal{L}$ be a finite set of pairwise non-associate linear forms and $\delta \in (0, 1]$ be a constant. We begin by defining ordinary and elementary spaces, as was done in [20, 5].

▶ **Definition 26** (Ordinary spaces). *Let $\ell_1, \ldots, \ell_k \in \mathcal{L}$, and let $V = \operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k\}$. The space V is called ordinary with respect to $\mathcal{L}$ if there are $\ell'_1, \ldots, \ell'_{k-1} \in S_1$, and $\ell \in \mathcal{L}$ such that $V \cap \mathcal{L} \subseteq \operatorname{span}_{\mathbb{C}} \{\ell'_1, \ldots, \ell'_{k-1}\} \cup \{\ell\}$.*

▶ **Definition 27** (Elementary spaces). *Let $\ell_1, \ldots, \ell_k \in \mathcal{L}$, and let $V = \operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k\}$. The space V is called elementary with respect to $\mathcal{L}$ if $V \cap \mathcal{L} = \{\ell_1, \ldots, \ell_k\}$.*

▶ **Definition 28.** *The set $\mathcal{L}$ is a $\delta - SG^*_k$ configuration if for every linearly independent $\ell_1, \ldots, \ell_k \in \mathcal{L}$, there are $\delta \cdot |\mathcal{L}|$ forms ℓ in $\mathcal{L}$ such that either*

1. *$\ell \in \operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k\}$,*
2. *or the linear space $\operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k, \ell\}$ contains a form in $\mathcal{L} \setminus (\operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k\} \cup \{\ell\})$.*

▶ **Definition 29.** *The set $\mathcal{L}$ is a $\delta - SG_k$ configuration if for every linearly independent $\ell_1, \ldots, \ell_k \in \mathcal{L}$ there are $\delta \cdot |\mathcal{L}|$ forms $\ell \in \mathcal{L}$ such that either*

1. *$\ell \in \operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k\}$,*
2. *or the linear space $\operatorname{span}_{\mathbb{C}} \{\ell_1, \ldots, \ell_k, \ell\}$ is not elementary.*

Given the above definitions, the following theorem was proved in [12, Theorem 1.14], improving on [5].

▶ **Theorem 30.** *If $\mathcal{L}$ is a $\delta - SG^*_k$ configuration then $\dim \operatorname{span}_{\mathbb{C}} \{\mathcal{L}\} = \mathcal{O}\left(k/\delta\right)$. If $\mathcal{L}$ is a $\delta - SG_k$ configuration then $\dim \operatorname{span}_{\mathbb{C}} \{\mathcal{L}\} = \mathcal{O}\left(C^k/\delta\right)$ where C is a universal constant independent of k.*

In the case when $k = 1$, Definition 28 and Definition 29 coincide, and match the usual notion of robust linear Sylvester–Gallai configurations. In this case, the constant C is explicit.

▶ **Theorem 31** ([11, Theorem 1.6]). *If $\mathcal{L}$ is a $\delta - SG_1$ configuration then $\dim \operatorname{span}_{\mathbb{C}} \{\mathcal{L}\} \leq 4/\delta$.*

▶ Remark 32. Note that in [5, 12], the SG configurations are described in terms of points in $\mathbb{C}^n$, instead of linear forms in S. Both settings are equivalent via duality between points in $\mathbb{C}^n$ and linear forms in S_1.

3.2 Radical Sylvester-Gallai configurations

We now define the higher dimension analogues of the above configurations. Let $\mathcal{F}$ be a finite set of irreducible forms of degree at most d that are pairwise non-associate.

▶ **Definition 33** (Relevant sets). *Let $\mathcal{P} = \{P_1, \ldots, P_t\}$ be a set of forms in $S_{\leq d}$. We say that $\mathcal{P}$ is relevant if for every $1 \leq i \leq t$, $P_i \notin \operatorname{rad}(\mathcal{P} \setminus P_i)$.*

A relevant set of forms of size k is called a k-relevant *set.*

Geometrically, a set $\mathcal{P}$ is relevant if no subset of $\mathcal{P}$ define the same variety as $\mathcal{P}$. We can now extend Definition 28 and Definition 29 to configurations with forms of higher degree.

▶ **Definition 34** (k-ordinary set). *Let $\mathcal{P} \subset \mathcal{F}$ be a k-relevant set. We say that $\mathcal{P}$ is k-ordinary with respect to $\mathcal{F}$ if there are forms $F_1, \ldots, F_k \in \mathcal{F}$ such that*

$$\operatorname{rad}(\mathcal{P}) \cap \mathcal{F} \subset \operatorname{rad}(F_1, \ldots, F_{k-1}) \cup \{F_k\}.$$

▶ **Definition 35** (k-elementary set). *Let $\mathcal{P} \subset \mathcal{F}$ be a k-relevant set. We say that $\mathcal{P}$ is k-elementary with respect to $\mathcal{F}$ if $\operatorname{rad}(\mathcal{P}) \cap \mathcal{F} = \mathcal{P}$.*

▶ **Definition 36** (Radical Sylvester Gallai condition for tuples). *Let $\mathcal{F} := \{F_1, \ldots, F_m\} \subset S_{\leq d}$ be a finite set of irreducible forms and $k \in \mathbb{N}$. We say that $\mathcal{F}$ is a $\delta - \mathrm{SG}_k^*(d)$ configuration if for every $i \neq j$ we have $F_i \notin (F_j)$ and for every k-relevant subset $\mathcal{P} \subset \mathcal{F}$, there are $\delta(m-k)$ many forms $F \in \mathcal{F} \setminus \mathcal{P}$ such that either*

- *$F \in \operatorname{rad}(\mathcal{P})$ or*
- *$\operatorname{rad}(F, \mathcal{P}) \cap \mathcal{F}$ contains a form R not in $\operatorname{rad}(\mathcal{P}) \cup \{F\}$.*

Note that the robust SG problem from [31, 16] is the $\delta - \mathrm{SG}_1^*(2)$. The higher codimensional radical SG problem for quadratics that we address here can be stated as follows: what is the maximum vector space dimension of any $1 - \mathrm{SG}_k^*(2)$ configuration? Our main theorem, which we now formally state, gives an answer to this question.

▶ **Theorem 37** (Radical SG Theorem for tuples of quadratics). *Let $\mathcal{F}$ be a $1 - \mathrm{SG}_k^*(2)$ configuration. There is a universal constant $c > 0$ such that $\dim(\operatorname{span}_{\mathbb{C}}\{\mathcal{F}\}) \leq 3^{c \cdot 4^k}$.*

4 Commutative algebraic preliminaries

4.1 Basic Definitions

In this section we recall the necessary definitions and results needed from commutative algebra and algebraic geometry [3, 14].

▶ **Definition 38** (Regular sequence). *Let R be a commutative ring with unity. A sequence of elements $f_1, f_2, \cdots f_n \in R$ is called a regular sequence if*

(1) *$(f_1, f_2, \cdots, f_n) \neq R$, and*

(2) *for all $i \in [n]$, we have that f_i is a non-zero divisor on $R/(f_1, \cdots, f_{i-1})R$.*

Ideals generated by regular sequences are well-behaved. For example, if $f_1, \cdots, f_m$ is a regular sequence in $S = \mathbb{C}[x_1, \cdots, x_n]$, we know that the ideal $I = (f_1, \cdots, f_m)$ is Cohen-Macaulay [14, Proposition 18.13]. Cohen-Macaulayness imposes a simple and well-behaved structure on the primary decomposition of I. In particular, every associated prime of I is a minimal prime and the height/codimension of every minimal prime of I is the same, i.e. Cohen-Macaulay ideals are unmixed and equidimensional [14, Corollaries 18.11, 18.14].

We note that if $f_1, \cdots, f_m$ is a regular sequence of forms in S, then $f_1, \cdots, f_m$ are algebraically independent. Therefore the subalgebra generated by $f_1, \cdots, f_m$ is isomorphic to a polynomial ring. In particular, the ring homomorphism $\mathbb{C}[y_1, \cdots, y_m] \to S$ defined by $y_i \mapsto f_i$ is an isomorphism onto its image.

Even though the $\mathbb{C}$-algebra $\mathbb{C}[f_1, \ldots, f_m] \subset S$ is isomorphic to a polynomial ring, its elements may not behave well when seen as elements of S. We next present a sufficient condition which will ensure to us that the subalgebra is well behaved with respect to S, in a way which we formalize later in Section 5.

▶ **Definition 39** (R_η-property). *Let η be a non-negative integer. We say that a Noetherian ring R satisfies the R_η property if the local ring $R_\mathfrak{p}$ is a regular local ring for all prime ideals $\mathfrak{p} \subset R$ such that* $\text{height}(\mathfrak{p}) \leq \eta$.

We recall the definition of an R_η-sequence below [2]. A subalgebra generated by an R_η-sequence has several inetresting properties such as intersection flatness, which were essential in [2, 28].

▶ **Definition 40.** *Let $\eta \in \mathbb{N}$ and R a Noetherian ring. A sequence of elements $f_1, \ldots, f_n \in R$ is called a prime sequence (respectively an R_η-sequence) if*

1. *$f_1, \cdots, f_n$ is a regular sequence, and*
2. *$R/(f_1, \cdots, f_i)$ is an integral domain (respectively, satisfies the R_η property) for all $i \in [n]$.*

▶ Remark 41. Note that a prime sequence in a ring R is also an R-regular sequence. Further, if R is a polynomial ring and $\eta \geq 1$, then any R_η-sequence is also a prime sequence.

4.2 Discriminant lemma

The following result provides an elimination theoretic criterion for a complete intersection ideal to be radical. It is a direct application of [28, Lemma 3.22].

▶ **Lemma 42.** *Let $\mathcal{A} = \mathbb{K}[x_1, \ldots, x_r, y_1, \ldots, y_s]$, $\mathcal{B} := \mathbb{K}[y_1, \ldots, y_s]$. Let $F_1, \cdots, F_k, P$ be a regular sequence of irreducible forms in $\mathcal{A}$ where $F_1, \cdots, F_k \in \mathcal{B}$. Suppose $P \in \mathcal{A} \backslash (y_1, \ldots, y_s)$. If $I = (F_1, \cdots, F_k) \subset \mathcal{B}$ is radical and $\text{disc}_{x_1}(P) \notin \mathfrak{q} \cdot S$ where $\mathfrak{q}$ is any minimal prime of I in $\mathcal{B}$, then the ideal $(F_1, \cdots, F_k, P)$ is radical in $\mathcal{A}$*

Proof. Let $\mathfrak{p}$ be a minimal prime of the ideal $(F_1, \cdots, F_k, P)$ in $\mathcal{A}$. Since $F_1, \cdots, F_k, P$ is a regular sequence we have $\text{codim}(\mathfrak{p}) = r + s - k - 1$. Let $\mathfrak{q} = \mathfrak{p} \cap \mathcal{B}$. Note that $\mathfrak{q}$ is a prime ideal containing $F_1, \cdots, F_k$ in $\mathcal{B}$. Therefore $\text{codim}(\mathfrak{q}) \geq s - k$. If $\text{codim}(\mathfrak{q}) > s - k$, then $\text{codim}(\mathfrak{q} \cdot \mathcal{A}) > r + s - k$. Since $\mathfrak{q} \cdot \mathcal{A} \subset \mathfrak{p}$, we must have $\mathfrak{q} \cdot \mathcal{A} = \mathfrak{p}$, which is a contradiction as $P \in \mathfrak{p}$, whereas $P \notin (y_1, \cdots, y_s)$. Therefore we must have that $\text{codim}(\mathfrak{q}) = s - k$. Then $\mathfrak{q}$ is a minimal prime of $(F_1, \cdots, F_k)$ in $\mathcal{B}$ and by [28, Lemma 3.22] we conclude that the ideal $(F_1, \cdots, F_k, P)$ is radical in $\mathcal{A}$. ◀

5 Wide vector spaces and relative linear spaces

5.1 Wide vector spaces and algebras

We now define the main object that we will use in order to prove that Sylvester-Gallai configurations are low dimensional: wide Ananyan-Hochster vector spaces. Such spaces were used in [28] to give a positive solution to the radical SG problem for cubic forms. Our definition is slightly simpler than the one from [28, Definition 4.8], as we don't need the multiplicative factor used there.

▶ **Definition 43** (Wide vector spaces). *A vector space $V = V_1 + V_2$ where $V_i \subset S_i$ is said to be r-wide if, for any nonzero $Q \in V_2$ we have $\operatorname{rank} Q \geq \dim V + r$. In this case, we also say that $\mathbb{C}[V]$ is an r-wide algebra.*

We note that an r-wide vector space is a special case of the (w, t)-wide AH vector spaces from [28]. An r-wide vector space is precisely a $(r, 1)$-wide AH vector space according to [28].

▶ **Proposition 44** ([28], Proposition 4.11). *Suppose $U = U_1 + U_2$ is a vector space in S and suppose $r \in \mathbb{N}$. There exists an r-wide vector space $V = V_1 + V_2$ with $\mathbb{C}[U] \subseteq \mathbb{C}[V]$ such that $\dim V_2 \leq \dim U_2$ and $\dim V_1 \leq 3^{\dim U_2 + 1} \cdot (r + \dim U)$.*

We now list some basic properties regarding these spaces. The first three of these are algebraic properties that show how these spaces are useful, and the next three show how we can build and modify these spaces, and how they behave with respect to projection.

▶ **Theorem 45** ([2], Theorem 1.10). *Let $V \subset S_2$ be a vector space of dimension d such that $\operatorname{rank} Q \geq d - 1 + \lceil \eta/2 \rceil$. Then every sequence of linearly independent elements of V is an R_η sequence.*

▶ **Corollary 46.** *Suppose $V = V_1 + V_2$ is a r-wide vector space with $r \geq 1$. If $\ell_1, \ldots, \ell_a$ is a linearly independent sequence in V_1 and $Q_1, \ldots, Q_b$ is a linearly independent subset of V_2, then the sequence $\ell_1, \ldots, \ell_a, Q_1, \ldots, Q_b$ is a prime sequence. In particular, the ideal $(Q_1, \ldots, Q_b)$ is a prime ideal in the quotient ring $S/(\ell_1, \ldots, \ell_a)$.*

Proof. That $\ell_1, \ldots, \ell_a$ form a prime sequence follows from the fact that they are independent linear forms. Let $U := \operatorname{span}_{\mathbb{C}} \{\overline{Q_1}, \ldots, \overline{Q_b}\}$ be the vector space spanned by $Q_1, \ldots, Q_b$ in $S/(\ell_1, \ldots, \ell_a)$. Every nonzero form in U has rank at least $\dim V_1 + \dim V_2 + r - a$, which is greater than $\dim U$. Therefore, by Theorem 45, the forms $\ell_1, \ldots, \ell_a, Q_1, \ldots, Q_b$ form a R_1 sequence. By [2, Discussion 1.3], such a sequence is also a prime sequence. The last statement follows by the definition of prime sequences (Definition 40). ◀

▷ Claim 47. Suppose $V := V_1 + V_2$ is r-wide with $V_i \subset S_i$. If $Q \in \mathbb{C}[V]$ is a quadratic form of rank less than r, then $Q \in \mathbb{C}[V_1]$. If $P \in (V)$ is a quadratic form of rank less than r, then $P \in (V_1)$.

Proof. Suppose $Q = Q_2 + Q_1$ with $Q_i \in \mathbb{C}[V_i]$. We have $Q_2 = Q - Q_1$ whence $\operatorname{rank} Q_2 \leq r + \dim V_1$. Therefore $Q_2 = 0$. Similarly, suppose $P = P_1 + P_2$ with $P_2 \in V_2$ and $P_1 \in (V_1)$. We have $P_2 = P - P_1$ whence $\operatorname{rank} P_2 \leq r + \dim V_1$. Therefore $P_2 = 0$. ◁

▶ Remark 48. Suppose $V = V_1 + V_2$ is a r-wide vector space, and suppose $U \subset S_1$ is a vector space of dimension k. We have $\dim V + U \leq \dim V + k$. Further, we have $(V + U)_2 = V_2$. For every $Q \in (V + U)_2$ we therefore have $\operatorname{rank} Q \geq (r - k) + \dim(V + U)$. Therefore $V + U$ is a $r - k$-wide vector space.

▶ Remark 49. Suppose $V = V_1 + V_2$ is a r-wide vector space and $\varphi := \varphi_{\alpha,V_1}$ is a projection mapping as defined in Definition 20. If $Q \in V_2$ is such that $\operatorname{rank} \varphi(Q) = a$ in $S\,[z]\,/\,(V_1)$ then $a - 1 \leq \operatorname{rank} Q \leq a$ in $S/\,(V_1)$. Since V is r-wide, this proves that $a \geq r + \dim V_2$. Since $\dim \phi(V_1) = 1$, and since $\dim \phi(V_2) \leq \dim V_2$, we get $a \geq r - 1 + (\dim \phi(V_1) + \dim \phi(V_2)$. This shows that $\phi(V)$ is at least $r - 1$ wide.

The following lemmas show that radical membership among linear forms and certain elements in the ideal (V) imply relationships between the "low rank" and "high rank" parts individually.

▶ **Lemma 50.** *Let $F_1, \ldots, F_k \in S_{\leq 2}$ be irreducible forms. Let $V = V_1 + V_2$ be r-wide with $r \geq k + 2$ and let $z \in V_1$. Suppose each F_i is either of the form $F_i = x_i$ with $x_i \in S_1$ or of the form $F_i = Q_i + zx_i$ with $Q_i \in V_2$ and $x_i \in S_1$. If*

$$F_k \in \operatorname{rad}(F_1, \ldots, F_{k-1})$$

then $zx_k \in (x_1, \ldots, x_{k-1})$ and $Q_k \in \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_{k-1}\}$ where $Q_i = 0$ if $F_i \in S_1$.

Proof. Let $U := (x_1, \ldots, x_k)$. In the ring S/U, the vector space V is $(r-k)$-wide by Remark 48. By Corollary 46, $(Q_1, \ldots, Q_{k-1})$ is a prime ideal in S/U. Therefore we have $Q_k = \sum_{i=1}^{k-1} \alpha_i Q$ in $S/\,(U)$ for $\alpha_i \in \mathbb{C}$. This implies $Q_k = \sum_{i=1}^{k-1} \alpha_i Q_i + E$ in S, where $E \in (U)$. Since $\operatorname{rank} E \leq \dim U \leq k$, and since V is r-wide, we must have $E = 0$, proving the first required statement.

Let $I := (Q_1, \ldots, Q_{k-1}, x_1, \ldots, x_{k-1})$. Since (U) is prime, and $(Q_1, \ldots, Q_{k-1})$ is prime in $S/\,(U)$, the ideal I is prime. Since $Q_k \in \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_{k-1}\}$ and since $F_i \in I$ for $i \leq k-1$, we have $zx_k \in I$. Since W is r-wide, this implies $zx_k \in (x_1, \ldots, x_{k-1})$, completing the proof. ◀

▶ **Lemma 51.** *Let $F_1, \ldots, F_k \in S_{\leq 2}$ be irreducible forms. Let $V = V_1 + V_2$ be r-wide with $r \geq k + 2$ and let $z \in V_1$. Suppose each F_i is either of the form $F_i = x_i$ with $x_i \in S_1$ or of the form $F_i = Q_i + zx_i$ with $Q_i \in V_2$ and $x_i \in S_1$. Suppose further that $z, x_1, \ldots, x_{k-1}$ are linearly independent. If*

$$F_k \in \operatorname{rad}(F_1, \ldots, F_{k-1}),$$

and if $x_k \in (x_1)$ in $S/\,(z)$, then $F_1 = F_k$.

Proof. First assume that $Q_1 \neq 0$. By relabelling $F_2, \ldots, F_k$ we can assume that $\operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_{k-1}\} = \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_t\}$ for some $t \leq k-1$. For each $i \in [t+1, k-1]$, suppose $Q_i = \sum_{j=1}^{t} \beta_{ij} Q_j$. For each such i, let $y_i := x_i - \sum_{j=1}^{t} \beta_{ij} x_j$. Note that $x_1, \ldots, x_t, y_{t+1}, \ldots, y_{k-1}$ are linearly independent in $S/\,(z)$. We have $(F_1, \ldots, F_{k-1}) = (F_1, \ldots, F_t, zy_{t+1}, \ldots, zy_{k-1})$. Let $J = (y_{t+1}, \ldots, y_{k-1})$. By Remark 48 the vector space V is $r-k$-wide in S/J, therefore $\operatorname{rank}(Q_1, \ldots, Q_t) \geq t + r - k$, and consequently $\operatorname{rank}(F_1, \ldots, F_t) \geq t + r - k - 1$. By Theorem 45, the ideal $(F_1, \ldots, F_t)$ is prime in S/J, therefore $(F_1, \ldots, F_t) + J$ is a prime ideal containing $\operatorname{rad}(F_1, \ldots, F_{k-1})$.

Let $x_k = x_1 + \alpha z$. Suppose $F_k \in S_2$. By Lemma 50 we have $Q_k \in \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_t\}$, say $Q_k = \sum_{j=1}^{t} \gamma_i Q_i$. We have $F_k - \sum_{j=1}^{t} \gamma_i Q_i = z(\alpha z + x_1 - \sum_{j=1}^{t} \gamma_j x_j) \in (F_1, \ldots, F_t) + J$. Since the latter ideal is a graded prime ideal, we have either $z \in J$ or $(\alpha z + x_1 - \sum_{j=1}^{t} \gamma_j x_j) \in J$. By the linear independence assumption on the x_i, this is only possible if $(\alpha z + x_1 - \sum_{j=1}^{t} \gamma_j x_j) = 0$. This implies $\alpha = 0$ and $\gamma_1 = 1$ and $\gamma_j = 0$ for $j \geq 2$. This implies $F_1 = F_k$ as required.

Suppose now that $F_k \in S_1$. We then have $F_k \in (F_1, \ldots, F_t) + J$, and therefore $x_1 + \alpha z \in J$, which contradicts the linear independence assumption.

We are left with the case when $Q_1 = 0$. After rearranging the forms, let $Q_2, \ldots, Q_t$ be such that $\operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_{k-1}\} = \operatorname{span}_{\mathbb{C}}\{Q_2, \ldots, Q_t\}$, and let y_i be defined as in the previous case. Note that $y_1 = x_1$. Let $J := (y_1, y_{t+1}, \ldots, y_{k+1})$. The ideal $(F_2, \ldots, F_t) + J$ is a prime ideal containing $\operatorname{rad}(F_1, \ldots, F_{k-1})$. As before suppose $x_k = x_1 + \alpha z$.

Suppose $F_k \in S_2$ so $Q_k = \sum_{j=2}^{t} \gamma_i Q_i$. We have $F_k - \sum_{j=2}^{t} \gamma_i F_i = z(x_1 + \alpha z - \sum_{j=2}^{t} \gamma_i x_i \in (F_2, \ldots, F_t) + J$. Therefore, either $z \in J$, or $x_1 + \alpha z - \sum_{j=2}^{t} \gamma_i x_i \in J$. By the linear independence assumption, this implies $\alpha_i = 0$ for $i = 2, \ldots, t$ contradicting the fact that $F_k \in S_2$.

Suppose $F_k \in S_1$. In this case we have $F_1 - F_k = \alpha z \in (F_2, \ldots, F_t) + J$, which implies $\alpha z \in J$. By the independence assuption, we must have $\alpha = 0$, whence $F_1 = F_k$ as required. ◀

5.2 Relative linear spaces

Now that we have proved some properties of wide vector spaces, we introduce the notion of relative linear spaces and establish some properties which will be useful to us in the next section. This notion of relative linear spaces was used in [16] in their proof of the robust SG theorem for quadratics.

▶ **Definition 52** (Forms close to a vector space). *Given a vector space $V = V_1 + V_2$ where $V_i \subseteq S_i$, we say that a quadratic form P is s-*close *to V if there is a form $Q \in \mathbb{C}[V]$ such that $\operatorname{rank}(P - Q) \leq s$. If a form P is not r-close to V, for any $r \leq s$, we say that P is s-*far *from V.*

Given a linear form ℓ, we say ℓ is 1-close to V if $\ell \notin V_1$.

▶ Remark 53. Given a set of forms $\mathcal{F}$, we will say that $\mathcal{F}$ is s-close to V if all forms in $\mathcal{F}$ are at most s-close to V.

▶ **Proposition 54** (Quadratics close to wide vector spaces). *Let $V = V_1 + V_2$ be an r-wide vector space and $s < r/2$. If P is s-close to V, then for any $Q, Q' \in \mathbb{C}[V]$ such that $\operatorname{rank}(P - Q) = \operatorname{rank}(P - Q') = s$, we have that*

$$\operatorname{Lin}(P - Q) + V_1 = \operatorname{Lin}(P - Q') + V_1.$$

In other words, $(\operatorname{Lin}(P - Q) + V_1)/V_1 = (\operatorname{Lin}(P - Q') + V_1)/V_1$ for any two decompositions.

Proof. Let $R = P - Q$ and $R' = P - Q'$. Thus, we have that $R - R' = Q' - Q \in \mathbb{C}[V]$ and we have $\operatorname{rank}(Q' - Q) = \operatorname{rank}(R - R') \leq \operatorname{rank}(R) + \operatorname{rank}(R') \leq 2s < r$. Hence, by Remark 47, we have that $Q' - Q \in \mathbb{C}[V_1]$. Now, from $R = R' + (Q' - Q)$ and $Q' - Q \in \mathbb{C}[V_1]$, we have that $\operatorname{Lin}(R) \subseteq \operatorname{Lin}(R') + V_1$, and similarly, we have that $\operatorname{Lin}(R') \subseteq \operatorname{Lin}(R) + V_1$. ◀

▶ **Definition 55** (Relative space of linear forms). *Let r, B be integers such that $r > 2B + 1$. If V is an r-wide vector space and P is s-close to V for $s < r/2$ we can define*

$$\mathbb{L}_V(P) := \begin{cases} \mathbb{L}(P) + V_1, & \text{if } P \in S_1 \\ \operatorname{Lin}(P - Q) + V_1, & \text{if } P \in S_2, s \leq B \\ \operatorname{span}_{\mathbb{C}}\{P\}, & \text{otherwise} \end{cases}$$

where $Q \in \mathbb{C}[V]$ is a form such that $\operatorname{rank}(P - Q) = s$. We also define the quotient space

$$\overline{\mathbb{L}}_V(P) := \begin{cases} \mathbb{L}_V(P)/V_1, & \text{if } s \leq B \\ 0, & \text{otherwise} \end{cases}$$

Further, we define P_V^H to be the unique polynomial in V_2 such that $P - P_V^H$ is s-close to V_1. Finally we define $P_V^L = P - P_V^H$. Note that $\mathbb{L}_V(P) = \mathbb{L}_V\left(P_V^L\right)$. The superscript H indicates that P_V^H is the high-rank part of P with respect to V and the superscript L indicates that P_V^L is the low-rank part of P with respect to V.

Note that while the definition of $\mathbb{L}_V(P)$ depends on the parameter B, we suppress this from the notation for brevity. It will be clear from context the value of the parameter B whenever we use $\mathbb{L}_V(P)$.

Here are some useful results about relative linear spaces, and how they change when V is modified. Lemma 58 characterises exactly when $\dim \overline{\mathbb{L}}_V(F)$ is unchanged when $\mathbb{L}_V(G)$ is added to V_1. As the lemma shows, this happens when F and G do not share any common variables other than those that occur in V.

▶ **Proposition 56.** *Suppose V is a r-wide space and P is s-close to V for $2s < r$. If $P \in (V)$ then $P_V^H \in V_2$ and $P_V^L \in (V_1)$.*

Proof. Since P is s-close to V we can write $P = P_V^H + P_V^L$. Since $P_V^H \in V_2$, we have $P_V^L \in (V)$ by assumption. We can write $P_V^L = P_2 + P_1$ with $P_1 \in (V_1)$ and $P_2 \in V_2$. In $S/(V_1)$ we have $P_2 = 0$. Since V is r-wide, this implies $P_2 = 0$ in S. Therefore $P_V^L \in (V_1)$. ◀

▶ **Proposition 57.** *Suppose $V = V_1 + V_2$ is a r-wide vector space with $r > 2B + 1$, and suppose $P \in S_2$ is B-close to V. Then $Y := \mathbb{L}_V(P) + V_2$ is a $r - 2B$ wide vector space. If further $r > 4B + 1$ then for any other polynomial Q that is also B close to V we have $Q_V^H = Q_Y^H$.*

Proof. The first statement follows since Y is obtained by adding at most $2B$ linear forms to a basis of V. We now have $Q = Q_V^H + Q_V^L = Q_Y^H + Q_Y^L$ whence $Q_V^H - Q_Y^H = Q_Y^L - Q_V^L$. Here, we use the fact that $B < 4r + 1$ to ensure that Q_Y^H, Q_Y^L are well defined. Since both Q_Y^L, Q_V^L have rank at most B in $S/(Y)$ we obtain that $Q_V^H = Q_Y^H$. ◀

▶ **Lemma 58.** *Suppose $V = V_1 + V_2$ is a r-wide vector space with $r > 4B + 1$, and suppose $F, G \in S_2$ are both B close to V. Let $Y := \mathbb{L}_V(G) + V_2$. Then the following hold.*

1. $\mathbb{L}_Y(F) = \mathbb{L}_V(G) + \mathbb{L}_V(F)$.
2. $\dim \overline{\mathbb{L}}_V(F) = \dim \overline{\mathbb{L}}_Y(F)$ *if and only if* $\overline{\mathbb{L}}_V(F) \cap \overline{\mathbb{L}}_V(G) = \{0\}$.
3. *If $F \notin (V)$ and $\dim \overline{\mathbb{L}}_V(F) = \dim \overline{\mathbb{L}}_Y(F)$ then $F \notin (Y)$.*

Proof. By Proposition 57 we have $H := F_V^H = F_Y^H$. Let P, R be such that $F - H - P = R$ with $P \in \mathbb{C}[V_1]$ and $\mathbb{L}_V(F) = \mathrm{Lin}(R) + V_1$. Let P', R' be such that $F - H - P' = R'$ with $P' \in \mathbb{C}[Y_1]$ and $\mathbb{L}_Y(F) = \mathrm{Lin}(R') + Y_1$. We have the equation $R' + P' = R + P$, which implies that $\mathrm{Lin}(R') + Y_1 = \mathrm{Lin}(R) + Y_1$. Since $V_1 \subset Y_1$, we have

$$\mathrm{Lin}(R') + Y_1 = \mathrm{Lin}(R) + V_1 + Y_1. \tag{1}$$

Substituting $\mathbb{L}_V(F), \mathbb{L}_Y(F)$ in Equation (1) and using the fact that $Y_1 = \mathbb{L}_V(G)$ we get $\mathbb{L}_Y(F) = \mathbb{L}_V(G) + \mathbb{L}_V(F)$.

Equation (1) also implies

$$\overline{\mathbb{L}}_Y(F) = \frac{\mathrm{Lin}(R') + Y_1}{Y_1} = \frac{\mathbb{L}_V(F) + Y_1}{Y_1} = \frac{\mathbb{L}_V(F)}{\mathbb{L}_V(F) \cap Y_1}$$

therefore

$$\begin{aligned} \dim \overline{\mathbb{L}}_V(F) = \dim \overline{\mathbb{L}}_Y(F) &\iff \dim V_1 = \dim\left(\mathbb{L}_V(F) \cap Y_1\right) \\ &\iff V_1 = \mathbb{L}_V(F) \cap Y_1 \qquad \text{(since } V_1 \subseteq \mathbb{L}_V(F) \cap Y_1\text{)} \\ &\iff \{0\} = \overline{\mathbb{L}}_V(F) \cap \overline{\mathbb{L}}_V(G), \end{aligned}$$

proving the second item.

Assume now that $F \in (Y)$. By Proposition 56 we have $F - H \in (Y_1)$. Further, by assumption we have $F - H \notin (V_1)$. In $S/(V_1)$ we have $0 \neq \overline{F-H} = \overline{R} \in (Y_1)$ which in turn implies that $\mathrm{Lin}\left(\overline{R}\right) \cap Y_1 \neq \{0\}$ by Lemma 19. We have $R = \sum a_i b_i$ for linear forms $a_1, \ldots, a_t, b_1, \ldots, b_t$ where $\overline{a_i}, \overline{b_j}$ span $\overline{\mathbb{L}}_V(F)$. Therefore $\mathrm{Lin}\left(\overline{R}\right) = \sum \overline{a_i}\overline{b_i}$, whence $\mathrm{Lin}\left(\overline{R}\right) \subseteq \overline{\mathbb{L}}_V(F)$. This shows that $\overline{\mathbb{L}}_V(F) \cap \overline{\mathbb{L}}_V(G) \neq 0$, which by item 2 implies $\dim \overline{\mathbb{L}}_V(F) \neq \dim \overline{\mathbb{L}}_Y(F)$, contradicting the assumption. ◀

Note that the condition $\overline{\mathbb{L}}_V(F) \cap \overline{\mathbb{L}}_V(G) = \{0\}$ is symmetric in F and G. Therefore, we have that $\dim \overline{\mathbb{L}}_V(F) = \dim \overline{\mathbb{L}}_{\mathbb{L}_V(G)+V_2}(F)$ if and only if $\dim \overline{\mathbb{L}}_V(G) = \dim \overline{\mathbb{L}}_{\mathbb{L}_V(F)+V_2}(G)$. Further, in this case we have $F \notin (\mathbb{L}_V(G), V_2)$ and also $G \notin (\mathbb{L}_V(F), V_2)$ if $F, G \notin (V)$. In the next subsection, we introduces the notion of integral sequences that generalises the above.

6 Integral sequences and strong sequences

In this section we define two special types of sequences of forms, namely integral sequences and strong sequences. We will use the strong sequences to construct our core algebra, that is, to prove that there is a small algebra such that all quadratics are close to it. We will then use integral sequence to handle the case where all the quadratics are close to a core algebra. We will prove that the ideals generated by integral and strong sequences are always radical and prime, respectively.

6.1 Integral sequences

Item 2 of Lemma 58 gives us a condition for when the relative linear spaces of two linear forms are disjointed. Intuitively, this is equivalent to the forms depending on disjoint sets of variables, other than those occurring in V. This is made formal in Corollary 62. The notion of integral sequences extends this to more that two forms. As in Lemma 58, we will require the forms to be close to a wide vector space for the notion to be well defined.

▶ **Definition 59** (Integral Sequences). *Let r, B, t be integers with $r > 4tB + 1$. Suppose $V = V_1 + V_2$ is a r-wide vector space. Let $F_1, \ldots, F_t \in \mathcal{F}$ be a sequence of forms that are B-close to V. Let $U_0 := V$ and let $U_i := \mathbb{L}_{U_{i-1}}(F_i) + V_2$. The sequence $F_1, \ldots, F_t$ is called an integral sequence over V if for each i we have*

- $\dim \overline{\mathbb{L}}_V(F_i) = \dim \overline{\mathbb{L}}_{U_{i-1}}(F_i)$, *and*
- $F_i \notin (V)$

When V is clear from context we just call $F_1, \ldots, F_t$ an integral sequence.

In the rest of this section, we will assume that $r > 4tB + 1$.

▶ **Proposition 60.** *Suppose V is a r-wide vector space. Suppose $F_1, \ldots, F_t$ are a sequence of forms, and suppose $U_i := \mathbb{L}_{U_{i-1}}(F_i) + V_2$ with $U_0 := V$. Then*

1. $U_t = \sum_{j=1}^{t} \mathbb{L}_V(F_j) + V_2$.
2. $\dim \overline{\mathbb{L}}_V(F_i) = \dim \overline{\mathbb{L}}_{U_{i-1}}(F_i)$ *for every $2 \leq i \leq t$ if and only if for every $2 \leq i \leq t$ we have*

$$\overline{\mathbb{L}}_V(F_i) \cap \left(\sum_{j=1}^{i-1} \overline{\mathbb{L}}_V(F_j) \right) = \{0\}.$$

3. *If additionally $F_i \notin (V)$ for every $1 \leq i \leq t$, then $F_i \notin (U_{i-1})$ for $2 \leq i \leq t$.*

Proof. We prove the statements by induction on t. We will prove the additional statement that $\mathbb{L}_{U_{t-1}}(F_t) = \sum_{i=1}^{t} \mathbb{L}_V(F_t)$. Each of the three items are true by definition when $t = 1$. Suppose the statements are true for $t-1$.

Now the space U_{t-2} is $4B+1$ wide by Remark 48. Applying Lemma 58 to U_{t-2}, F_t, and F_{t-1} we can deduce that $\mathbb{L}_{U_{t-1}}(F_t) = \mathbb{L}_{U_{t-2}}(F_t) + \mathbb{L}_{U_{t-2}}(F_{t-1})$. The space U_{t-3} is also $4B+1$ wide, therefore applying Lemma 58 to U_{t-3}, F_t, and F_{t-2} we can deduce that $\mathbb{L}_{U_{t-2}}(F_t) = \mathbb{L}_{U_{t-3}}(F_t) + \mathbb{L}_{U_{t-3}}(F_{t-2})$. Repeating this and substituting, we deduce that $\mathbb{L}_{U_{t-1}}(F_t) = \mathbb{L}_V(F_t) + \sum_{i=2}^{t} \mathbb{L}_{U_{t-i}}(F_{t-i+1})$. By the induction hypothesis, we get that $\mathbb{L}_{U_{t-i}}(F_{t-i+1}) = \sum_{j=1}^{t-i} \mathbb{L}_V(F_j)$. Therefore we get $\mathbb{L}_{U_{t-1}}(F_t) = \sum_{i=1}^{t} \mathbb{L}_V(F_i)$. The first item now follows by adding V_2 to both sides.

Suppose now that $\dim \overline{\mathbb{L}}_V(F_i) = \dim \overline{\mathbb{L}}_{U_{i-1}}(F_i)$ for every $2 \le i \le t-1$. Suppose $\dim \overline{\mathbb{L}}_V(F_t) = \dim \overline{\mathbb{L}}_{U_{t-1}}(F_t)$. This implies $\dim \overline{\mathbb{L}}_{U_{t-1}}(F_t) = \dim \overline{\mathbb{L}}_{U_{t-2}}(F_t)$, since $V \subset U_{t-2} \subset U_{t-1}$. By item 2 of Lemma 58 applied to U_{t-2}, F_{t-1}, F_t we can deduce that $\overline{\mathbb{L}}_{U_{t-2}}(F_t) \cap \overline{\mathbb{L}}_{U_{t-2}}(F_{t-1}) = \{0\}$. Using the fact that $\mathbb{L}_{U_{t-1}}(F_t) = \sum_{i=1}^{t} \mathbb{L}_V(F_i)$, this is equivalent to $\overline{\mathbb{L}}_V(F_i) \cap \left(\sum_{j=1}^{i-1} \overline{\mathbb{L}}_V(F_j)\right) = \{0\}$. Conversely, starting with this assumption we can deduce that $\dim \overline{\mathbb{L}}_{U_{t-1}}(F_t) = \dim \overline{\mathbb{L}}_{U_{t-2}}(F_t)$. Note that $F_1, \ldots, F_{t-2}, F_t$ also satisfy the conditions of item 2. Therefore, by induction we can deduce that $\dim \overline{\mathbb{L}}_{U_{t-2}}(F_t) = \dim \overline{\mathbb{L}}_V(F_t)$. This completes the proof of item 2.

Applying the induction hypothesis to $F_1, \ldots, F_{t-2}, F_t$, we can deduce that $F_t \notin (U_{t-2})$. We can now apply Lemma 58 to U_{t-2}, F_t and F_{t-1} to deduce that $F_t \notin (U_{t-1})$, proving item 3. ◀

▶ **Corollary 61.** *If $F_1, \ldots, F_t$ is a integral sequence, then so is any permutation of $F_1, \ldots, F_t$.*

Proof. The second condition for integral sequences holds irrespective of the order of the forms. By Proposition 60, the first condition for integral sequences is equivalent to

$$\overline{\mathbb{L}}_V(F_i) \cap \left(\sum_{j=1}^{i-1} \overline{\mathbb{L}}_V(F_j)\right) = \{0\}$$

for every $2 \le i \le t$. This in turn is equivalent to $\dim \sum_{j=1}^{t} \overline{\mathbb{L}}_V(F_j) = \sum_{j=1}^{t} \dim \overline{\mathbb{L}}_V(F_j)$, which is independent of the order of the forms. ◀

▶ **Corollary 62.** *Let $F_1, \ldots, F_t$ be an integral sequence with respect to V and $\mathcal{A} := \mathbb{C}\left[V_2, \sum_{i=1}^{t} \mathbb{L}_V(F_i)\right]$. There exist vector spaces of linear forms $Y_1, \ldots, Y_t \subset \mathcal{A}$ such that $Y_i \cap (V + Y_1 + \cdots + Y_{i-1}) = (0)$ for all i and $F_i \in \mathbb{C}[V, Y_i]$. Furthermore, $F_i \notin (V, Y_1, \cdots, Y_{i-1}, Y_{i+1}, \cdots, Y_t)$.*

Proof. By Proposition 60 we can take $Y_j := \overline{\mathbb{L}}_V(F_j)$. By Corollary 61, we may switch F_i and F_t. Then by Proposition 60 part (3), we see that $F_i \notin (V, Y_1, \cdots, Y_{i-1}, Y_{i+1}, \cdots, Y_t)$. ◀

▶ **Lemma 63.** *Suppose V is a r-wide vector space and $F_1, \ldots, F_t$ is an integral sequence with respect to V. Suppose $F_0 \in \mathbb{C}[V] \setminus \{0\}$. Then $F_0, F_1, \ldots, F_t$ is a regular sequence in S.*

Proof. Note that by Corollary 62 we may assume that there exist vector spaces of linear forms $Y_1, \ldots, Y_t$ of $\mathcal{A}$ such that $Y_i \cap (V + Y_1 + \cdots + Y_{i-1}) = (0)$ and $F_i \in \mathbb{C}[V, Y_i]$. Let $U = V + Y_1 + \cdots + Y_t$ and $\mathcal{A} = \mathbb{C}[U]$. Since V is r-wide and $r > 4Bt + 1$, we know that U is $2Bt+1$-wide, and hence has a basis consisting of a prime sequence. Thus $\mathcal{A} \to S$ is a free extension (see [2, Section 2]) and hence any regular sequence in $\mathcal{A}$ is also a regular sequence in S (see [38, Tag 00LM]). Therefore it is enough to prove that $F_0, F_1, \cdots, F_t$ is a regular sequence in $\mathcal{A} = \mathbb{C}[U]$.

Note that the element F_0 is a regular sequence in $\mathcal{A} = \mathbb{C}[V + Y_1 + \cdots + Y_t]$. We will prove by induction that if $F_0, \cdots, F_i$ is a regular sequence in $\mathcal{A}$, then so is $F_0, \cdots, F_{i+1}$. Suppose $F_0, \cdots, F_i$ is a regular sequence in $\mathcal{A}$ (and hence in $\mathcal{A}_i = \mathbb{C}[V + Y_1 + \cdots + Y_i]$). If F_{i+1} is a zero divisor in $\mathcal{A}/(F_0, \cdots, F_i)$, then F_{i+1} is in a minimal prime $\mathfrak{p}$ of $(F_0, \cdots, F_i)$ in $\mathcal{A}$. Since $\mathcal{A}_i \to \mathcal{A}$ is a free extension and $\mathcal{A}_i$ is generated by a prime sequence in S, we must have that $\mathfrak{p} = \mathfrak{q} \cdot \mathcal{A}$ for some minimal prime $(F_0, \cdots, F_i) \subset \mathfrak{q}$ in $\mathcal{A}_i$. Note that by Proposition 60 we know that $F_{i+1} \notin (V + Y_1 + \cdots + Y_i)$. This is a contradiction since $F_{i+1} \in \mathfrak{q} \cdot \mathcal{A}$ and $\mathfrak{q} \subset (V + Y_1 + \cdots + Y_i)$. ◀

▶ **Lemma 64.** *Suppose V is a r-wide vector space and $F_1, \ldots, F_t$ is an integral sequence with respect to V. Then $(F_1, \ldots, F_t)$ is radical and for any minimal prime $\mathfrak{p} \supset (F_1, \ldots, F_t)$ we have that $\mathfrak{p} \cap \mathbb{C}[V] = (0)$.*

Proof. Note that by Corollary 62 we may assume that there exist vector spaces of linear forms $Y_1, \ldots, Y_t$ of $\mathcal{A}$ such that $Y_i \cap (V + Y_1 + \cdots + Y_{i-1}) = (0)$ and $F_i \in \mathbb{C}[V, Y_i]$. By Lemma 63, we know that $F_1, \cdots, F_t$ is a regular sequence. Hence $\mathrm{ht}(\mathfrak{p}) = t$ for any minimal prime $\mathfrak{p} \supset (F_1, \ldots, F_t)$. Let F_0 be a non-zero element in $\mathbb{C}[V]$. Then $F_0, \cdots, F_t$ is again a regular sequence and hence $\mathrm{ht}(F_0, \cdots, F_t) = t + 1$. This implies $F_0 \notin \mathfrak{p}$, as $\mathrm{ht}(\mathfrak{p}) = t$ implies that $\mathfrak{p}$ contains no regular sequence of length $t + 1$. Therefore we must have that $\mathfrak{p} \cap \mathbb{C}[V] = (0)$.

Now we will show that $(F_1, \cdots, F_t)$ is a radical ideal in S. Let $\mathcal{A} = \mathbb{C}[V + Y_1 + \cdots + Y_t]$. Since $\mathcal{A} \to S$ is a free extension and the generators of $\mathcal{A}$ form a prime sequence in S, it is enough to prove that $(F_1, \cdots, F_t)$ is radical in $\mathcal{A}$.

For each i, we assume that F_i is monic in $y_i \in Y_i$ after a possible change of coordinates in Y_i. There exists such a variable since $F_i \notin (V)$. Let $U_i := Y_i / \mathrm{span}_{\mathbb{C}} \{y_i\}$ and $Z = V + U_1 + \cdots + U_t$. Then $\mathcal{A} = \mathbb{C}[Z, y_1, \cdots, y_t]$. We will show by induction that $(F_1, \cdots, F_t)$ is radical.

Note that (F_1) is prime. Assume the statement holds for $i - 1$. We have $\mathrm{disc}_{y_i}(F_i) \in \mathbb{C}[V, U_i]$. Note that $\mathfrak{p} \cap \mathbb{C}[V, U_i] = (0)$ for every minimal prime $\mathfrak{p}$ of $(F_1, \ldots, F_{i-1})$, as $F_{i-1} \notin (V, Y_1, \cdots, Y_{i-2}, Y_i)$ by Corollary 62. Therefore Lemma 42 implies $(F_1, \ldots, F_i)$ is radical. ◀

▶ **Corollary 65.** *Suppose $F_1, \ldots, F_t$ is an integral sequence with respect to V. Then $F_1, \ldots, F_t$ form a t-relevant set.*

Proof. The sequence $F_1, \ldots, F_{t-1}$ is an integral sequence, and therefore by Lemma 63 it is a regular sequence. Since any regular sequence is a relevant set, we are done. ◀

6.2 Strong sequences

Integral sequences are only defined when the forms are close to a wide vector space. One special case is when every form is of low rank, and therefore every form is close to the vector space $\{0\}$. To deal with forms that are not close to a vector space (which is the general case), we introduce the notion of strong sequences.

We first extend the notion of the rank of a quadratic form to vector spaces of quadratic forms.

▶ **Definition 66.** *Let $V_2 \subset S_2$ be a vector space. Define* $\mathrm{minrank}(V)$ *as* $\min_{Q \in V_2, Q \neq 0} \mathrm{rank}\, Q$. *If $Q_1, \ldots, Q_t$ are quadratic forms then define* $\mathrm{minrank}(Q_1, \ldots, Q_t) = \mathrm{minrank}(\mathrm{span}_{\mathbb{C}} \{Q_1, \ldots, Q_t\})$.

▶ **Definition 67.** *Let $k, t \in \mathbb{N}$ be such that $t \leq k+1$. Given forms $Q_1, \ldots, Q_t \in S_2$ we say that $Q_1, \ldots, Q_t$ is a* k-strong *sequence if $Q_1, \ldots, Q_t$ are linearly independent and* $\operatorname{minrank}(Q_1, \ldots, Q_t) \geq k+5$.

▶ Remark 68. By Theorem 45, if $Q_1, \ldots, Q_t$ is k-strong then $Q_1, \ldots, Q_t$ is a R_3 sequence. By the discussion in [2, Discussion 1.3], the ideal $(Q_1, \ldots, Q_t)$ is prime and the ring $S/(Q_1, \ldots, Q_t)$ is a UFD.

▶ **Lemma 69.** *Suppose $\mathcal{F}_2 \subset S_2$, and suppose $Q_1, \ldots, Q_t$ is a maximal* k-strong *sequence in $\mathcal{F}_2$ with $t \leq k$. For any $r \geq 2(k+5)$ there exists a r-wide vector space W with $\dim W_1 \leq 7 \cdot r \cdot 3^t, \dim W_2 \leq t$ such that every $Q \in \mathcal{F}_2$ is $k+4$-close to W.*

Proof. Let $U := \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_t\}$. By Proposition 44, there exists r-wide vector space W such that $U \subset \mathbb{C}[W]$, $\dim W_1 \leq 3^{t+1} \cdot (r+t)$ and $\dim W_2 \leq t$. Let $Q \in \mathcal{F}_2$ be a form. Consider the sequence $Q_1, \ldots, Q_t, Q$, which has length at most $k+1$. By assumption, $Q_1, \ldots, Q_t, Q$ is not a k-strong sequence. Therefore, we have either $\operatorname{minrank}(Q_1, \ldots, Q_t, Q) \leq k+4$ or $Q \in \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_t\}$.

Suppose $P = \beta Q + \sum \alpha_i Q_i$ is such that $\operatorname{rank} P = \operatorname{minrank}(Q_1, \ldots, Q_t, Q) \leq k+4$. Since $Q_1, \ldots, Q_t$ is k-strong we have $\beta \neq 0$. Therefore after scalar multiple we have $Q = \sum \alpha_i Q_i + P$, and Q is $k+4$-close to W. If $Q \in \operatorname{span}_{\mathbb{C}}\{Q_1, \ldots, Q_t\}$ then $Q \in W$ and therefore Q is $k+4$-close to W. ◀

We now define the notion of strong Sylvester-Gallai configurations. We show that a constant fraction of the forms in any such configuration is close to a vector space of constant dimension.[5]

▶ **Definition 70.** *Let $\mathcal{F}_2 \subset S_2$ be a finite set of forms. Let $0 < \epsilon \leq 1$ and $k, t \in \mathbb{N}$ with $t \leq k$. We say that $\mathcal{F}_2$ is a* strong $(\epsilon, k) - \mathrm{SG}^*_t(2)$ *configuration if for every* k-strong *sequence $Q_1, \ldots, Q_t$ with $Q_i \in \mathcal{F}_2$, there are $\epsilon(|\mathcal{F}_2| - 1)$ forms $Q_{t+1} \in \mathcal{F}_2$ such that either:*

1. *$Q_1, \ldots, Q_t, Q_{t+1}$ is not a* k-strong *sequence, or*
2. *there is a form $R \in \mathcal{F}_2$ such that $R \in (Q_1, \ldots, Q_{t+1}) \setminus (Q_1, \ldots, Q_t) \cup (Q_{t+1})$.*

▶ **Lemma 71.** *Let $\mathcal{F}_2 \subset S_2$ finite, with $m := |\mathcal{F}_2|$. Let $0 < \epsilon \leq 1$ and $k, t \in \mathbb{N}$ with $2 \leq t \leq k$. If $\mathcal{F}_2$ is a* strong $(\epsilon, k) - \mathrm{SG}^*_t(2)$ *configuration then either*

1. *$\mathcal{F}_2$ is a* strong $(\epsilon/4, k) - \mathrm{SG}^*_{t-1}(2)$ *configuration, or*
2. *there exist a vector space W with $\dim W_1 \leq 7 \cdot r \cdot 3^{t+1+16/\epsilon}, \dim W_2 \leq t + 1 + 16/\epsilon$ such that at least $\epsilon m/4$ forms in $\mathcal{F}_2$ are $k+4$ close to W.*

Proof. Let $\epsilon' := \epsilon/4$. Suppose $\mathcal{F}_2$ is not a strong $(\epsilon', k) - \mathrm{SG}^*_{t-1}(2)$ configuration. If there exist no k-strong sequences of length $t-1$, then there exists some maximal k-strong sequence of length at most $t-2$, and the required space W exists by Lemma 69. We can therefore assume that there exists a k-strong sequence $Q_1, \ldots, Q_{t-1}$, and a set $\mathcal{B} \subset \mathcal{F}_2$ of size at least $(1-\epsilon')m$ such that for every $Q \in \mathcal{B}$ we have that $Q_1, \ldots, Q_{t-1}, Q$ is a k-strong sequence, and

$$\mathcal{F}_2 \cap (Q_1, \ldots, Q_{t-1}, Q) \setminus (Q_1, \ldots, Q_{t-1}) = \{Q\}. \tag{2}$$

[5] As we mentioned in Section 1, we need this notion of strong SG configurations since in our setting we cannot quotient by quadratic forms, as the quotient ring will not be a polynomial ring and the previous results on SG configurations may not apply. In particular, this is where our approach is more complex than [5], as in their case their quotients were all isomorphic to polynomial rings.

Let $V := \mathrm{span}_{\mathbb{C}}\{Q_1, \ldots, Q_{t-1}\}$. Forms $P_1, P_2 \in \mathcal{B}$ are pairwise independent over S_2/V, since if $(P_1) = (P_2)$ in S_2/V, then $P_2 \in (Q_1, \ldots, Q_{t-1}, P_1) \setminus (Q_1, \ldots, Q_{t-1}) \cup (P_1)$, contradicting $P_1 \in \mathcal{B}$.

Let $P \in \mathcal{B}$. The sequence $Q_1, \ldots, Q_{t-1}, P$ is k-strong by definition of $\mathcal{B}$. Since $\mathcal{F}_2$ is a strong $(\epsilon, k) - \mathrm{SG}^*_t(2)$ configuration, there are $P_1, \ldots, P_s \in \mathcal{F}_2$ with $s \geq \epsilon m$ such that either $Q_1, \ldots, Q_{t-1}, P, P_i$ is not k-strong, or there is $R_i \in \mathcal{F}_2$ such that $R_i \in (Q_1, \ldots, Q_{t-1}, P, P_i) \setminus (Q_1, \ldots, Q_{t-1}, P) \cup (P_i)$.

Let $\mathcal{G} := \{P_i \mid Q_1, \ldots, Q_{t-1}, P, P_i \text{ is not a k-strong sequence}\}$. Let W be the r-wide vector space obtained by applying Proposition 44 to $V + \mathrm{span}_{\mathbb{C}}\{P\}$, we have $\dim W_1 \leq 7 \cdot r \cdot 3^t, \dim W_2 \leq t$. Every form in $\mathcal{G}$ is $k+4$-close to W. Hence, if $|\mathcal{G}| \geq \epsilon' m$ then we are done.

We are left with the case that $|\mathcal{G}| \leq \epsilon' m$. After relabelling, let $P_1, \ldots, P_{s'}$ be the forms that are in $\mathcal{B} \setminus \mathcal{G}$. Since $|\mathcal{B}| \geq (1-\epsilon')m$ and $|\mathcal{G}| \leq \epsilon' m$ we have $s' \geq (\epsilon - 2\epsilon')m$.

Now for each $i \leq s'$, there is a form $R_i \in \mathcal{F}_2 \cap (Q_1, \ldots, Q_{t-1}, P, P_i) \setminus (Q_1, \ldots, Q_{t-1}, P) \cup (P_i)$ say $R_i = \sum \alpha_j Q_j + \beta P + P_i$. Since $P_i \in \mathcal{B}$ we have $\beta \neq 0$. Suppose $P_1, \ldots, P_{s''}$ are such that

$$\mathcal{B} \cap ((Q_1, \ldots, Q_{t-1}, P, P_i) \setminus (Q_1, \ldots, Q_{t-1}, P)) = \{P_i\}. \tag{3}$$

If $R_i = \alpha R_j$ with $\alpha \neq 0$ for $i, j \leq s''$, then we have $\alpha P_j = \sum \alpha'_i Q_i + P_i + \beta' P$, contradicting Equation (3) for P_i. Therefore we have $s'' \leq |\mathcal{F}_2 \setminus \mathcal{B}| \leq \epsilon' m$. Hence, there are at least $\epsilon' m$ forms P_i such that $|\mathrm{span}_{\mathbb{C}}\{P, P_i\} \cap \mathcal{B}| \geq 3$ in S_2/V. Since this holds for every $P \in \mathcal{B}$, the set $\mathcal{B}$ is a $(\epsilon', 2)$-linear-SG configuration in S_2/V. By Theorem 31 we have that $\dim \mathrm{span}_{\mathbb{C}}\{\mathcal{B}\} \leq 4/\epsilon'$ in S_2/V and that $\dim \mathrm{span}_{\mathbb{C}}\{\mathcal{B}\} + V \leq t + 1 + 4/\epsilon'$. Applying Proposition 44 to $\mathrm{span}_{\mathbb{C}}\{\mathcal{B}\} + V$ gives us a r-wide vector space W with $\dim W_1 \leq 7 \cdot r \cdot 3^{t+1+4/\epsilon'}, \dim W_2 \leq t + 1 + 4/\epsilon'$ and $\mathcal{B} \subset W$. ◀

▶ **Lemma 72.** *Let $\mathcal{F}_2 \subset S_2$ finite, with $m := |\mathcal{F}_2|$. Suppose $\mathcal{F}_2$ is a* strong $(\epsilon, k) - \mathrm{SG}^*_1(2)$ *configuration. Then there is a r-wide vector space W with* $\dim W_1 \leq 7 \cdot r \cdot 3^{2+16/\epsilon}, \dim W_2 \leq 2 + 16/\epsilon$ *such that at least $\epsilon m/4$ forms in $\mathcal{F}_2$ are $k+5$ close to W.*

Proof. Let $\epsilon' := \epsilon/4$. Let $\mathcal{B}$ be the set of forms in $\mathcal{F}_2$ of rank at least $k+5$. If $|\mathcal{B}| \leq (1-\epsilon')m$, then there are at least $\epsilon' m$ forms that are $k+5$ close to the zero vector space and we are done with $W = 0$. We are left with the case when $|\mathcal{B}| \geq (1-\epsilon')m$.

Let $P \in \mathcal{B}$. Let $\mathcal{G} := \{P_i \mid P, P_i \text{ is not a k-strong sequence}\}$. Let W be the r-wide vector space obtained by applying Proposition 44 to $\mathrm{span}_{\mathbb{C}}\{P\}$, we have $\dim W_1 \leq 21 \cdot r, \dim W_2 \leq 1$. Every form in $\mathcal{G}$ is $k+4$ close to W. If therefore $|\mathcal{G}| \geq \epsilon' m$ then we are done. We are left with the case that $|\mathcal{G}| \leq \epsilon' m$.

Suppose $P_1, \ldots, P_{r'}$ are the forms in $\mathcal{B} \setminus \mathcal{G}$ such that P, P_i is a k-strong sequence and there exist $R_i \in (P, P_i) \setminus (P) \cup (P_i)$. We have $r' \geq 2\epsilon' m$. Suppose $P_1, \ldots, P_{r''}$ are such that $(P, P_i) \cap \mathcal{B} = \{P, P_i\}$. If $R_i = \beta R_j$ for $i, j \leq r''$ then $P_j \in \mathrm{span}_{\mathbb{C}}\{P, P_i\}$, contradicting choice of P_i. Therefore there are at least $\epsilon' m$ many forms P_i such that $|(P, P_i) \cap \mathcal{B}| \geq 3$. Since this holds for every P, we have that $\mathcal{B}$ is a $(\epsilon', 2)$-linear-SG, and by Theorem 31 we have that $\dim \mathrm{span}_{\mathbb{C}}\{\mathcal{B}\} \leq 4/\epsilon'$. If W is the r-wide space obtained by applying Proposition 44 to $\mathcal{B}$ then $\dim W_1 \leq 7 \cdot r \cdot 3^{4/\epsilon'}, \dim W_2 \leq 2 + 4/\epsilon'$ and $\mathcal{B} \subset W$, completing the proof. ◀

▶ **Corollary 73.** *Suppose $\mathcal{F} = \mathcal{F}_1 \sqcup \mathcal{F}_2$ be a $1 - \mathrm{SG}^*_k(2)$ configuration with $|\mathcal{F}_2| = m_2$. Then there exist a r-wide vector space W with* $\dim W_1 \leq 7 \cdot r \cdot 3^{k+1+16 \cdot 4^{k-1}}, \dim W_2 \leq k + 1 + 16 \cdot 4^{k-1}$ *such that at least $m_2/4^k$ forms in $\mathcal{F}_2$ are $k+5$ close to W.*

Proof. We first show that $\mathcal{F}_2$ is a strong $(1, k) - \mathrm{SG}^*_k(2)$ configuration. Suppose $Q_1, \ldots, Q_k$ is a k-strong sequence. Every subset of $Q_1, \ldots, Q_k$ is also a k-strong sequence, and hence generates a prime ideal by Remark 68. By definition $Q_1, \ldots, Q_k$ are linearly independent,

therefore $Q_1, \ldots, Q_k$ form a k-relavent set. For every $Q_{k+1} \in \mathcal{F}_2$, if $Q_1, \ldots, Q_{k+1}$ is a k-strong sequence, then $(Q_1, \ldots, Q_{k+1})$ is prime and $Q_{k+1} \notin \operatorname{rad}(Q_1, \ldots, Q_k)$. Therefore there exists $R \in \mathcal{F}$ such that $R \in (Q_1, \ldots, Q_{k+1}) \setminus (Q_1, \ldots, Q_k) \cup (Q_{k+1})$. Since $Q_i \in \mathcal{F}_2$ it must be that $R \in \mathcal{F}_2$. This shows that $\mathcal{F}_2$ is a strong $(1, k) - \mathrm{SG}^*_k(2)$ configuration.

Now let $t \geq 1$ be the smallest number such that $\mathcal{F}_2$ is a strong $\left(4^{k-t}, k\right) - \mathrm{SG}^*_t(2)$ configuration. By the previous paragraph, we have $t \leq k$. If $t = 1$, the required vector space exists by Lemma 72. If $t > 1$, we apply Lemma 71. Since $\mathcal{F}_2$ is not a strong $\left(4^{k-t-1}, k\right) - \mathrm{SG}^*_{t-1}(2)$ configuration, case 1 of Lemma 71 does not hold, Therefore there exists a vector space W with $\dim W_1 \leq 7 \cdot r \cdot 3^{t+1+16 \cdot 4^{k-t}}, \dim W_2 \leq t + 1 + 16 \cdot 4^{k-t}$ such that at least $4^{k-t-1} \cdot m_2$ forms in $\mathcal{F}_2$ are $k+4$ close to W. ◀

7 Proof of Sylvester-Gallai Theorem

In this section, we prove our main theorem: $1 - \mathrm{SG}^*_k(2)$ configurations have constant vector space dimension. Throughout this section we denote our $1 - \mathrm{SG}^*_k(2)$ configuration by $\mathcal{F} = \mathcal{F}_1 \sqcup \mathcal{F}_2$ where $\mathcal{F}_d$ is the set of forms of degree d in our configuration. Additionally, we define $m := |\mathcal{F}|$, $m_1 := |\mathcal{F}_1|$ and $m_2 := |\mathcal{F}_2|$.

Our proof has three main steps. In Section 7.1 we show that given $\mathcal{F}$, we can find a constant dimensional wide vector space W such that $\mathcal{F}$ is close to W. We call any such $\mathbb{C}$-algebra $\mathbb{C}[W]$ a *core algebra* of our configuration $\mathcal{F}$. This step uses the notion of strong sequences. In Section 7.2 we show that given such a vector space W, we can extend it to obtain a constant dimensional wide vector space $W \subset V$ such that $\mathcal{F}_2 \subset (V)$. This step uses the notion of integral sequences. In Section 7.3 we show that our main theorem follows given such a vector space V. This step uses general projections and the bound for linear SG configurations from [5, 12].

Define functions $\lambda_2(r, k) := k + 1 + 16 \cdot 4^{k-1}$, $\lambda_1(r, k) := 7 \cdot r \cdot 3^{\lambda_2(r,k)}$ and $B(k) := 3k + 15$. For the rest of this section, we set the parameter B in the definition of $\mathbb{L}_V(P)$ to $B(k)$. Note that while this parameter depends on k, it is independent of $|\mathcal{F}|$.

7.1 Constructing core algebras

We begin by showing that, to put all forms close to a wide algebra, it is enough to construct a small wide algebra which contains a constant fraction of the quadratics. More precisely, the next lemma allows us to increase the fraction of forms close to a given vector space without increasing the size of the vector space too much, so long as we start with a wide vector space which contains a constant fraction of the quadratics.[6]

Before we state and prove the lemma, the following notation will be very useful in this subsection: if $\gamma \in \mathbb{N}$, $\mathcal{G}$ is a set of forms and W is a graded vector space, we let

$$\mathcal{G}(\gamma, W) := \{P \in \mathcal{G} \mid P \text{ is } \gamma\text{-close to } W\}.$$

▶ **Lemma 74** (Increasing algebra intersection). *Let $0 < \delta \leq 1$, $r, \gamma, k \in \mathbb{N}$ be such that $r > 2\gamma \geq k + 5$ and W be a r-wide vector space. If $|\mathcal{F}(\gamma, W)| \geq \delta m$ then there is a r-wide vector space Y with $\dim Y_1 \leq 3^k \cdot (\dim W + r)$, $\dim Y_2 \leq \dim W_2 + k$ such that either $|\mathcal{F}(\gamma, Y)| \geq 3\delta m/2$, or $\mathcal{F} = \mathcal{F}(3\gamma, Y)$.*

[6] This is similar in spirit to [5, Proposition 7.11] and [16, Lemma 5.15].

Proof. Note that $\mathcal{F}_1 \subseteq \mathcal{F}(\gamma, W)$. Let $\mathcal{H} := \mathcal{F}_2 \setminus \mathcal{F}(3\gamma, W)$. In other words, $\mathcal{H}$ is the set of forms that are 3γ-far from W. Let $H_1, \ldots, H_t \in \mathcal{H}$ be the longest sequence of linearly independent forms such that

1. $\operatorname{minrank}(H_1, \ldots, H_t) \geq k + 5$, and
2. No nonzero form in $\operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_t\}$ is 2γ-close to W.

Suppose $t < k$. Let Y be the r-wide vector space obtained by applying Proposition 44 to $W + \operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_t\}$. Since $H_1, \ldots, H_t$ is the longest linearly independent sequence that satisfies the above conditions, for every other $H \in \mathcal{H}$, it must be that

- either $H \in \operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_t\}$, or
- $\operatorname{minrank}(H, H_1, \ldots, H_t) \leq k + 4$, or
- there exists $R \in \operatorname{span}_{\mathbb{C}}\{H, H_1, \ldots, H_t\} \setminus \operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_t\}$ such that R is 2γ-close to W.

In each of these three cases, it follows that $H \in \mathcal{F}(3\gamma, Y)$. Therefore in this case, Y is the required vector space.

We are now in the case where $t \geq k$. Consider the k elements $H_1, \ldots, H_k$. Note that $H_1, \ldots, H_k$ are linearly independent, and also satisfy $\operatorname{minrank}(H_1, \ldots, H_k) \geq k + 5$. Therefore, $H_1, \ldots, H_k$ is a k-strong sequence. By Remark 68, the ideal $(H_1, \ldots, H_k)$ is prime and k-relevant, and $S/(H_1, \ldots, H_k)$ is a UFD. Let Y be the r-wide vector space obtained by applying Proposition 44 to $W + \operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_k\}$, so $\dim Y_1 \leq 3^k \cdot (\dim W + r), \dim Y_2 \leq \dim W_2 + k$.

Now for each $G_i \in \mathcal{F}(\gamma, W)$ we have $G_i \notin (H_1, \ldots, H_k)$. In the graded UFD $S/(H_1, \ldots, H_k)$, the image of G_i must be irreducible: if not then $G_i = ab + \sum \alpha_j H_j$ in S, with $a, b \in S_1$, contradicting the fact that $\operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_k\}$ does not contain forms 2γ-close to W. The ideal $(H_1, \ldots, H_k, G_i)$ is therefore prime, and we have $R_i \in (H_1, \ldots, H_k, G_i) \setminus (H_1, \ldots, H_k)$ since $\mathcal{F}$ is a $1 - \mathrm{SG}^*_k(2)$ configuration. We have $R_i \in \mathcal{F}(\gamma, Y)$.

If $R_i \in \mathcal{F}_1$ then we must have $R_i \in (G_i)$, contradicting the pairwise independence of elements of $\mathcal{F}_2$, therefore $R_i \in \mathcal{F}_2$. After scaling we have either $R_i - G_i \in \operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_k\}$ (if $G_i \in \mathcal{F}_2$) or $R_i - aG_i \in \operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_k\}$ (if $G_i \in \mathcal{F}_1$). Therefore $R_i \notin \mathcal{F}(\gamma, W)$ since otherwise $\operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_k\}$ contains a form 2γ-close to W. If G_j is another form such that $R_i = R_j$, then $R_i - G_j$ or $R_i - bG_j$ is in $\operatorname{span}_{\mathbb{C}}\{H_1, \ldots, H_t\}$, and it must be that $G_i, G_j \in \mathcal{F}_1$ and $aG_i = bG_j$ so $G_j \in (a), G_i \in (b)$. This shows that $|\{R_i\}_i| \geq \delta m/2$. Since $\mathcal{F}(\gamma, W) \cup \{R_i\}_i \subseteq \mathcal{F}(\gamma, Y)$, we are done. ◀

We are now ready to prove the main lemma of this subsection.

▶ **Lemma 75** (Constructing core algebras). *Suppose $\mathcal{F}$ is a $1-\mathrm{SG}^*_k(2)$ configuration. For any r there exists a r-wide vector space W with $\dim W_1 \leq 2 \cdot 3^{k^2} \cdot \lambda_1(r, k)$ and $\dim W_2 \leq 4k^2 + \lambda_2(r, k)$ such that $\mathcal{F} = \mathcal{F}(B(k), W)$.*

Proof of Lemma 75. We build a sequence of vector spaces $W^{(i)}$ such that either $\mathcal{F} = \mathcal{F}(B(k), W^{(i)})$ or $\left|\mathcal{F}(k+5, W^{(i)})\right| \geq (3/2)^i \cdot m/4^k$.

Set $W^{(0)}$ to be the r-wide vector space obtained by Corollary 73. By Corollary 73, at least $m_2/4^k$ forms in $\mathcal{F}_2$ are $k+5$ close to $W^{(0)}$. Further, every form in $\mathcal{F}_1$ is 1-close to $W^{(0)}$. Since $m_1 + m_2/4^k \geq m/4^k$, we have $\left|\mathcal{F}(k+5, W^{(0)}\right| \geq m/4^k$. Therefore, $W^{(0)}$ satisfies the above property. We have $\dim W^{(0)}_i \leq \lambda_i(r, k)$.

Given $W^{(i)}$, if $\mathcal{F} = \mathcal{F}(B(k), W^{(i)})$ then terminate. If not, then apply Lemma 74 to $W^{(i)}$ with $\gamma = k + 5$ and $\delta = (3/2)^i \cdot 1/4^k$ to obtain $W^{(i+1)}$. By Lemma 74, either $\mathcal{F} = \mathcal{F}(B(k), W^{(i+1)} = \mathcal{F}$ or $\left|\mathcal{F}(k+5, W^{(i+1)})\right| \geq (3/2)^{i+1} \cdot m/4^k$. Therefore $W^{(i+1)}$ also has the required property.

The above process must terminate when $(3/2)^i \cdot 1/4^k \geq 1$, which holds when $i > 4k$. Further, by induction we have $\dim W_1^{(4k)} \leq 3^{k^2}\lambda_1(r,k) + 3^{k^2} \cdot 2^k \cdot r \leq 2 \cdot 3^{k^2} \cdot \lambda_1(r,k)$ and $\dim W_2 \leq 4k^2 + \lambda_2(r,k)$. Therefore $W^{(4k)}$ is the required space. ◀

7.2 Finding small ideal containing the quadratic forms

In this section we show that all quadratics in any $1 - \mathrm{SG}_k^*(2)$ must be contained in an ideal generated by a small number of forms. The main idea is that given any wide vector space, there exist short maximal integral sequences with respect to the vector space. Recall that the we set the parameter B in the definition of relative linear spaces to $B(k) := 3k + 15$.

▶ **Lemma 76.** *Suppose $r \geq 4(k+2)B(k)+1$. Suppose $\mathcal{F}$ is a $1 - \mathrm{SG}_k^*(2)$ configuration, and suppose W is a r-wide vector space such that every $F \in \mathcal{F}$ is $B(k)$-close to W. Then there exists a maximal integral sequence with respect to inclusion of length at most k with respect to W.*

Proof. For each $F \in \mathcal{F}$ let $\overline{F_W^L}$ be the image of F_W^L in $S/(W_1)$. Define potential function Φ on integral sequences as

$$\Phi(G_1, \ldots, G_c) := \sum_{i=1}^{c} \dim \mathrm{Lin}\left(\overline{(G_i)_W^L}\right).$$

If $\mathcal{F} \subset (W)$ then there are no integral sequences with respect to W, and the statement holds vacuously, therefore we can assume that $\mathcal{F} \setminus (W) \neq \emptyset$. Combined with the fact that W is r-wide, and that every form in $\mathcal{F}$ is $B(k)$-close to W, there exists nonempty integral sequences with respect to W. Among all integral sequences of length at most $k+1$, pick $F_1, \ldots, F_c$ such that the above potential function is maximised. If $c \leq k$, then $F_1, \ldots, F_c$ is maximal: if not, and if $F_1, \ldots, F_{c+1}$ is an integral sequence that extends $F_1, \ldots, F_c$ then $\Phi(F_1, \ldots, F_{c+1}) > \Phi(F_1, \ldots, F_c)$, contradicting maximality.

We are left with the case where $c = k+1$. We will find an integral sequence of length at most k with the same potential function value, and therefore the new integral sequence will be maximal. The sequence $F_1, \ldots, F_k$ is an integral sequence, therefore by Lemma 64 we have that $(F_1, \ldots, F_k)$ is a radical ideal. Further, by Corollary 65 we have that $F_1, \ldots, F_k$ is a k-relevant set. Similarly, $F_1, \ldots, F_{k+1}$ is a $k+1$-relevant set, therefore $F_{k+1} \notin (F_1, \ldots, F_k)$.

Since $\mathcal{F}$ is a $1-\mathrm{SG}_2^*(2)$, we have $R \in (F_1, \ldots, F_{k+1}) \setminus (F_1, \ldots, F_k)$, that is, $R = \sum_{j=1}^{k+1} \alpha_j F_j$ with $\alpha_{k+1} \neq 0$. Without loss of generality, suppose $\alpha_j = 0$ for $j = 1, \ldots, b$ and $\alpha_j \neq 0$ for $j = b+1, \ldots, k+1$. Since the polynomials in $\mathcal{F}$ are pairwise linearly independent we have $b < k$.

Now $R = R_W^H + R_W^L = \sum_{j>b} \alpha_i F_i = \sum_{j>b} \alpha_i((F_i)_W^H + (F_i)_W^L)$. Since the space W is r-wide, we have $R_W^H = \sum_{j>b} \alpha_i (F_i)_W^H$ and $R_W^L = \sum_{j>b} \alpha_i (F_i)_W^L$. By Corollary 62, after a change of basis we can assume that there are disjoint sets of variables $Y, Y_1, \ldots, Y_k$ such that W_1 is spanned by Y and $(F_i)_W^L \in \mathbb{C}[Y, Y_i]$. We have $\mathrm{Lin}\left(R_W^L\right) \subseteq \mathbb{C}[Y, Y_{b+1}, \ldots, Y_{k+1}]$, whence $F_1, \ldots, F_b, R$ is an integral sequence by Proposition 60. Further $\overline{R_W^L} = \sum_{j>b} \alpha_j \overline{(F_j)_W^L}$, and since $\overline{(F_i)_W^L} \in \mathbb{C}[Y_i]$, by Lemma 18 we can deduce that $\dim \mathrm{Lin}\left(\overline{R_W^L}\right) = \sum_{j>b} \dim \mathrm{Lin}\left(\overline{(F_j)_W^L}\right)$. Therefore $F_1, \ldots, F_b, R$ is an integral sequence of length at most k with $\Phi(F_1, \ldots, F_b, R) = \Phi(F_1, \ldots, F_{k+1})$. This proves that $F_1, \ldots, F_b, R$ is a maximal integral sequence. ◀

▶ **Lemma 77.** *Suppose $\mathcal{F}$ is a $1 - \mathrm{SG}_k^*(2)$ configuration. Suppose $r \geq 8(k+2)B(k)^2 + 1$. There exists a $\left(r - 4kB(k)^2\right)$-wide vector space W with $\dim W_1 \leq 3 \cdot 3^{k^2} \cdot \lambda_1(r,k)$ and $\dim W_2 \leq 4k^2 + \lambda_2(r,k)$ such that $\mathcal{F}_2 \subset (W)$.*

Proof. For any r-wide vector space U such that every polynomial $F \in \mathcal{F}$ is $B(k)$ close to U, define potential function Ψ as

$$\Psi(U) = \max_{F \in \mathcal{F}_2 \setminus (U)} \dim \overline{\mathbb{L}}_U(F).$$

If $\Psi(U) = 0$ for some such U, then $\mathcal{F}_2 \subset (U)$.

We now construct W iteratively. Let $W^{(0)}$ be the r-wide vector space whose existence is guaranteed by Lemma 75. Since every $F \in \mathcal{F}$ is $B(k)$-close to $W^{(0)}$ we have $\Psi(W^{(0)}) \leq 2B(k)$. The vector space $W^{(0)}$ is r-wide, and $r \geq 4(k+2)B(k)+1$, therefore by Lemma 76 we can find a maximal integral sequence $F_1, \ldots, F_c$ with respect to $W^{(0)}$ with $c \leq k$. Set $W^{(1)} = \sum_{i=1}^{c} \mathbb{L}_{W^{(0)}}(F_i) + W_2^{(0)}$. That $F_1, \ldots, F_c$ is maximal implies the following: for every $G \in \mathcal{F}_2 \setminus \left(W^{(0)}\right)$ we have $\dim \overline{\mathbb{L}}_{W^{(0)}}(G) > \dim \overline{\mathbb{L}}_{W^{(1)}}(G)$. Therefore $\Psi(W^{(1)}) < \Psi(W^{(0)})$. By Lemma 58, the vector space $W^{(1)}$ is $r - 2kB(k)$-wide since $W_1^{(1)}$ is obtained by adding at most $2kB(k)$ linear forms to $W_1^{(0)}$. In general, given $W^{(i)}$ we use Lemma 76 to find a maximal integral sequence $F_{i1}, \ldots, F_{ic}$, and set $W^{(i+1)} := \sum_{j=1}^{c} \mathbb{L}_{W^{(i)}}(F_{ij}) + W_2^{(i)}$. Maximality of the sequence implies $\Psi(W^{(i+1)}) < \Psi(W^{(i)})$. By the bound on r, at every step $W^{(i)}$ is at least $4(k+2)B(k)+1$-wide. After at most $2B(k)$ steps we find a t such that $\Psi(W^{(t)}) = 0$.

By Lemma 75 we have $\dim W_1^{(0)} \leq 2 \cdot 3^{k^2} \lambda_1(r,k)$. Since $W^{(i+1)}$ is obtained by adding $2B(k)k$ linear forms to $W^{(i)}$ we get $\dim W_1^{(t)} \leq 2 \cdot 3^{k^2} \lambda_1(r,k) + 4B(k)^2 k \leq 3 \cdot 3^{k^2} \lambda_1(r,k)$. Further we have $\dim W_2^{(i)} = \dim W_2^{(i-1)}$ for all i, therefore $\dim W_2^{(t)} = \dim W_0^{(t)} \leq 4k^2 + \lambda_2(r,k)$. This completes the proof. ◀

7.3 Basic configuration

In this section we prove Theorem 37 for the special case where all the quadratics are in the ideal generated by an r-wide algebra.

▶ **Lemma 78.** *Suppose $\mathcal{F}$ is a $1 - \mathrm{SG}_k^*(2)$ configuration. Suppose there is an r-wide linear subspace W with $r \geq k+5$ such that $\mathcal{F}_2 \subset (W)$. Then there is linear subspace W_1' with $\dim(W_1') = \binom{C'^k}{} \cdot \dim W_1$, such that $\mathcal{F} \subseteq W_2 + \mathbb{C}[W_1']$.*

Proof. Let $\varphi := \varphi_{\alpha, W_1}$ be a projection mapping as defined in Definition 20. By Remark 49, the space $\varphi(W)$ is a $r-1$-wide vector space. Let $\Delta := \dim W_1$. As $\mathcal{F}_2 \subseteq (W)$, every $F \in \mathcal{F}_2$ satisfies $\varphi(F) = \alpha \varphi(F_W^H) + z \cdot \ell$ for some linear form $\ell \in S[z]_1$.

Let $\mathcal{L}$ be the union of all the linear forms that occur in the above way, and all the linear forms in $\mathcal{F}$. Formally, $\mathcal{L} := \left\{\ell \mid \varphi(F) = \alpha \varphi(F_W^H) + z \cdot \ell, F \in \mathcal{F}_2\right\} \cup \varphi(\mathcal{F}_1)$. Let $\mathcal{L}/(z)$ denote the image of $\mathcal{L}$ in the vector space $(S[z]/(z))_1$, that is, the linear forms modulo z. We show that $\mathcal{L}/(z)$ is a $1 - SG_k(1)$ configuration.

Let $\overline{\ell_1}, \ldots, \overline{\ell_k} \in \mathcal{L}/(z)$ be independent. Let $\overline{\ell_{k+1}} \in \mathcal{L}/(z)$. We need to show that one of the following cases holds:

1. $\bar{\ell}_{k+1} \in \mathrm{span}_{\mathbb{C}} \left\{\overline{\ell_1}, \ldots, \overline{\ell_k}\right\}$.
2. there is $\bar{g} \in \mathrm{span}_{\mathbb{C}} \left\{\overline{\ell_1}, \ldots, \overline{\ell_{k+1}}\right\} \setminus \left\{\overline{\ell_1}, \ldots, \overline{\ell_k}\right\}$ with $\bar{g} \in \mathcal{L}/(z)$.

Consider the corresponding $F_1, \ldots, F_{k+1} \in \mathcal{F}$ such that $\varphi(F_i) = \alpha_i \pi(F_{iW}^H) + z \cdot \ell_i$, with $\ell_i/(z) = \bar{\ell}_i$, or, if $F_i \in \mathcal{F}_1$ then $F_i = \ell_i$.

The first step is to show that $F_1, \ldots, F_k$ form a k-relevant set. Without loss of generality, assume that $F_1 \in \mathrm{rad}(F_2, \ldots, F_k)$. We have $\varphi(F_1) \in \mathrm{rad}(\varphi(F_2), \ldots, \varphi(F_k))$, and by Lemma 50 we have $z\ell_1 \in (\ell_2, \ldots, \ell_k)$. Since the ideal $\ell_2, \ldots, \ell_k$ is prime, and since $\overline{\ell_2}, \ldots, \overline{\ell_k}$ are independent, we get $\ell_1 \in \mathrm{span}_{\mathbb{C}} \{\ell_2, \ldots, \ell_k\}$ contradicting choice of $\ell_1, \ldots, \ell_k$. Therefore $F_1, \ldots, F_k$ is k-relevant.

By the same argument, if $F_{k+1} \in \operatorname{rad}(F_1, \ldots, F_k)$ then $\overline{\ell_{k+1}} \in \operatorname{span}_{\mathbb{C}} \{\overline{\ell_1}, \ldots, \overline{\ell_k}\}$. We are left with the case when $F_{k+1} \notin \operatorname{rad}(F_1, \ldots, F_k)$. Since $\mathcal{F}$ is a $1 - \mathrm{SG}_2^*(2)$ configuration, there exists $R \in \operatorname{rad}(F_1, \ldots, F_{k+1}) \setminus \operatorname{rad}(F_1, \ldots, F_k)$. Let $\overline{g}$ be such that $\varphi(R) = \alpha_i \varphi(R_W^H) + z \cdot g$ if $R \in \mathcal{F}_2$, and $R = g$ otherwise. We have $\varphi(R) \in \operatorname{rad}(\varphi(F_1), \ldots, \varphi(F_{k+1}))$. By Lemma 50, we have $z\overline{g} \in \operatorname{span}_{\mathbb{C}} \{\overline{\ell_1}, \ldots, \overline{\ell_{k+1}}\}$ which implies $\overline{g} \in \operatorname{span}_{\mathbb{C}} \{\overline{\ell_1}, \ldots, \overline{\ell_{k+1}}\}$. Finally, by Lemma 51, we have that $\overline{g} \notin (\overline{\ell_i})$ for any i. This completes the proof that $\mathcal{L}/(z)$ is a $1 - SG_k(1)$ configuration.

By Theorem 30 we have

$$\dim(\mathbb{L}_W(\varphi(\mathcal{F})) = \dim(\mathcal{L}/(z)) + 1 \leq C'^k,$$

for some universal constant C'. Applying Proposition 24 it follows that $\dim(\mathbb{L}_W(\mathcal{F})) \leq C'^k \cdot \Delta$. In particular, it follows that there is a linear space of linear forms W_1', with $\dim(W_1') \leq C'^k \cdot \Delta$, satisfying $\mathcal{F} \subseteq W_2 + \mathbb{C}[W_1']$, completing the proof. ◀

7.4 Proof of main theorem

We now prove the main theorem, which we restate for convenience.

▶ **Theorem 37** (Radical SG Theorem for tuples of quadratics)**.** *Let $\mathcal{F}$ be a $1 - \mathrm{SG}_k^*(2)$ configuration. There is a universal constant $c > 0$ such that $\dim(\operatorname{span}_{\mathbb{C}} \{\mathcal{F}\}) \leq 3^{c \cdot 4^k}$.*

Proof. Let $r := 8(k+2)B(k)^2 + k + 6$. By Lemma 77, there exists a $k+5$-wide vector space W with $\dim W_1 \leq 3 \cdot 3^{k^2} \cdot \lambda_1(r,k)$ and $\dim W_2 \leq 4k^2 + \lambda_2(r,k)$ such that $\mathcal{F}_2 \subseteq (W)$. Applying Lemma 78 with this W, we obtain a vector space $W_1' \subseteq S_1$ with $\dim W_1' \leq 3 \cdot 3^{k^2} \cdot \lambda_1(r,k) \cdot C'^k$ such that $\mathcal{F} \subseteq W_2 + \mathbb{C}[W_1']$. If $Y \subseteq S_2$ is the space spanned by pairwise products of forms in W_1', then $\mathcal{F} \subseteq W_2 + Y$ and $\dim Y \leq 9 \cdot 9^{k^2} \cdot \lambda_1^2(r,k) \cdot C'^{2k}$. Substituting for λ_1, λ_2 gives us the required result. ◀

▶ Remark 79. Suppose the set $\mathcal{F}$ does not have any k-relevant sequences. In this case, $\mathcal{F}$ is vacuously an $1 - \mathrm{SG}_k^*(2)$ configuration. There are no k-strong sequences in $\mathcal{F}$ of length k, since any such sequence is a k-relevant set. Therefore every form in such a configuration is $k+5$-close to a r-wide vector space W of dimension $7 \cdot r \cdot 3^k$ by Lemma 69. Further, such a configuration has no integral sequences of length $k+1$. Therefore, by the arguments in Lemma 76 and Lemma 77, by adding $4kB(k)^2$ linear forms to W, we get a wide vector space Y such that $\mathcal{F} \subseteq Y$. If we project to Y_1 and pick out the linear forms corresponding to each element of $\mathcal{F}$ as in Lemma 78, then there are no set of $k+1$ linearly independent forms by Lemma 50. Therefore, we can deduce by the properties of the projection map that $\dim \operatorname{span}_{\mathbb{C}} \{\mathcal{F}\} = 2^{O(k)}$ in this case.

8 Conclusion

In this work, we prove a higher codimension analogue of the quadratic Sylvester–Gallai theorem, generalising the results of [36, 20]. Our ability to handle ideals of higher codimension shows our approach is a promising one towards a full derandomisation of PIT for $\Sigma^k\Pi\Sigma\Pi^2$ circuits.

To prove our main theorem, we build upon the results of [2, 28] and use the wide algebras developed in these works to control the cancellations in SG configurations. One key difference between this work and previous works [36, 29, 30, 31, 16, 28] is that we prove our Sylvester-Gallai theorem without a fine classification of the ideals we deal with.

Our work leaves several open questions which are of interest to combinatorialists, algebraic geometers, and complexity theorists. On the combinatorial and geometric side, understanding the different generalizations of Sylvester's problems to higher codimension (such as the elementary SG configurations defined in [20] and also studied in [5]) is a problem of independent interest, as well as the generalization to higher codimension of the "product" version of Sylvester's question, defined in [19, 29]. And of course, fully derandomizing PIT for $\Sigma^k\Pi\Sigma\Pi^2$ is still a major open question.

References

1 Manindra Agrawal, Chandan Saha, Ramprasad Saptharishi, and Nitin Saxena. Jacobian hits circuits: Hitting sets, lower bounds for depth-d occur-k formulas and depth-3 transcendence degree-k circuits. *SIAM Journal on Computing*, 45(4):1533–1562, 2016. doi:10.1137/130910725.

2 Tigran Ananyan and Melvin Hochster. Strength conditions, small subalgebras, and stillman bounds in degree ≤ 4. *Transactions of the American Mathematical Society*, 373(7):4757–4806, 2020.

3 M. F. Atiyah and I. G. MacDonald. *Introduction to Commutative Algebra*. Addison Wesley Publishing Company, 1969.

4 Boaz Barak, Zeev Dvir, Avi Wigderson, and Amir Yehudayoff. Fractional sylvester-gallai theorems. *Proceedings of the National Academy of Sciences*, 110, 2013.

5 Boaz Barak, Zeev Dvir, Amir Yehudayoff, and Avi Wigderson. Rank bounds for design matrices with applications to combinatorial geometry and locally correctable codes. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 519–528, 2011. doi:10.1145/1993636.1993705.

6 David Bayer and Michael Stillman. On the complexity of computing syzygies. *Journal of Symbolic Computation*, 6(2):135–147, 1988. doi:10.1016/S0747-7171(88)80039-7.

7 Peter Borwein and William OJ Moser. A survey of sylvester's problem and its generalizations. *Aequationes Mathematicae*, 40(1):111–135, 1990.

8 Chi-Ning Chou, Mrinal Kumar, and Noam Solomon. Hardness vs randomness for bounded depth arithmetic circuits. In *33rd Computational Complexity Conference (CCC 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

9 Leonard Eugene Dickson. The points of inflexion of a plane cubic curve. *The Annals of Mathematics*, 16(1/4):50–66, 1914.

10 Pranjal Dutta, Prateek Dwivedi, and Nitin Saxena. Deterministic identity testing paradigms for bounded top-fanin depth-4 circuits. In *Proceedings of the 36th Computational Complexity Conference*, CCC '21, 2021. doi:10.4230/LIPIcs.CCC.2021.11.

11 Zeev Dvir, Ankit Garg, Rafael Oliveira, and József Solymosi. Rank bounds for design matrices with block entries and geometric applications. *Discrete Analysis*, 2018.

12 Zeev Dvir, Shubhangi Saraf, and Avi Wigderson. Improved rank bounds for design matrices and a new proof of kelly's theorem. In *Forum of Mathematics, Sigma*, volume 2. Cambridge University Press, 2014.

13 Michael Edelstein and Leroy M Kelly. Bisecants of finite collections of sets in linear spaces. *Canadian Journal of Mathematics*, 18:375–380, 1966.

14 David Eisenbud. *Commutative Algebra with a View Toward Algebraic Theory*. Springer-Verlag, New York, 1995.

15 Tibor Gallai. Solution of problem 4065. *American Mathematical Monthly*, 51:169–171, 1944.

16 Abhibhav Garg, Rafael Oliveira, and Akash Kumar Sengupta. Robust Radical Sylvester-Gallai Theorem for Quadratics. In *38th International Symposium on Computational Geometry (SoCG 2022)*, volume 224 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 42:1–42:13, 2022. doi:10.4230/LIPIcs.SoCG.2022.42.

17 Siyao Guo, Tal Malkin, Igor C Oliveira, and Alon Rosen. The power of negations in cryptography. In *Theory of Cryptography Conference*, pages 36–65. Springer, 2015.

18 Zeyu Guo. Variety Evasive Subspace Families. In Valentine Kabanets, editor, *36th Computational Complexity Conference (CCC 2021)*, volume 200 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 20:1–20:33, 2021. doi:10.4230/LIPIcs.CCC.2021.20.

19 Ankit Gupta. Algebraic geometric techniques for depth-4 pit & sylvester-gallai conjectures for varieties. In *Electron. Colloquium Comput. Complex.*, volume 21, page 130, 2014.

20 Sten Hansen. A generalization of a theorem of sylvester on the lines determined by a finite point set. *Mathematica Scandinavica*, 16(2):175–180, 1965.

21 Friedrich Hirzebruch. Arrangements of lines and algebraic surfaces. In *Arithmetic and geometry*, pages 113–140. Springer, 1983.

22 Pavel Hrubeš and Amir Yehudayoff. Monotone separations for constant degree polynomials. *Information Processing Letters*, 110(1):1–3, 2009.

23 Neeraj Kayal and Shubhangi Saraf. Blackbox polynomial identity testing for depth 3 circuits. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 198–207. IEEE, 2009.

24 Leroy Milton Kelly. A resolution of the sylvester-gallai problem of j.-p. serre. *Discrete & Computational Geometry*, 1(2):101–104, 1986.

25 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 804–814. IEEE, 2022.

26 Ernst W Mayr and Albert R Meyer. The complexity of the word problems for commutative semigroups and polynomial ideals. *Advances in mathematics*, 46(3):305–329, 1982.

27 Eberhard Melchior. Uber vielseite der projektiven ebene. *Deutsche Math*, 5:461–475, 1940.

28 Rafael Oliveira and Akash Sengupta. Radical sylvester-gallai theorem for cubics. *FOCS*, 2022.

29 Shir Peleg and Amir Shpilka. A Generalized Sylvester-Gallai Type Theorem for Quadratic Polynomials. In *35th Computational Complexity Conference (CCC 2020)*, pages 8:1–8:33, 2020. doi:10.4230/LIPIcs.CCC.2020.8.

30 Shir Peleg and Amir Shpilka. Polynomial time deterministic identity testing algorithm for $\Sigma^{[3]}\Pi\Sigma\Pi^{[2]}$ circuits via edelstein-kelly type theorem for quadratic polynomials. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 259–271. ACM, 2021. doi:10.1145/3406325.3451013.

31 Shir Peleg and Amir Shpilka. Robust sylvester-gallai type theorem for quadratic polynomials. In *38th International Symposium on Computational Geometry, SoCG 2022*, volume 224 of *LIPIcs*, pages 43:1–43:15, 2022. doi:10.4230/LIPIcs.SoCG.2022.43.

32 Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *Journal of the ACM (JACM)*, 39(3):736–744, 1992.

33 Alexander A Razborov. On submodular complexity measures. *Boolean Function Complexity,(M. Paterson, Ed.)*, pages 76–83, 1992.

34 Nitin Saxena and Comandur Seshadhri. From sylvester-gallai configurations to rank bounds: Improved blackbox identity test for depth-3 circuits. *Journal of the ACM (JACM)*, 60(5):1–33, 2013.

35 Jean-Pierre Serre. Advanced problem 5359. *Amer. Math. Monthly*, 73(1):89, 1966.

36 Amir Shpilka. Sylvester-gallai type theorems for quadratic polynomials. *Discrete Analysis*, page 14492, 2020.

37 Gaurav Sinha. Reconstruction of real depth-3 circuits with top fan-in 2. In *31st Conference on Computational Complexity (CCC 2016)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2016.

38 The Stacks Project Authors. *Stacks Project*. Open Source, 2015. URL: http://stacks.math.columbia.edu.

39 James Joseph Sylvester. Mathematical question 11851. *Educational Times*, 59(98):256, 1893.

40 Endre Szemerédi and William T. Trotter. Extremal problems in discrete geometry. *Combinatorica*, 3(3):381–392, 1983.

41 Éva Tardos. The gap between monotone and non-monotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988.

42 Leslie G Valiant. Negation can be exponentially powerful. In *Proceedings of the eleventh annual ACM symposium on theory of computing*, pages 189–196, 1979.

A Alternative proof of Lemma 64

We give here an alternative proof suggested by an anonymous reviewer (with proper formalizations).

Alternative Proof of Lemma 64. Let $I := (F_1, \ldots, F_t)$. Since the inclusion $\mathcal{A} := \mathbb{C}[V, Y_1, \ldots, Y_t] \to S$ is a free extension, it is enough to prove that I is radical in $\mathcal{A}$. Moreover, since $\mathcal{A}$ is isomorphic to a polynomial ring, by Corollary 62 we can assume that our polynomial ring is $\mathcal{A} := \mathbb{C}[Z, y_1, y_2, \ldots, y_t]$ where $\mathcal{F}_i \in \mathbb{C}[Z, y_i]$. By Lemma 63, we know that $F \in \mathbb{C}[Z] \setminus \{0\}$ is regular with $F_1, \ldots, F_t$, and hence it is not in any minimal prime of I. Thus, F is not a zero divisor over $\mathcal{A}/I$.

Since $\mathcal{B} := \mathbb{C}(Z)[y_1, \ldots, y_t]$ is the localization of $\mathcal{A}$ over $\mathbb{C}[Z] \setminus \{0\}$, by the above, we have that I is radical in $\mathcal{A}$ iff $I \cdot \mathcal{B}$ is radical in $\mathcal{B}$. Let $R := \overline{\mathbb{C}(Z)}[y_1, \ldots, y_t]$. It is easy to see that $I \cdot \mathcal{B}$ is radical in $\mathcal{B}$ if $I \cdot R$ is radical over R. To see that $I \cdot R$ is a radical ideal, note that $F_i \in \mathbb{C}[Z, y_i]$ irreducible implies that $\mathrm{disc}_{y_i}(F_i) \in \mathbb{C}[Z] \setminus \{0\}$ and hence $F_i = (y_i - \alpha_i)(y_i - \beta_i)$ over R, with $\alpha_i \neq \beta_i$. Thus, $I \cdot R$ is the intersection of maximal ideals and therefore radical. ◀

Reducing Tarski to Unique Tarski (In the Black-Box Model)

Xi Chen
Columbia University, New York, NY, USA

Yuhao Li
Columbia University, New York, NY, USA

Mihalis Yannakakis
Columbia University, New York, NY, USA

Abstract

We study the problem of finding a Tarski fixed point over the k-dimensional grid $[n]^k$. We give a black-box reduction from the Tarski problem to the same problem with an additional promise that the input function has a unique fixed point. It implies that the Tarski problem and the unique Tarski problem have exactly the same query complexity. Our reduction is based on a novel notion of partial-information functions which we use to fool algorithms for the unique Tarski problem as if they were working on a monotone function with a unique fixed point.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness; Theory of computation → Exact and approximate computation of equilibria

Keywords and phrases Tarski fixed point, Query complexity, TFNP

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.21

Related Version *ECCC Report*: https://eccc.weizmann.ac.il/report/2023/073/

Funding *Xi Chen*: Supported by NSF grants IIS-1838154, CCF-2106429 and CCF-2107187.
Yuhao Li: Supported by NSF grants CCF-1563155, CCF-1703925, IIS-1838154, CCF-2106429 and CCF-2107187.
Mihalis Yannakakis: Supported by NSF grants CCF-2107187 and CCF-2212233.

Acknowledgements We would like to thank anonymous CCC reviewers for their helpful comments to improve the presentation of the paper.

1 Introduction

We start with the definition of *monotone* functions and state Tarski's fixed point theorem [12]:

▶ **Definition 1** (Monotone functions). *Let $(\mathcal{L}, \preceq)$ be a complete lattice. A function $f : \mathcal{L} \to \mathcal{L}$ is said to be monotone if $f(a) \preceq f(b)$ for all $a, b \in \mathcal{L}$ with $a \preceq b$.*

▶ **Theorem 2** (Tarski). *For any complete lattice $(\mathcal{L}, \preceq)$ and any monotone function $f : \mathcal{L} \to \mathcal{L}$, there must be a point $x \in \mathcal{L}$ such that $f(x) = x$, i.e., x is a fixed point. In fact, the fixed points form a sublattice, with a greatest and a smallest element.*

Tarski's fixed point theorem has extensive applications in many fields, including for example verification, semantics, game theory and economics. For example in game theory there is an important class of games, called supermodular games (or games with strategic complementarities) which model economic settings where a player's best response is a monotone function (or correspondence) of the other players' actions [13, 14, 10]. These games always have pure equilibria (in fact a lattice of pure equilibria) by Tarski's theorem. Computing a pure equilibrium in such a game corresponds to finding a Tarski fixed point. In fact, as shown in [4], finding a pure equilibrium in supermodular games is essentially equivalent to finding a fixed point of monotone functions.

COMPUTATIONAL COMPLEXITY CONFERENCE

There are several other types of games that reduce to the Tarski problem. For example, Condon's simple stochastic games [2] have been intensely studied both in theoretical computer science as well as in the verification field (and they subsume other well-studied problems, such as parity games); their complexity remains a notorious open problem. The problem can be reduced to the Tarski problem of finding a fixed point of a given monotone function f, and in fact in this case we can even guarantee that the function has a unique fixed point. A similar property holds for the broader class of stochastic games defined originally by Shapley [11], and studied extensively since then. These games have in general irrational solutions, but it can be shown again that approximating the solution to any desired accuracy reduces to the problem of computing a fixed point of a monotone function that is furthermore guaranteed to have a unique fixed point (see [4] for more details). More generally, uniqueness of solutions is a desirable property in many applications in game theory, economics, and other fields. For sufficient conditions that ensure the uniqueness of Tarski fixed points, see [9] and references therein.

Thus, these facts raise the question, how hard is it to find a fixed point of a given monotone function? And if we know that the function has a unique fixed point, does this make the problem easier?

In this paper, we study the deterministic query complexity of finding a fixed point of a monotone function f over the complete lattice of a k-dimensional grid $([n]^k, \preceq)$, where $[n]$ denotes $\{1, \ldots, n\}$ and $\preceq$ denotes the natural partial order over $\mathbb{Z}^k$: $a \preceq b$ if and only if $a_i \leq b_i$ for every $i \in [k]$. So a monotone function $f : [n]^k \to [n]^k$ satisfies $f(a) \preceq f(b)$ for all $a, b \in [n]^k$ with $a \preceq b$, and we write $\texttt{Fix}(f)$ to denote the set of fixed points x of f satisfying $f(x) = x$. In the applications, n is typically exponential in the input size and k is polynomial. Thus, polynomial complexity in this context means polynomial in $\log n$ and k. Under the query model, an algorithm has oracle access to an unknown monotone function $f : [n]^k \to [n]^k$. In each round, it can send a query $x \in [n]^k$ to the oracle to reveal $f(x)$, and it succeeds after making a query x that returns $f(x) = x$. We write $\textsc{Tarski}(n, k)$ to denote this problem.

Our understanding of the query complexity of $\textsc{Tarski}(n, k)$ remains rather limited. On the upper bound side, there are two basic algorithms. Tarski's algorithm (or Kleene iteration in a different literature) starts from the bottom element 1^k of the lattice (or the top element n^k) and applies repeatedly f until it reaches a fixed point; the query complexity is $\Theta(nk)$ in the worst case. Another algorithm by [3] applies a binary search strategy in a recursive way and has query complexity $O(\log^k n)$. More recently, [7] gave an algorithm for $\textsc{Tarski}(n, k)$ with $O(\log^{\lceil 2k/3 \rceil} n)$ queries, which was further improved to $O(\log^{\lceil (k+1)/2 \rceil} n)$ in [1]. Both algorithms of [7] and [1] are based on decomposition theorems that lead to more efficient recursive schemes for Tarski fixed points.

On the lower bound side, [4] showed that $\textsc{Tarski}(n, 2)$ requires $\Omega(\log^2 n)$ queries. Their lower bound uses the family of "herringbone" functions which have a unique fixed point. Therefore, the same $\Omega(\log^2 n)$ lower bound also holds for the unique Tarski fixed point problem over $[n]^2$, where the input function is not only monotone but also promised to have a unique fixed point. Let $\textsc{UniqueTarski}(n, k)$ denote the unique Tarski problem over $[n]^k$. Given that $\textsc{UniqueTarski}(n, 2)$ is as hard as $\textsc{Tarski}(n, 2)$, is it the case for general k? or maybe $\textsc{UniqueTarski}(n, k)$ is easier than $\textsc{Tarski}(n, k)$ for larger k? This was posed as an open question in [4].

Our main result is a black-box reduction from $\textsc{Tarski}(n, k)$ to $\textsc{UniqueTarski}(n, k)$, which shows that the phenomenon observed in [4] between query complexities of $\textsc{Tarski}(n, 2)$ and $\textsc{UniqueTarski}(n, 2)$ holds for general k.

▶ **Theorem 3.** *Let $q^{\mathrm{T}}(n,k)$ be the query complexity of* Tarski(n,k) *and $q^{\mathrm{UT}}(n,k)$ be the query complexity of* UniqueTarski(n,k)*. Then $q^{\mathrm{T}}(n,k) = q^{\mathrm{UT}}(n,k)$.*

Remark. In fact, we will show that the query complexity of Tarski(n,k) is exactly the same as that of a seemingly even easier (more structural) problem: finding a fixed point of a monotone function over $[n]^k$ with the promise that *every slice has a unique fixed point*. See Lemma 16 for more details.

Note that the query complexity of UniqueTarski(n,k) is trivially at most that of Tarski(n,k). In the rest of the paper, we prove the other direction by giving a reduction from Tarski(n,k) to UniqueTarski(n,k) via a novel framework we call *partial information reductions*. We believe that this framework is of independent interest and we expect that it can be applied to a wider range of search problems concerning their query complexities.

1.1 Sketch of the Reduction

Unlike standard reductions that map from instances to instances, our reduction transforms any given algorithm for UniqueTarski (denoted by $\mathcal{U}$) to an algorithm for Tarski (our main algorithm, Algorithm 1) while keeping the query complexity the same. To the best of our knowledge, we have not seen such a non-standard black-box reduction before, and we view this as a conceptual contribution of this work. We would like to highlight the following high-level roadmap: Algorithm 1 will simulate $\mathcal{U}$, but provide it with modified answers to its queries to the oracle. These modified answers are constructed adaptively on-the-fly, and depend on what previous queries $\mathcal{U}$ has made. It may seem dangerous to modify the answers to the queries in the first place, but our reduction makes sure that the answers fed to $\mathcal{U}$ are always safe, in the sense that they always correspond to some monotone function with a unique fixed point, and any fixed point that is found by $\mathcal{U}$ must also be a fixed point of the original monotone function. Let's explain the reduction in more detail next.

Let $\mathcal{U}$ be a deterministic query algorithm for UniqueTarski(n,k) with query complexity $q(n,k)$. Given any monotone function $g : [n]^k \to [n]^k$ that has a unique fixed point x^*, $\mathcal{U}$ always finds x^* by querying it within the first $q(n,k)$ queries. At a high level, we would like to simulate $\mathcal{U}$ to find a fixed point of *any* monotone function $f : [n]^k \to [n]^k$ as an input to Tarski(n,k). But clearly we cannot run $\mathcal{U}$ on f directly since the latter may have multiple fixed points and it is likely that after some queries, answers that $\mathcal{U}$ receives are not consistent with any monotone function with a unique fixed point, in which case $\mathcal{U}$ may fail to find a fixed point within $q(n,k)$ queries. (See Figure 2 in Section 4 for an example.)

Instead, our reduction needs to serve as a surrogate between f and $\mathcal{U}$ to achieve the following two goals that are seemingly contradictory to each other:

- **(i)** On the one hand, we need to fool $\mathcal{U}$ by making sure that answers it receives during the whole process are consistent with some monotone function that has a unique fixed point.
  So from $\mathcal{U}$'s point of view, the function it interacts with can totally be a monotone function with a unique fixed point. Let's refer to this function, which is made up by our reduction, by g. Given that we cannot always return $f(x)$ to each query x of $\mathcal{U}$, the true input function f can potentially disagree significantly with the fake function g that $\mathcal{U}$ interacts with (see the comparison of Figure 2b and 4d);
- **(ii)** On the other hand, the way we answer queries to $\mathcal{U}$ (or the way we make up this fake function g) needs to achieve that, whenever $\mathcal{U}$ finds a fixed point of the fake function g (which always happens within $q(n,k)$ queries if the first goal is met), the same point must be a fixed point of the true input function f as well.

We achieve these goals using *partial information functions* (or PI functions in short).

A PI function p over $[n]^k$ is a map from $[n]^k$ to $\{-1, 0, 1, \leq, \geq, \diamond\}^k$. Intuitively a PI function p reveals some partial information of an unknown function $h : [n]^k \to [n]^k$. (For example, $p(x)_i = 1$ implies that $h(x)_i > x_i$, $p(x)_i = \geq$ implies that $h(x)_i \geq x_i$ and $p(x)_i = \diamond$ implies no information about $h(x)_i$; the connection will become cleaner after we introduce the notion of simple functions at the beginning of Section 2.) Moreover, we say a PI function p is *monotone* if it reveals some partial information of an unknown monotone function so one should not be able to infer from p any violation to monotonicity; see Definition 7.

Let $f : [n]^k \to [n]^k$ be the input monotone function. Our main algorithm, Algorithm 1 solves $\textsc{Tarski}(n, k)$ on f by simulating $\mathcal{U}$ round by round as follows: During the t-th round, $t = 1, 2, \ldots,$

1. Algorithm 1 runs $\mathcal{U}$ to obtain the t-th point $q^t \in [n]^k$ that $\mathcal{U}$ would like to query;
2. Algorithm 1 queries f to obtain $f(q^t)$ and uses it to obtain the answer $a^t \in [n]^k$ to the query. (As discussed earlier, a^t is not necessarily the same as $f(q^t)$; picking a^t based on $f(q^t)$ and the query history is the part that heavily relies on the use of PI functions.)
3. Finally Algorithm 1 sends a^t to $\mathcal{U}$ as the result of its t-th query, and moves onto round $t + 1$ (unless $f(q^t) = q^t$ so a fixed point of f has already been found).

Algorithm 1 picks answers a^t to queries of $\mathcal{U}$ by maintaining a monotone PI function p to connect f with $\mathcal{U}$. After receiving the t-th query q^t from $\mathcal{U}$, Algorithm 1 uses $f(q^t)$ to update the current PI function and then uses the updated PI function to set the answer a^t to $\mathcal{U}$. The design of the updating rule for the PI function (see the main subroutine `Generate-PI-Function` in Section 3) to achieve both goals (i) and (ii) discussed earlier is the most challenging part of the paper.

2 Partial-Information Functions

For $a, b \in \mathbb{Z}^k$ with $a \preceq b$, we write $\mathcal{L}_{a,b}$ to denote the set of points $x \in \mathbb{Z}^k$ with $a \preceq x \preceq b$.

We say a function $f : [n]^k \to [n]^k$ is a *simple* function if it satisfies $|f(x)_i - x_i| \leq 1$ for all $x \in [n]^k$ and $i \in [k]$ (i.e., $f(x)_i - x_i \in \{-1, 0, 1\}$). Let $\text{sgn}(a)$ for a number a be $1, 0, -1$ respectively if $a > 0, a = 0, a < 0$. We include the following folklore observations:

▶ **Observation 4.** *For any monotone function $f : [n]^k \to [n]^k$, let $g : [n]^k \to [n]^k$ be defined as*

$$g(x)_i \coloneqq x_i + \text{sgn}(f(x)_i - x_i), \quad \text{for all } x \in [n]^k \text{ and } i \in [k].$$

Then g is a monotone simple function and satisfies $\texttt{Fix}(g) = \texttt{Fix}(f)$.

It follows that for both Tarski and UniqueTarski, we may assume without loss of generality that the input monotone function $f : [n]^k \to [n]^k$ is simple.

▶ **Observation 5.** *A simple function $f : [n]^k \to [n]^k$ is monotone if and only if it satisfies the following conditions:*

(1) *$f(x)_i = x_i + 1$ implies $f(y)_i = y_i + 1$ and $f(y + e_i)_i \geq y_i + 1$ for all y with $x \preceq y$ and $x_i = y_i$;*

(2) *$f(x)_i = x_i - 1$ implies $f(y)_i = y_i - 1$ and $f(y - e_i)_i \leq y_i - 1$ for all y with $x \succeq y$ and $x_i = y_i$; and*

(3) *$f(x)_i = x_i$ implies (a) $f(y)_i \leq y_i$ for all y with $x \succeq y$ and $x_i = y_i$, and (b) $f(y)_i \geq y_i$ for all y with $x \preceq y$ and $x_i = y_i$.*

Observation 5 provides an alternative way to check the monotonicity of a simple function. It will mainly serve to verify the monotonicity of the following introduced *partial-information* functions. All functions from $[n]^k \to [n]^k$ we deal with from now on are assumed to be simple; for convenience, we will skip the word "simple" in the rest of the paper.

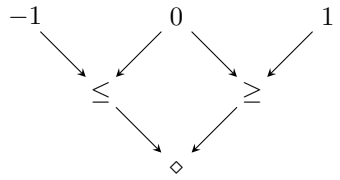


Figure 1 The information partial order. Arrow means "dominates" or "more informative".

Now we define *partial-information* (PI) functions. A PI function over $[n]^k$ is a function from $[n]^k$ to $\{-1, 0, 1, \leq, \geq, \diamond\}^k$. Intuitively a PI function reveals some partial information on the values of an underlying function $f : [n]^k \to [n]^k$; the next definition illustrates meanings of symbols in $\{-1, 0, 1, \leq, \geq, \diamond\}$:

▶ **Definition 6** (Consistency). *A function $g : [n]^k \to [n]^k$ and a PI function $p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}^k$ are* consistent *if the following conditions hold for all $x \in [n]^k$ and $i \in [k]$:*

- *$p(x)_i = -1$ implies $g(x)_i - x_i = -1$;*
- *$p(x)_i = 0$ implies $g(x)_i - x_i = 0$;*
- *$p(x)_i = 1$ implies $g(x)_i - x_i = 1$;*
- *$p(x)_i = \leq$ implies $g(x)_i - x_i \in \{-1, 0\}$;*
- *$p(x)_i = \geq$ implies $g(x)_i - x_i \in \{0, 1\}$; and*
- *$p(x)_i = \diamond$ implies nothing about $g(x)_i$.*

We introduce a natural partial order over symbols in $\{-1, 0, 1, \leq, \geq, \diamond\}$, illustrated in Figure 1. We say α *dominates* β (or α is more informative than β, denoted by $\alpha \Rightarrow \beta$), for some $\alpha, \beta \in \{-1, 0, 1, \leq, \geq, \diamond\}$, if either $\alpha = \beta$ or there is a path from α to β. With this notation, we have that $g : [n]^k \to [n]^k$ is consistent with a PI function p iff $g(x)_i - x_i \Rightarrow p(x)_i$ for all $x \in [n]^k$ and $i \in [k]$. Given two PI functions $p', p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}^k$, we say p' *dominates* p (or p' is more informative than p, denoted by $p' \Rightarrow p$) if $p'(x)_i \Rightarrow p(x)_i$ for all $x \in [n]^k$ and $i \in [k]$.

Given that we are interested in monotone functions $f : [n]^k \to [n]^k$, we introduce the notion of *monotone* PI functions below. Intuitively a PI function p is monotone if it reveals some partial information of a monotone function (so one cannot infer from p any violation to monotonicity):

▶ **Definition 7** (Monotone PI Functions). *A PI function $p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}^k$ is said to be* monotone *if it satisfies the following conditions: For any $x \in [n]^k$ and $i \in [k]$,*

(1) *$p(x)_i = 1$ implies $p(y)_i = 1$ and $p(y + e_i)_i \in \{1, 0, \geq\}$ for all y with $x \preceq y$ and $x_i = y_i$;*
(2) *$p(x)_i = -1$ implies $p(y)_i = -1$ and $p(y - e_i)_i \in \{-1, 0, \leq\}$ for all y with $x \succeq y$ and $x_i = y_i$;*
(3) *$p(x)_i = 0$ implies (a) $p(y)_i \in \{0, -1, \leq\}$ for all y with $x \succeq y$ and $x_i = y_i$, and (b) $p(y)_i \in \{0, 1, \geq\}$ for all y with $x \preceq y$ and $x_i = y_i$;*
(4) *$p(x)_i = \leq$ implies $p(y)_i \in \{-1, \leq\}$ for all y with $x \succeq y$ and $x_i = y_i$;*
(5) *$p(x)_i = \geq$ implies $p(y)_i \in \{1, \geq\}$ for all y with $x \preceq y$ and $x_i = y_i$;*
(6) *If $x_i = 1$, then $p(x)_i \in \{0, 1, \geq\}$; and*
(7) *If $x_i = n$, then $p(x)_i \in \{0, -1, \leq\}$.*

A PI function is weakly monotone *if it satisfies (1)–(5) above, but not necessarily (6) and (7).*

Note that items (6) and (7) are only about the boundary constraints. Weak monotonicity will only appear for the simplicity of the proofs below and there are no technical details behind them.

The next lemma shows that every monotone PI function is consistent with at least one monotone function. (Looking ahead, later in Section 3.2 we will give a sufficient condition for a monotone PI function to be consistent with at least one monotone function with a unique fixed point.)

▶ **Lemma 8.** *For every monotone PI function p over $[n]^k$, there exists a monotone function $g : [n]^k \to [n]^k$ that is consistent with p.*

Proof. Given p we define $g : [n]^k \to [n]^k$ as follows:

$$g(x)_i \coloneqq \begin{cases} x_i + p(x)_i & \text{if } p(x)_i \in \{-1, 0, 1\}; \\ x_i & \text{otherwise.} \end{cases}$$

We will prove g is a monotone function that is consistent with p by Observation 5.

Fix any point x and coordinate i.

Suppose that $g(x)_i = x_i + 1$, then we have $p(x)_i = 1$. Since $p(x)_i = 1$ implies $p(y)_i = 1$ and $p(y + e_i)_i \in \{1, 0, \geq\}$ for all y such that $x \preceq y$ and $x_i = y_i$, we have $g(x)_i = x_i + 1$ implies $g(y)_i = y_i + 1$ and $g(y + e_i)_i \geq y_i + 1$ for all y such that $x \preceq y$ and $x_i = y_i$.

The proof of the case that $g(x)_i = x_i - 1$ is symmetric.

Suppose that $g(x)_i = x_i$, then we have $p(x)_i \in \{0, \leq, \geq, \diamond\}$. This implies that (a) $p(y)_i \neq 1$ for all y such that $x \succeq y$ and $x_i = y_i$, and (b) $p(y)_i \neq -1$ for all y such that $x \preceq y$ and $x_i = y_i$. So we have (a) $g(y)_i \leq y_i$ for all y such that $x \succeq y$ and $x_i = y_i$, and (b) $g(y)_i \geq y_i$ for all y such that $x \preceq y$ and $x_i = y_i$. ◀

Given two elements $\alpha, \beta \in \{-1, 0, 1, \leq, \geq, \diamond\}$, if their least upper bound (or join) in the partial order exists, we write $\alpha \cap \beta$ to denote it and say that $\alpha \cap \beta$ is well defined; otherwise (when their least upper bound does not exist), we say $\alpha \cap \beta$ is not well defined. (For example, $\geq \cap \leq = 0$ and $\geq \cap -1$ is not well defined.) Given two PI functions $p^1, p^2 : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}^k$, we define their *intersection* $p^1 \cap p^2$ to be the PI function p such that $p(x)_i = p^1(x)_i \cap p^2(x)_i$ for all $x \in [n]^k$ and $i \in [k]$. The intersection $p^1 \cap p^2$ is well defined only when $p^1(x)_i \cap p^2(x)_i$ is well defined for all $x \in [n]^k$ and $i \in [k]$.

The reason that we introduce the operation of intersections is the following lemma which we will often use to modify a given monotone PI function:

▶ **Lemma 9.** *Let p^1 be a monotone PI function and p^2 be a weakly monotone PI function, both over $[n]^k$, such that $p^1 \cap p^2$ is well defined. Then $p^1 \cap p^2$ is also a monotone PI function and it satisfies $p^1 \cap p^2 \Rightarrow p^1$.*

Proof. The part about $p^1 \cap p^2 \Rightarrow p^1$ is trivial.

Note that $p^1 \cap p^2$ satisfies (6) and (7) in Definition 7 since p^1 is a monotone PI function. So in what follows, we will verify (1)-(5) for $p^1 \cap p^2$.

To show $p^1 \cap p^2$ satisfies (1), fix $x \in [n]^k$ and $i \in [k]$. If $p^1 \cap p^2(x)_i = 1$, then either $p^1(x)_i = 1$ or $p^2(x)_i = 1$. Suppose that $p^\tau(x)_i = 1$ for $\tau \in \{1, 2\}$. Then we have $p^\tau(y)_i = 1$ and $p^\tau(y + e_i)_i \in \{1, \geq\}$ for all $y \succeq x$ and $y_i = x_i$. So we have $p^1 \cap p^2(y)_i = 1$ and $p^1 \cap p^2(y + e_i)_i \in \{1, \geq\}$ for all $y \succeq x$ and $y_i = x_i$.

Items (2)-(5) can be verified similarly. ◀

Given a monotone function $f : [n]^k \to [n]^k$ and a monotone PI function p over $[n]^k$, we define a function $f|_p : [n]^k \to [n]^k$ as follows: For any $x \in [n]^k$ and $i \in [k]$, let

$$f|_p(x)_i = \begin{cases} x_i + p(x)_i & \text{if } p(x)_i \in \{-1, 0, 1\}; \\ \max(f(x)_i, x_i) & \text{if } p(x)_i = \geq; \\ \min(f(x)_i, x_i) & \text{if } p(x)_i = \leq; \\ f(x)_i, & \text{if } p(x)_i = \diamond. \end{cases}$$

Note that $f|_p$ is a function that is consistent with p (but may disagree with f in general). Looking ahead, our algorithm for TARSKI running on f will maintain a monotone PI function p and (essentially) use $f|_p$ to answer the next query from an algorithm for UNIQUETARSKI it simulates. As it will become clear in Section 3, using $f|_p$ (with a carefully updated p) instead of f to answer queries is crucial in making sure answers to the algorithm for UNIQUETARSKI are consistent with a monotone function with a unique fixed point (given that the input function f to TARSKI can have multiple fixed points in general).

We record the following lemma about $f|_p$:

▶ **Lemma 10.** *Let f be a monotone function and p be a monotone PI function, both over $[n]^k$. Then $f|_p : [n]^k \to [n]^k$ is also a monotone function and is consistent with p.*

Proof. The part about $f|_p$ being consistent with p is easy, since $f|_p(x)_i - x_i = p(x)_i$ when $p(x)_i \in \{-1, 0, 1\}$; $f|_p(x)_i - x_i \in \{0, 1\}$ when $p(x)_i = \geq$; $f|_p(x)_i - x_i \in \{0, -1\}$ when $p(x)_i = \leq$; and $f|_p(x)_i - x_i \in \{-1, 0, 1\}$ when $p(x)_i = \diamond$.

Note that since f is a function from $[n]^k$ to $[n]^k$ and p is a monotone PI function that satisfies the boundary conditions (6) and (7), we have $1 \leq f|_p(x)_i \leq n$ for all x and i.

To show $f|_p$ is monotone, given any $x \in [n]^k$ and $i \in [k]$, we consider three cases where $f|_p(x)_i = x_i + 1$, $f|_p(x)_i = x_i - 1$, and $f|_p(x)_i = x_i$.

Suppose that $f|_p(x)_i = x_i + 1$. Then either $f(x)_i - x_i = 1$ or $p(x)_i = 1$. If $f(x)_i - x_i = 1$ and $p(x)_i \in \{1, \geq, \diamond\}$, then we have (i) $f(y)_i - y_i = 1$ and $f(y + e_i)_i \geq y_i + 1$, and (ii) $p(y)_i \in \{1, \geq, \diamond\}$ and $p(y + e_i)_i \neq -1$ for all $y \succeq x$ with $y_i = x_i$, which imply $f|_p(y)_i - y_i = 1$ and $f(y + e_i)_i \geq y_i + 1$. The proof is similar in the case $p(x)_i = 1$.

The proof of the case $f|_p(x)_i = x_i - 1$ is symmetric.

Suppose that $f|_p(x)_i = x_i$. Then one of the following four cases meets: (1) $f(x)_i = x_i$ and $p(x)_i \in \{\leq, \geq, \diamond\}$; (2) $p(x)_i = 0$; (3) $f(x)_i - x_i = 1$ and $p(x)_i = \leq$; (4) $f(x)_i - x_i = -1$ and $p(x)_i = \geq$. Let's prove the first case and others are similar. Suppose that $f(x)_i = x_i$ and $p(x)_i \in \{\leq, \geq, \diamond\}$, then we have $f(y)_i \leq y_i$ for all $y \preceq x$ with $y_i = x_i$ and $f(y)_i \geq y_i$ for all $y \succeq x$ with $y_i = x_i$. Since $p(x)_i \in \{\leq, \geq, \diamond\}$, we have $p(y)_i \neq 1$ for all $y \preceq x$ with $y_i = x_i$ and $p(y)_i \neq -1$ for all $y \succeq x$ with $y_i = x_i$. This finishes the proof. ◀

Before moving to the main reduction, we need to introduce the notion of slices. We note that the notion of slices was also used in the literature.

▶ **Definition 11** (Slices). *A* slice *of $[n]^k$ is specified by a tuple $s \in ([n] \cup \{*\})^k$. Given s, we write $\mathcal{L}_s$ to denote the set of points x such that $x_i = s_i$ for all i such that $s_i \neq *$.*

Given a monotone PI function p and a slice s, we say a point $x \in \mathcal{L}_s$ is a *postfixed* point of p on the slice s if $p(x)_i \in \{1, 0, \geq\}$ for all i with $s_i = *$ and a point $x \in \mathcal{L}_s$ is a *prefixed* point of p on the slice s if $p(x)_i \in \{-1, 0, \leq\}$ for all i with $s_i = *$.

We use $\texttt{Post}_s(p)$ to denote the set of postfixed points of p on s and $\texttt{Pre}_s(p)$ to denote the set of prefixed points of p on s.

▶ **Lemma 12.** *Given a monotone PI function p, for any slice s, $\texttt{Post}_s(p)$ is a join-semilattice and $\texttt{Pre}_s(p)$ is a meet-semilattice.*

Proof. Fix a slice s and consider any two points $x, y \in \texttt{Post}_s(p)$. Then we have $p(x)_i, p(y)_i \in \{1, 0, \geq\}$ for all i with $s_i = *$. Let $z = x \vee y$ be the join of x and y, namely, the coordinatewise maximum of x and y. Then we have $x \preceq z$ and $y \preceq z$ and either $z_i = x_i$ or $z_i = y_i$ for all i with $s_i = *$. So by the monotonicity of p, we have $p(z)_i \in \{1, 0, \geq\}$.

The proof of that $\texttt{Pre}_s(p)$ is a meet-semilattice is similar. This finishes the proof. ◀

Lemma 12 guarantees that the join of $\texttt{Post}_s(p)$ is well defined and the meet of $\texttt{Pre}_s(p)$ is well defined. We write $J_s(p)$ to denote the join of $\texttt{Post}_s(p)$ and $M_s(p)$ to denote the meet of $\texttt{Pre}_s(p)$. When the context is clear, we may omit p for the simplicity of notations.

▶ **Proposition 13.** *Given a monotone PI function p, for any slice s, we have $p(J_s)_i \in \{0, \geq\}$ for all i with $s_i = *$ and $p(M_s)_i \in \{0, \leq\}$ for all i with $s_i = *$.*

Proof. Consider any point $x \in \texttt{Post}_s$. Suppose that there exists i with $s_i = *$ such that $p(x)_i = 1$, then we have $x + e_i \in \texttt{Post}_s$ as well. This means x can not be J_s. So we have $p(J_s)_i \in \{0, \geq\}$ for all i with $s_i = *$. The proof of $p(M_s)$ is similar. ◀

3 The Partial-Information Reduction and Proof of Theorem 3

We prove Theorem 3 in this section. Let $\mathcal{U}$ be any query algorithm for UniqueTarski(n, k) with query complexity $q(n, k)$. We show that our main algorithm, Algorithm 1, can employ $\mathcal{U}$ to solve Tarski(n, k) with the same number of queries.

Let's continue from the sketch presented in Section 1.1 and elaborate more on how Algorithm 1 works. Algorithm 1 computes the answer a^t to the t-th query q^t of $\mathcal{U}$ by maintaining a sequence of monotone PI functions $p^0, p^1, \ldots$, where p^0 is the initial monotone PI function set by the boundary conditions (see line 2 of Algorithm 1) and p^t is the monotone PI function it maintains at the end of the t-th round. During the t-th round, Algorithm 1 first continues to run $\mathcal{U}$ to obtain the t-th query q^t. It then queries f to obtain $f(q^t)$ and uses the latter to update the PI function p^{t-1} to p^t. Finally the answer a^t to $\mathcal{U}$ is set to be $f|_{p^t}(q^t) \in [n]^k$.

The correctness of Algorithm 1 relies on the following list of properties of p^t: For every t, p^t is a monotone PI function such that

1. $p^t(q^j) + q^j = a^j$ for all $j \in [t]$ (i.e., p^t agrees with answers to all queries $\mathcal{U}$ has made so far);
2. There is a monotone function g that is consistent with p^t and has a unique fixed point;
3. Any fixed point of $f|_{p^t}$ must be a fixed point of f.

To see that Algorithm 1 always finds a fixed point of f within $q(n, k)$ queries, we note that item 3 above implies that q^t is a fixed point of f if $a^t = f|_{p^t}(q^t)$ is the same as q^t. So the only bad case is that $a^t \neq q^t$ for all $t = 1, \ldots, q(n, k)$. However, this cannot happen because after $q(n, k)$ rounds, by item 2, there is a monotone function g that is consistent with $p^{q(n,k)}$ and has a unique fixed point, and by item 1, $g(q^j) = a^j$ for all $j \in [q(n, k)]$. So g is a monotone function that has a unique fixed point, on which $\mathcal{U}$ fails to find a fixed point (since $a^j \neq q^j$ for all $j \in [q(n, k)]$).

3.1 Subroutine `Generate-PI-Function`

The main challenge is about how to update the PI function p^{t-1} to p^t during the t-th round to maintain properties listed above for the correctness of the algorithm. This is done by making calls to a subroutine called `Generate-PI-Function` (see Algorithm 1; in general it may take k calls to `Generate-PI-Function` to obtain p^t during the t-th round).

The subroutine `Generate-PI-Function`(p, q, ℓ, b) takes four inputs, namely, a PI function p, a point $q \in [n]^k$, an index $\ell \in [k]$, and a sign $b \in \{-1, 0, 1\}$, and returns a new PI function. Before stating the main technical theorem about `Generate-PI-Function`, we need the following definition:

▶ **Definition 14.** *We say a monotone PI function p is* safe *if, for every slice $s \in ([n] \cup \{*\})^k$, it satisfies*

(1) *for any point* $x \in \mathcal{L}_s$ *and* $x \prec J_s(p)$, $p(x)_i \in \{-1,0,1\}$ *for all* i *with* $x_i < J_s(p)_i$ *and* $p(x)_i = 1$ *for some* i *with* $x_i < J_s(p)_i$*; and*
(2) *for any point* $x \in \mathcal{L}_s$ *and* $x \succ M_s(p)$, $p(x)_i \in \{-1,0,1\}$ *for all* i *with* $x_i > M_s(p)_i$ *and* $p(x)_i = -1$ *for some* i *with* $x_i > M_s(p)_i$.

We are now ready to state our main technical theorem:

▶ **Theorem 15** (Main Technical Theorem). *Given a monotone and safe PI function* p, $q \in [n]^k$, $\ell \in [k]$, *and* $b \in \{-1,0,1\}$ *such that* $(p(q)_\ell, b)$ *satisfies the following condition:*

$$p(q)_\ell \in \{\geq, \diamond\} \text{ if } b = 1;\ \ p(q)_\ell \in \{\leq, \diamond\} \text{ if } b = -1;\ \ p(q)_\ell \in \{\leq, \geq, \diamond\} \text{ if } b = 0, \tag{1}$$

the function p^r *returned by* `Generate-PI-Function`(p, q, ℓ, b) *satisfies the following properties:*
1. p^r *is also a monotone PI function;*
2. $p^r \Rightarrow p$*;*
3. $p^r(q)_\ell = b$*; and*
4. p^r *remains safe.*

Additionally, if $f : [n]^k \to [n]^k$ *is a monotone function such that* $\texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$ *and* $f|_p(q)_\ell = q_\ell + b$*, then we have* $\texttt{Fix}(f|_{p^r}) \subseteq \texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$.

We prove Theorem 15 in the rest of the section. An important property of safe, monotone PI functions is given in the following lemma which we prove in the next subsection.

▶ **Lemma 16.** *If* p *is a monotone and safe PI function, then there is a monotone function* g *that is consistent with* p *and has a unique fixed point in every slice* s*. In particular,* g *has a unique fixed point in the whole lattice.*

We can use Theorem 15 and Lemma 16 to prove the main theorem:

Proof of Theorem 3. Let f be the input function. We first note that every time Algorithm 1 obtains $p^{(t,i)}$ from $p^{(t,i-1)}$, either $p^{(t,i)}$ is the same as $p^{(t,i-1)}$ or $p^{(t,i)}$ is set to be

$$\texttt{Generate-PI-Function}\big(p^{(t,i-1)}, q, i, b\big)$$

for some q, i, b that satisfy $b = f|_{p^{(t,i-1)}}(q)_i - q_i$ and (1):

$$p^{(t,i-1)}(q)_i \in \{\geq, \diamond\} \text{ if } b = 1;\ \ p^{(t,i-1)}(q)_i \in \{\leq, \diamond\} \text{ if } b = -1;\ \ p^{(t,i-1)}(q)_i \in \{\leq, \geq, \diamond\} \text{ if } b = 0,$$

Given that $p^{(1,0)} = p^0$ is monotone and safe, it follows directly from an induction using Theorem 15 that every PI function p in the following list:

$$p^{(1,0)}, \ldots, p^{(1,k)}, p^{(2,0)}, \ldots, p^{(2,k)}, \ldots, p^{(t,0)}, \ldots, p^{(t,k)}, \ldots$$

is monotone and safe, and satisfies $\texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$. Furthermore, every PI function p in the list dominates all of its predecessors and $p^{(t,i)}(q^t)_i \in \{-1,0,1\}$ for all t, i. Combining the latter with $a^t = f|_{p^t}(q^t)$, as well as that $p^t = p^{(t,k)}$ dominates all of its predecessors, we have $a^t - q^t = p^t(q^t)$. It follows that $a^j - q^j = p^t(q^j)$ for all $j \leq t$.

Let $N = q(n,k)$. Consider the following two cases:
1. If $a^t = q^t$ for some $t \in [N]$, then given that $a^t = f|_{p^t}(q^t)$ and $\texttt{Fix}(f|_{p^t}) \subseteq \texttt{Fix}(f)$, we have that q^t is a fixed point of f. In this case Algorithm 1 succeeds within $q(n,k)$ queries;
2. Otherwise, we have $a^t \neq q^t$ for all $t \in [N]$. In this case, given that p^N is both monotone and safe, Lemma 16 implies that there exists a monotone function g that is consistent with p^N and has a unique fixed point. However, given that $a^j - q^j = p^N(q^j)$ for all $j \leq N$, we have that $q^j \neq a^j = g(q^j)$ for all $j \in [N]$. As a result, $\mathcal{U}$ fails to find a fixed point of g within its N queries $q^1, \ldots, q^N$, which it should given that g is a monotone function with a unique fixed point, a contradiction.

This finishes the proof of the theorem. ◀

Algorithm 1 Algorithm for $\textsc{Tarski}(n,k)$ via the algorithm $\mathcal{U}$ for $\textsc{UniqueTarski}(n,k)$.

Let $\mathcal{U}$ be an algorithm for $\textsc{UniqueTarski}(n,k)$.
Let p^0 be an empty PI function with the initial boundary conditions, i.e., $p^0(x)_i = \geq$ if $x_i = 1$; $p^0(x)_i = \leq$ if $x_i = n$; and $p^0(x)_i = \diamond$ otherwise for all $x \in [n]^k$ and $i \in [k]$.
Let $t \leftarrow 1$ be the round number.
do
  Let q^t be the point queried by $\mathcal{U}$ and make one query to get $f(q^t)$.
  Let $p^{(t,0)} \leftarrow p^{t-1}$.
  for *each* i *from* 1 *to* k **do**
    If $p^{(t,i-1)}(q^t)_i \in \{-1,0,1\}$, let $p^{(t,i)} \leftarrow p^{(t,i-1)}$.
    Otherwise, let $p^{(t,i)} \leftarrow \texttt{Generate-PI-Function}(p^{(t,i-1)}, q^t, i, f|_{p^{(t,i-1)}}(q^t)_i - q^t_i)$.
  Let $p^t \leftarrow p^{(t,k)}$ and use $a^t \leftarrow f|_{p^t}(q^t)$ as the answer to the algorithm $\mathcal{U}$.
  If $q^t = a^t$, then **return** q^t as the fixed point and **terminate**.
  Else, let $t \leftarrow t+1$.
while;

Subroutine 2 `Generate-PI-Function`(p,q,ℓ,b).

If $b = 1$, then **return** `Generate-PI-Function-Plus`(p,q,ℓ).
If $b = -1$, then **return** `Generate-PI-Function-Minus`(p,q,ℓ).
If $b = 0$, then **return** `Generate-PI-Function-Zero`(p,q,ℓ).

Subroutine 3 `Generate-PI-Function-Plus`(p,q,ℓ).

Initialize $p' \leftarrow p$.
Let $p'(x)_\ell \leftarrow 1$ and $p'(x+e_\ell)_\ell \leftarrow p'(x+e_\ell)_\ell \cap \geq$ for all x such that $x \succeq q$ and $x_\ell = q_\ell$.
Initialize $p^+(x)_i \leftarrow \diamond$ for all x and i as a weak PI function.
for *each* $x \in [n]^k$ *and each* $i \in [k]$ **do**
  if *there exists* y *such that (a)* $x \preceq y$*; (b)* $x_i < y_i$*; and (c)* $x_j = y_j$ *for all* j *with* $p'(y)_j \notin \{1,0,\geq\}$ **then**
    If $p'(x)_i \in \{1,\geq,\diamond\}$, let $p^+(x)_i \leftarrow 1$ and $p^+(x+e_i)_i \leftarrow p^+(x+e_i)_i \cap \geq$.
    If $p'(x)_i \in \{0,\leq\}$, let $p^+(x)_i \leftarrow \geq$.
Let $p^r \leftarrow p' \cap p^+$.
return p^r.

3.2 Consequences of PI Function Being Safe

The motivation to focus on Definition 14 is that they have nice properties given in the following lemmas.

▶ **Lemma 17.** *Suppose that a PI function p is monotone and safe, then we have*
1. *$J_s(p) \preceq M_s(p)$ for all slices s; and*
2. *$g(x) \neq x$ for any monotone function g that is consistent with p and any x such that there exists s with $x \notin \mathcal{L}_{J_s,M_s}$.*

Proof. We will prove the following claim, by which we will deduce this lemma.

Subroutine 4 `Generate-PI-Function-Minus`(p, q, ℓ).

Initialize $p' \leftarrow p$.
Let $p'(x)_\ell \leftarrow -1$ and $p'(x - e_\ell)_\ell \leftarrow p'(x - e_\ell)_\ell \cap \leq$ for all x such that $x \preceq q$ and $x_\ell = q_\ell$.
Initialize $p^-(x)_i \leftarrow \diamond$ for all x and i as a weak PI function.
for *each* $x \in [n]^k$ *and each* $i \in [k]$ **do**
  if *there exists* y *such that (a)* $x \succeq y$*; (b)* $x_i > y_i$*; and (c)* $x_j = y_j$ *for all* j *with* $p'(y)_j \notin \{-1, 0, \leq\}$ **then**
    If $p'(x)_i \in \{-1, \leq, \diamond\}$, let $p^-(x)_i \leftarrow -1$ and $p^-(x - e_i)_i \leftarrow p^-(x - e_i)_i \cap \leq$.
    If $p'(x)_i \in \{0, \geq\}$, let $p^-(x)_i \leftarrow \leq$.
Let $p^r \leftarrow p' \cap p^-$.
return p^r.

Subroutine 5 `Generate-PI-Function-Zero`(p, q, ℓ).

Initialize $p^r \leftarrow p$.
If $q_\ell > 1$ and $p(q - e_\ell)_\ell \neq 1$, let $p^r \leftarrow$ `Generate-PI-Function-Plus`$(p^r, q - e_\ell, \ell)$.
If $q_\ell < n$ and $p(q + e_\ell)_\ell \neq -1$, let $p^r \leftarrow$ `Generate-PI-Function-Minus`$(p^r, q + e_\ell, \ell)$.
return p^r.

▷ Claim 18. Given the PI function p is monotone and safe, we have
(a) for any point $x \in \mathcal{L}_s$ and $x \not\preceq M_s(p)$, there exists i with $s_i = *$ and $p(x)_i = -1$; and
(b) for any point $x \in \mathcal{L}_s$ and $x \not\succeq J_s(p)$, there exists i with $s_i = *$ and $p(x)_i = 1$.

Proof. We will show that the first item in Definition 14 implies item (b), and the second item in Definition 14 implies item (a). Fix a slice s, a point $x \in \mathcal{L}_s$ and $x \not\succeq J_s(p)$. We will prove there exists i with $s_i = *$ and $p(x)_i = 1$. The proof for the item (a) is similar.

Construct a sub-slice s' as follows:

$$s'_i := \begin{cases} s_i & s_i \neq *; \\ x_i & s_i = * \text{ and } x_i \geq J_s(p^r)_i; \\ * & \text{otherwise } (s_i = * \text{ and } x_i < J_s(p^r)_i). \end{cases}$$

Then we have $s'_i = *$ implies $s_i = *$ and $x \in \mathcal{L}_{s'}$. Let z be the join of x and $J_s(p)$. Note that $z \in \mathcal{L}_{s'}$ as well. In addition, we have $x_i < z_i$ for all i with $s'_i = *$.

We will prove that $z \in \mathtt{Post}_{s'}(p)$, so we will have $z \preceq J_{s'}(p)$, which implies $x_i < J_{s'}(p)_i$ for all i with s'_i. Since p is safe, we conclude that there exists i with $s'_i = *$ and $p(x)_i = 1$. Such an i also satisfies $s_i = *$.

The statement $z \in \mathtt{Post}_{s'}(p)$ follows from the observation that whenever $s'_i = *$, we have $s_i = *$ and $z_i = J_s(p)_i$. Since $p(J_s)_i \in \{\geq, 0\}$, we have $p(z)_i \in \{1, 0, \geq\}$. ◁

We show that each of items (a) and (b) is strong enough to deduce the first item ($J_s(p) \preceq M_s(p)$ for all s). (This will be used in the proof of Lemma 22 below). Take item (b) as an example: Given any point $x \in \mathcal{L}_s$ such that $x \not\preceq M_s(p)$, since there exists i with $s_i = *$ and $p(x)_i = -1$, we have $x \notin \mathtt{Pre}_s(p)$ by definition. So $J_s(p)$ must be somewhere that is $\preceq M_s(p)$.

For the second item, consider any point x such that there exists s with $x \notin \mathcal{L}_{J_s, M_s}$. Then we know either $x \not\preceq M_s(p)$ or $x \not\succeq J_s(p)$. Since there exists i with $p(x)_i = -1$ or $p(x)_i = 1$, we have $g(x) \neq x$ as long as g is a monotone function that is consistent with p. ◀

We also present the proof of Lemma 16 in this subsection.

Proof of Lemma 16. We will refine p to a more informative monotone PI function p' such that every monotone function that is consistent with p' has in each slice s only one fixed point, $M_s(p)$.

Consider a slice s, and let $M_s = M_s(p)$ be the lowest prefixed point of p in the slice. By Claim 18, for every point $x \in \mathcal{L}_s$, if $x \not\preceq M_s$, there exists i with $s_i = *$ and $p(x)_i = -1$. Consider a point $x \in \mathcal{L}_s$ where $x \preceq M_s$, $x \neq M_s$. If i is a coordinate with $s_i = *$ and $x_i = (M_s)_i$ then $p(x)_i \in \{0, -1, \leq\}$ since $p(M_s)_i \in \{0, -1, \leq\}$. Since M_s is the lowest prefixed point in $\mathcal{L}_s$, there is a coordinate i such that $s_i = *$ and $p(x)_i \notin \{0, -1, \leq\}$, therefore $p(x)_i \in \{1, \geq, \diamond\}$ and $x_i < (M_s)_i$.

Define p' as follows. Initialize $p'(x) = p(x)$ for all $x \in [n]^k$. For each slice s and every point $x \in \mathcal{L}_s$ where $x \preceq M_s$, $x \neq M_s$, and each coordinate i such that $s_i = *$ and $p(x)_i \in \{1, \geq, \diamond\}$, set $p'(x)_i = 1$, and for every $y \succeq x$ with $y_i = x_i$ set $p'(y)_i = 1$ and $p'(y+e_i)_i = p'(y+e_i)_i \cap \geq$. (Note that $p'(y + e_i)_i$ may be also updated due to other points x', including possibly being set to 1.)

We first claim that p' is a well defined PI function and dominates (is more informative than) p. Note that p' changes the value of $p(z)_i$ for some points z and some coordinates i by either setting the value to 1 or taking the join with $\geq$. Thus, to show the claim it suffices to show that (i) p' does not set the value to 1 for any point z and coordinate i such that $p(z)_i \in \{-1, 0, \leq\}$, and (ii) it does not take the join with $\geq$ for any z and i such that $p(z)_i = -1$. To see this, consider any slice s, a point $x \in \mathcal{L}_s$ with $x \preceq M_s$, $x \neq M_s$, and a coordinate i such that $s_i = *$ and $p(x)_i \in \{1, \geq, \diamond\}$. Since $p(x)_i \in \{1, \geq, \diamond\}$, the new value $p'(x)_i = 1$ dominates $p(x)_i$. Consider any other point $y \succeq x$ with $y_i = x_i$. If $p(y)_i$ was in $\{-1, 0, \leq\}$, then $p(x)_i$ would also be in $\{-1, 0, \leq\}$ by Definition 7. We infer therefore that $p(y)_i \in \{1, \geq, \diamond\}$, thus $p'(y)_i = 1$ dominates $p(y)_i$. Also, if $p(y + e_i)_i$ was -1 then $p(x)_i$ would be in $\{-1, 0, \leq\}$. We infer therefore that $p(y + e_i)_i \neq -1$, thus the join with $\geq$ exists, it dominates $p(y + e_i)_i$, and is not -1. We conclude that p' is well defined and dominates p.

We then claim that p' is monotone. Consider any slice s, a point $x \in \mathcal{L}_s$ with $x \preceq M_s$, $x \neq M_s$, and a coordinate i such that $s_i = *$ and $p(x)_i \in \{1, \geq, \diamond\}$. Then we know $p'(x)_i = 1$. Consider any other point $y \succeq x$ with $y_i = x_i$, we have $p(y)_i \in \{1, \geq, \diamond\}$, so $p'(y)_i$ would also be 1. Also, since $p(y + e_i)_i$ is not -1, we conclude that $p'(y + e_i)_i \Rightarrow p(y + e_i)_i \cap \geq$. Since $p'(y + e_i)_i$ is well defined, we infer therefore that $p'(y + e_i)_i \in \{\geq, 0, 1\}$. For other points x and coordinates i, we have $p'(x)_i = p(x)_i$ and $p'(x)_i$ satisfying Definition 7 follow from the monotonicity of p and $p' \Rightarrow p$. We conclude that p' is monotone.

The PI function p' has the property that for every slice s and for every point $x \in \mathcal{L}_s$ with $x \neq M_s$, there exists a coordinate i such that either $p'(x)_i = -1$ (this is the case if $x \not\preceq M_s$) or $p'(x)_i = 1$ (this is the case if $x \preceq M_s$). We conclude that any monotone function that is consistent with p' has only one fixed point in each slice s, namely, M_s. Since p' dominates p, any such monotone function is also consistent with p. In particular, there exists at least one such monotone function, as constructed in Lemma 8. ◀

3.3 Preserving Monotonicity and Safety

In this subsection, we will prove items (1)–(4) of Theorem 15.

▶ **Lemma 19** (Monotonicity Preserving of Subroutine 3). *Given a monotone PI function p, a point $q \in [n]^k$ and a coordinate $\ell \in [k]$ such that $p(q)_\ell \in \{\geq, \diamond\}$ (which implies $q_\ell < n$), we have the PI function p^r returned by* `Generate-PI-Function-Plus`(p, q, ℓ) *remains monotone. Furthermore, we have $p^r \Rightarrow p$.*

Proof. We start by proving the monotonicity of p' on line 2. Since $p(q)_\ell \in \{\geq, \diamond\}$, we have $p(x)_\ell \in \{1, \geq, \diamond\}$ and $p(x + e_\ell)_\ell \in \{0, 1, \leq, \geq, \diamond\}$ for all x such that $x \succeq q$ and $x_\ell = q_\ell$. So $p(x + e_\ell)_\ell \cap \geq$ is well defined. The monotonicity of p' follows from the observation that we changed $p'(q)_\ell \leftarrow 1$ and maintained the consequences it should imply. Clearly, $p' \Rightarrow p$.

After that, we will maintain a new function p^+ from line 3 to line 7. Note that we will update $p^r \leftarrow p' \cap p^+$ on line 8 and return it. So by Lemma 9, it suffices for us to prove p^+ is a weakly monotone PI function, and $p'(x)_i \cap p^+(x)_i$ is well defined for all x and i at the end of the **for** loop.

To this end, we will prove that, at the end of the **for** loop, item (1) in Definition 7 is true for every point x and coordinate i such that $p^+(x)_i = 1$; and item (5) in Definition 7 is true for every point x and coordinate i such that $p^+(x)_i = \geq$. Before getting into details, we first provide a clearer picture of the condition of **if** on line 5.

▷ Claim 20. Given a coordinate i and two points $x \preceq x'$ such that $x_i = x'_i$, if the **if** condition on line 5 is true for x and i, then the **if** condition on line 5 is also true for x' and i.

Proof. By definition, we know there exists y such that (a) $x \preceq y$; (b) $x_i < y_i$; and (c) $x_j = y_j$ for all j with $p'(y)_j \notin \{1, 0, \geq\}$. Now we explicitly show there also exists such a y' for x'. Let y' be the join of x' and y (i.e., $y'_j = \max(x'_j, y_j)$ for all j). Then obviously we have (a) $x' \preceq y'$. Since $x_i = x'_i$, we have (b) $y'_i = y_i > x_i = x'_i$. For the last property (c), note that $y \preceq y'$. By the monotonicity, as long as $y'_j = y_j$ and $p'(y)_j \in \{1, 0, \geq\}$, we have $p'(y')_j \in \{1, 0, \geq\}$. The contrapositive tells us for every j such that $p'(y')_j \notin \{1, 0, \geq\}$, either $y'_j \neq y_j$ (then $y'_j = \max(x'_j, y_j) = x'_j$) or $p'(y)_j \notin \{1, 0, \geq\}$ (then $x_j = y_j$, so $y'_j = \max(x'_j, y_j) = \max(x'_j, x_j) = x'_j$), which is the statement of (c).

This finishes the existence of y' for x' and i. ◁

We divide the proof into two cases:

Case 1: item (1) in Definition 7. Fix a coordinate i and two points $x \preceq x'$ such that $x_i = x'_i$. Suppose that $p^+(x)_i = 1$ (which means $p'(x)_i \in \{1, \geq, \diamond\}$). By monotonicity, we have $p'(x')_i \in \{1, \geq, \diamond\}$ as well. Since the **if** condition on line 5 is true for x, by Claim 20, we know that the **if** condition is also true for x'. Combining with $p'(x')_i \in \{1, \geq, \diamond\}$, we know that $p^+(x')_i \leftarrow 1$ and $p^+(x' + e_i)_i$ is updated by $p^+(x' + e_i)_i \cap \geq$ on line 6, which means $p^+(x')_i = 1$ and $p^+(x' + e_i)_i \in \{1, \geq\}$ at the end of the **for** loop.

Case 2: item (5) in Definition 7. Fix a coordinate i and two points $x \preceq x'$ and $x_i = x'_i$. Suppose that $p^+(x)_i = \geq$. We will prove $p^+(x')_i \in \{1, \geq\}$ at the end of the **for** loop. There are two possibilities: $p^+(x)_i$ is updated on line 6 or line 7. If $p^+(x)_i$ is updated on line 6, then we have $p^+(x' - e_i)_i = 1$ and $p^+(x')_i \in \{1, \geq\}$. If $p^+(x)_i$ is updated on line 7 (which means $p'(x)_i \in \{0, \leq\}$), then we have $p'(x')_i \neq -1$. Meanwhile, by Claim 20, we know that the **if** condition on line 5 is true. So $p^+(x')_i$ will be updated by either 1 or $\geq$.

This finishes the proof that p^+ is a weakly monotone PI function before line 8.

The final step is to show that $p'(x)_i \cap p^+(x)_i$ is well defined for all x and i, which follows from the observation that $p^+(x)_i = 1$ only if $p'(x)_i \in \{1, \geq, \diamond\}$ and $p^+(x)_i = \geq$ only if $p'(x)_i \in \{0, 1, \leq, \geq, \diamond\}$ for all x and i. ◀

Symmetrically, we conclude the following lemma.

▶ **Lemma 21** (Monotonicity Preserving of Subroutine 4). *Given a monotone PI function p, a point $q \in [n]^k$ and a coordinate $\ell \in [k]$ such that $p(q)_\ell \in \{\leq, \diamond\}$ (which implies $q_\ell > 1$), we have the function p^r returned by* `Generate-PI-Function-Minus`(p, q, ℓ) *remains monotone. Furthermore, we have $p^r \Rightarrow p$.*

▶ **Lemma 22** (Safety Preserving of Subroutine 3). *Given a monotone and safe PI function $p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}^k$, a point q and a coordinate ℓ such that $p(q)_\ell \in \{\geq, \diamond\}$, we have the PI function p^r returned by* `Generate-PI-Function-Plus`(p, q, ℓ) *remains safe.*

Proof. Since $p(q)_\ell \in \{\geq, \diamond\}$, we know that p^r returned by `Generate-PI-Function-Plus`(p, q, ℓ) is also monotone and $p^r \Rightarrow p$ by Lemma 19.

Note that in the subroutine `Generate-PI-Function-Plus`, p^r is obtained by only adding 1 and $\geq$ on the function p. So we have $M_s(p^r) = M_s(p)$ for every slice s. By the same reason, p is safe, and $p^r \Rightarrow p$, we have for any point $x \in \mathcal{L}_s$ with $x \succ M_s(p^r)$, $p^r(x)_i \in \{-1, 0, 1\}$ for all i with $x_i > M_s(p^r)_i$ and $p^r(x)_i = -1$ for some i with $x_i > M_s(p^r)_i$ for all slices s. As a corollary, we have $J_s(p^r) \preceq M_s(p^r)$ for all s, derived from the proof of Lemma 17. (This corollary will be used in this proof later).

So we will focus on proving the first item in Definition 14 for p^r, namely, we will prove for any point $x \in \mathcal{L}_s$ with $x \prec J_s(p^r)$, $p^r(x)_i \in \{-1, 0, 1\}$ for all i such that $x_i < J_s(p^r)_i$ and $p^r(x)_i = 1$ for some i with $x_i < J_s(p^r)_i$.

We first prove the first part: for any point $x \in \mathcal{L}_s$ and $x \prec J_s(p^r)$, $p^r(x)_i \in \{-1, 0, 1\}$ for all i such that $x_i < J_s(p^r)_i$. Fix arbitrarily a slice s, a point $x \in \mathcal{L}_s$ such that $x \prec J_s(p^r)$ and i such that $x_i < J_s(p^r)_i$. We will show that the **if** condition on line 5 is true for x and i. (Note that we need to show there exists a point y such that (a) $x \preceq y$; (b) $x_i < y_i$; and (c) $x_j = y_j$ for all j with $p'(y)_j \notin \{1, 0, \geq\}$. One may try to directly use $J_s(p^r)$ to serve as that y. But note that the definition of $J_s(p^r)$ only guarantees that $x_j = y_j$ for all j with $p^r(y)_j \notin \{1, 0, \geq\}$ instead of what we need in (c) (which concerns $p'(y)$). So extra effort is needed here.)

Let $\mathtt{Y} \coloneqq \{y \mid \text{there exists } i' \text{ such that } J_s(p^r) - e_{i'} \preceq y, (J_s(p^r) - e_{i'})_{i'} < y_{i'} \text{ and } (J_s(p^r) - e_{i'})_j = y_j \text{ for all } j \text{ with } p'(y)_j \notin \{1, 0, \geq\}\}$. Let y^* be the join of $\mathtt{Y} \cup \{J_s(p^r)\}$. Then we prove the following claim.

▷ Claim 23. y^* could serve as the y for the **if** condition on line 5 for x and i.

Proof. Since $x \preceq J_s(p^r)$ and $x_i < J_s(p^r)_i$, we have $x \preceq y^*$ and $x_i < y^*_i$. So in what follows, we will show $p'(y^*)_j \in \{1, 0, \geq\}$ for all j such that $x_j < y^*_j$.

If $\mathtt{Y} = \emptyset$, then we know that $p^+(J_s(p^r))_i = \diamond$ for all i (since any $y \in \mathtt{Y}$ along with the i' should active the condition on line 5, which will update $(J_s(p^r))_{i'}$). So we have $p^r(J_s(p^r))_i = p'(J_s(p^r))_i$ for all i, which implies $x_j = J_s(p^r)_j$ for all j with $p'(J_s(p^r))_j \notin \{1, 0, \geq\}$. This means $y^* = J_s(p^r)$ itself could serve as the y for the **if** condition on line 5 for x and i.

Now let's consider the case $\mathtt{Y} \neq \emptyset$ and let j be such that $x_j < y^*_j$. Let $y \in \mathtt{Y}$ be such that $y_j = y^*_j$ (which must exist since $J_s(p^r) \preceq y$ for all $y \in \mathtt{Y}$). Since $J_s(p^r)_j \leq y^*_j$, we have $(J_s(p^r) - e_j)_j < y_j$. So we have $p'(y)_j \in \{1, 0, \geq\}$, which implies $p'(y^*)_j \in \{1, 0, \geq\}$ as well.

This finishes the proof. ◁

Claim 23 tells us that y^* could serve as the y for the **if** condition on line 5 for x and i. So we know that $p^+(x)_i \in \{1, \geq\}$. Furthermore, $p^+(x)_i = \geq$ only if $p'(x)_i \in \{0, \leq\}$, which implies $p^r(x)_i \in \{-1, 0, 1\}$.

We then prove the second part: for any point $x \in \mathcal{L}_s$ and $x \prec J_s(p^r)$, $p^r(x)_i = 1$ for some i with $x_i < J_s(p^r)_i$. Fix arbitrarily a slice s and a point $x \in \mathcal{L}_s$ such that $x \prec J_s(p^r)$. Assume for the sake of contradiction that $p^r(x)_i \in \{-1, 0\}$ for all i with $x_i < J_s(p^r)_i$. Then construct a new slice s' as follows:

$$s'_i \coloneqq \begin{cases} s_i & s_i \neq *; \\ x_i & s_i = * \text{ and } x_i = J_s(p^r)_i; \\ * & \text{otherwise } (s_i = * \text{ and } x_i < J_s(p^r)_i). \end{cases}$$

Then clearly $x, J_s(p^r) \in \mathcal{L}_{s'}$ and $x_i < (J_s(p^r))_i$ for all i with $s'_i = *$. Note that $p^r(J_s(p^r))_i \in \{\geq, 0\}$ for all i with $s'_i = *$. However, we have $p^r(x)_i \in \{-1, 0\}$ for all i with $s'_i = *$ by assumption. This means $J_{s'}(p^r) \not\preceq M_{s'}(p^r)$, which leads to a contradiction.

This finishes the proof. ◀

Again, symmetrically, we conclude the following lemma.

▶ **Lemma 24** (Safety Preserving of Subroutine 4)**.** *Given a monotone and safe PI function $p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}$, a point q and a coordinate i such that $p(q)_i \in \{\leq, \diamond\}$, we have the PI function p^r returned by* `Generate-PI-Function-Minus`*(p, q, i) remains safe.*

Before proving the analogs for `Generate-PI-Function-Zero`, we first derive a simple but crucial characterization for any 1-dimensional slice s from the safety.

▷ Claim 25. Given a monotone and safe PI function p, and any 1-dimensional slice s with its free coordinate j, we have

- $p(x)_j = 1$ for all $x \in \mathcal{L}_s$ and $x \prec J_s$; and
- $p(x)_j = -1$ for all $x \in \mathcal{L}_s$ and $x \succ M_s$.

In addition, if $J_s = M_s$ then $p(J_s)_j = p(M_s)_j = 0$; otherwise $(J_s \prec M_s)$, we have

- $p(x)_j = \diamond$ for all $J_s \prec x \prec M_s$; and
- $p(J_s)_j = \geq$ and $p(M_s)_j = \leq$.

Proof. Note that in 1-dimensional slice, for any point $x \in \mathcal{L}_s$, $x \not\succeq J_s$ is actually equivalent to $x \prec J_s$. So by the first item of the Definition 14, we have $p(x)_j = 1$ for all $x \in \mathcal{L}_s$ and $x \prec J_s$. Symmetrically, we also have $p(x)_j = -1$ for all $x \in \mathcal{L}_s$ and $x \succ M_s$.

Given that $J_s \preceq M_s$ by Lemma 17, we divide the proof into two simple cases.

Case 1: $J_s = M_s$. Note that by Proposition 13, we have $p(J_s)_j \in \{0, \geq\}$ and $p(M_s)_j \in \{0, \leq\}$. Take the intersection then we have $p(J_s)_j = p(M_s)_j = 0$;

Case 2: $J_s \prec M_s$. Given s is a 1-dimensional slice and $J_s(p) \prec M_s(p)$, for any point $J_s(p) \prec x \prec M_s(p)$, the only way that is consistent with the definition of $J_s(p)$ and $M_s(p)$ is to have $p(x)_j = \diamond$.

Then we move to $p(J_s)_j$. By Proposition 13, we have $p(J_s)_j \in \{0, \geq\}$. Since $J_s \prec M_s$, we know that $p(J_s)_j = \geq$. Symmetrically, we have $p(M_s)_j = \leq$. ◁

Now we are ready to present the analogs for `Generate-PI-Function-Zero`.

▶ **Lemma 26** (Monotonicity and Safety Preserving of Subroutine 5)**.** *Given a monotone and safe PI function $p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}$, a point q, and a coordinate ℓ such that $p(q)_\ell \in \{\leq, \geq, \diamond\}$, we have the PI function p^r returned by* `Generate-PI-Function-Zero`*(p, q, ℓ) remains monotone and safe. Furthermore, we have $p^r \Rightarrow p$.*

Proof. We first prove two easy cases.

Case 1: $p(q)_\ell = \geq$. We note that in this case, line 2 (the call of `Generate-PI-Function-Plus`) will be skipped, since we have either $q_\ell = 1$ or $p(q - e_\ell)_\ell = 1$ given p is safe. So when we run line 3, *either* it is also skipped then nothing gets changed *or* this lemma can be deduced directly by Lemma 21 and Lemma 24, whose conditions can be verified easily.

Case 2: $p(q)_\ell = \leq$. This case follows from a similar reason. It is easy to show line 3 (the call of `Generate-PI-Function-Minus`) will be skipped and this lemma can be deduced directly by Lemma 19 and Lemma 22.

The following claim essentially proves the last trickier case.

▷ Claim 27. Suppose that we are given a monotone and safe PI function $p : [n]^k \to \{-1, 0, 1, \leq, \geq, \diamond\}$, a point q and a coordinate ℓ such that $1 < q_\ell < n$ and $p(q)_\ell = \diamond$. Let p^r be the PI function returned by `Generate-PI-Function-Plus`$(p, q - e_\ell, \ell)$, then we have $p^r(q)_\ell = \geq$ (so that $p^r(q + e_\ell)_\ell \in \{\leq, \diamond\}$).

Proof. Note that $p'(q)_\ell = \geq$ at the end of line 2. So it suffices for us to prove that $p^+(q)_\ell \neq 1$ at the end of **for** loop.

Assume that $p^+(q)_\ell = 1$ for the sake of contradiction. Then we know there exists y such that (a) $q \preceq y$; (b) $q_\ell < y_\ell$; and (c) $q_j = y_j$ for all j with $p'(y)_j \notin \{1, 0, \geq\}$. Since $q_\ell < y_\ell$, we have $p'(y)_j = p(y)_j$ for all j. So we have (a) $q \preceq y$; (b) $q_\ell < y_\ell$; and (c) $q_j = y_j$ for all j with $p(y)_j \notin \{1, 0, \geq\}$. Define the slice s as follows:

$$s_j := \begin{cases} y_j & q_j = y_j; \\ * & \text{otherwise}. \end{cases}$$

Then we have $q, y \in \mathcal{L}_s$ and $y \in \mathtt{Post}_s(p)$. Since $q \preceq y$ and $q_\ell < y_\ell$, by the first property in Definition 14, we know that $p(q)_\ell \neq \diamond$, which contradicts the condition that $p(q)_\ell = \diamond$.

This finishes the proof. ◁

Case 3: $p(q)_\ell = \diamond$.

This implies that $1 < q_\ell < n$. At the end of line 2, by Lemma 19 and Lemma 22, we have p^r remains monotone and safe. Furthermore, $p^r \Rightarrow p$. Now by Claim 27, we know that $p^r(q)_\ell = \geq$, which means $p^r(q + e_\ell) \in \{\leq, \diamond\}$ by Claim 25.

So at the end of line 3, by Lemma 21 and Lemma 24 (which need the condition of $p^r(q + e_\ell) \in \{\leq, \diamond\}$), we have p^r remains monotone and safe. Furthermore, $p^r \Rightarrow p$.

This finishes the proof. ◀

▶ **Lemma 28.** *Given a monotone and safe PI function p, a point $q \in [n]^k$, a coordinate $\ell \in [k]$, and $b \in \{-1, 0, 1\}$ such that $(p(q)_\ell, b)$ satisfies the following condition:*

$$p(q)_\ell \in \{\geq, \diamond\} \text{ if } b = 1; \;\; p(q)_\ell \in \{\leq, \diamond\} \text{ if } b = -1; \;\; p(q)_\ell \in \{\leq, \geq, \diamond\} \text{ if } b = 0,$$

the function p^r returned by `Generate-PI-Function`(p, q, ℓ, b) *satisfies $p^r(q)_\ell = b$.*

Proof. When $b = 1$, we have $p'(q)_\ell = 1$ at the end of line 2. Since $p^r \Rightarrow p'$, we have $p^r(q)_\ell = 1$ as well. The case of $b = -1$ is similar.

If $q_\ell = 1$, $q_\ell = n$, or $p(q)_\ell \neq \diamond$, then $p^r(q)_\ell = 0$ can be derived by previous cases since at most one of `Generate-PI-Function-Plus` and `Generate-PI-Function-Minus` is called. For the case $1 < q_\ell < n$ and $p(q)_\ell = \diamond$, by Claim 27, we have both `Generate-PI-Function-Plus` and `Generate-PI-Function-Minus` are called and $p^r(q - e_\ell)_\ell = 1$ and $p^r(q - e_\ell)_\ell = -1$, which forces $p^r(q)_\ell = 0$ since p is monotone. ◀

3.4 Not Creating New Fixed Points

▶ **Lemma 29** (Fixed Points of Subroutine 3)**.** *Given a monotone function $f : [n]^k \to [n]^k$, a PI function p, a point q and a coordinate ℓ such that*

- *p is monotone and safe;*
- *$p(q)_\ell \in \{\geq, \diamond\}$;*
- *$f|_p(q + e_\ell)_\ell \geq q_\ell + 1$; and*
- *$\mathtt{Fix}(f|_p) \subseteq \mathtt{Fix}(f)$,*

the function p^r returned by `Generate-PI-Function-Plus`(p, q, ℓ) *satisfies* $\mathtt{Fix}(f|_{p^r}) \subseteq \mathtt{Fix}(f|_p) \subseteq \mathtt{Fix}(f)$.

Proof. Note that whenever we have $p^r(x)_i \neq p(x)_i$ for some x and i, it must be the case that $p^r(x)_i = 1$ or $p^r(x)_i = p(x)_i \cap \geq$. If $p^r(x)_i = 1$, then we have $f|_{p^r}(x) \neq x$, which means x is not a fixed point of $f|_{p^r}$. So we only need to analyze the case that $p(x)_i \neq p^r(x)_i = p(x)_i \cap \geq$.

Fix arbitrary z and i such that $p(z)_i \neq p^r(z)_i = p(z)_i \cap \geq$. First consider the updating rule on line 2, in which case $i = \ell$ and $z = x + e_\ell$ for some $x \succeq q$ and $x_\ell = q_\ell$, then we have $f|_p(z)_\ell \geq z_\ell$ by the third condition. Note that it suffices for us to know $f|_p(x)_i \geq x_i$, since it implies that either $f|_p(x)_i = f|_{p^r}(x)_i$ or $f|_{p^r}(x)_i = x_i + 1$ since $p^r(x)_i \in \{0, 1, \geq\}$ given that $p^r(x)_i = p(x)_i \cap \geq$.

Next, we consider the case that $p^+(z)_i$ is updated on line 6 and 7, where we will show $f|_{p^r}(x) \neq x$. Let y be such that (a) $z \preceq y$; (b) $z_i - 1 < y_i$ ($z_i \leq y_i$); and (c) $z_j = y_j$ for all j with $p(y)_j \notin \{1, 0, \geq\}$ on line 5. Define a slice s as follows:

$$s_j := \begin{cases} y_j & p(y)_j \notin \{1, 0, \geq\}; \\ * & \text{otherwise.} \end{cases}$$

Then we have $z, y \in \mathcal{L}_s$ and $z \preceq J_s(p)$ (given that $z \preceq y$ and $y \preceq J_s(p)$). Further note that $z \neq J_s(p)$, otherwise we have $p(z)_i = p^r(z)_i = p(z)_i \cap \geq$. So it suffices for us to argue $f|_p(z) \neq z$ for all $z \prec J_s(p)$, which follows from that p is safe and Lemma 17.

This finishes the proof. ◀

We conclude the analog for `Generate-PI-Function-Minus`.

▶ **Lemma 30** (Fixed Points Preserving of Subroutine 4). *Given a monotone function $f : [n]^k \to [n]^k$, a PI function p, a point q and a coordinate ℓ such that*

- *p is monotone and safe;*
- *$p(q)_\ell \in \{\leq, \diamond\}$;*
- *$f|_p(q - e_\ell)_\ell \leq q_\ell - 1$; and*
- *$\texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$,*

the function p^r returned by `Generate-PI-Function-Minus`(p, q, ℓ) *satisfies* $\texttt{Fix}(f|_{p^r}) \subseteq \texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$.

▶ **Lemma 31** (Fixed Points of Subroutine 5). *Given a monotone function $f : [n]^k \to [n]^k$, a PI function p, a point q and a coordinate ℓ such that*

- *p is monotone and safe;*
- *$p(q)_\ell \in \{\leq, \geq, \diamond\}$;*
- *$f|_p(q)_\ell = q_\ell$; and*
- *$\textit{Fix}(f|_p) \subseteq \textit{Fix}(f)$,*

the function p^r returned by `Generate-PI-Function-Zero`(p, q, ℓ) *satisfies* $\textit{Fix}(f|_{p^r}) \subseteq \textit{Fix}(f|_p) \subseteq \textit{Fix}(f)$.

Proof. Let us consider the non-trivial case where both subroutines `Generate-PI-Function-Plus` and `Generate-PI-Function-Minus` are called. Otherwise, this lemma can be derived by either Lemma 29 or Lemma 30 (given that $f|_p(q)_\ell = q_\ell$).

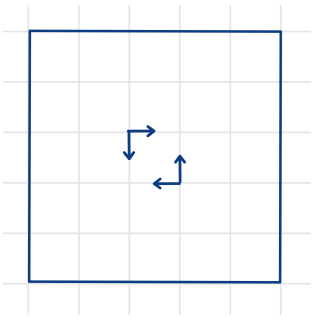
Suppose that both subroutines are called, then we have $1 < q_\ell < n$ and $p(q)_\ell = \diamond$. Since $f|_p(q)_\ell = q_\ell$, we have $f(q)_\ell = q_\ell$.

By Claim 27, we know that at the end of line 2, we have $p^r(q)_\ell = \geq$ and $p^r(q + e_\ell)_\ell \in \{\leq, \diamond\}$. At this time, we still have $f|_{p^r}(q)_\ell = q_\ell$ as well as other properties in the condition of this lemma by Lemmas 19, 22, and 29. So this lemma can be derived by Lemmas 21, 24, and 30.

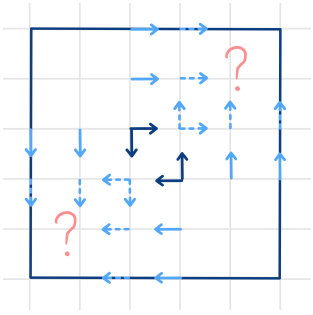
This finishes the proof. ◀

4 An Illustrating Example

In this section, we illustrate how our reduction works in one concrete but tricky example. Recall that we have to make sure our Algorithm 1 works for *any* monotone function and *any* algorithm solving UniqueTarski. For simplicity, we pick the following 2D example: a monotone function $f : [6]^2 \to [6]^2$ with $f(3,4) = (4,3)$ and $f(4,3) = (3,4)$ as shown in Figure 2a, as well as an algorithm $\mathcal{U}$ for UniqueTarski, which will first query $(3,4)$, given the answer $f(3,4) = (4,3)$, then query $(4,3)$.

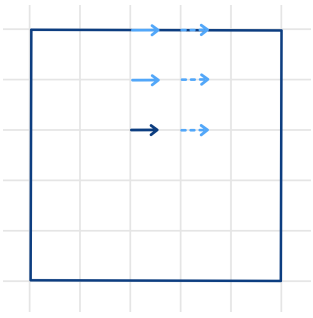


(a) A monotone function $f : [6]^2 \to [6]^2$ with $f(3,4) = (4,3)$ and $f(4,3) = (3,4)$.

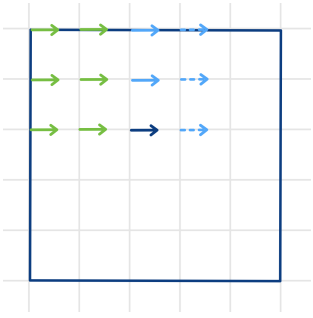


(b) The standard partial information derived by $(3,4)$ and $(4,3)$, described in the light blue color. The solid arrows mean -1 or 1 and the dashed arrows mean $\leq$ or $\geq$ (the same rule applies below).

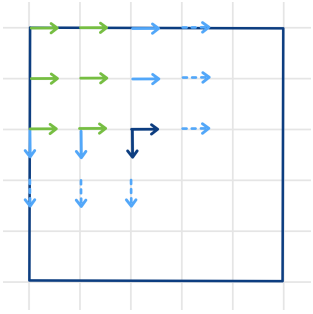
Figure 2 A 2D example for which after two queries the algorithm $\mathcal{U}$ for UniqueTarski will fail.



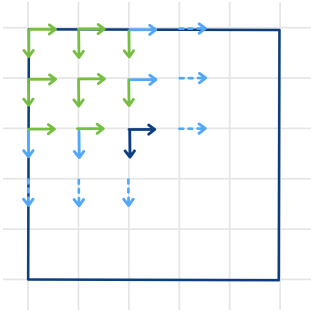
(a) The partial information by adding $f(3,4)_1 = 4$.



(b) The safe PI function constructed by Algorithm 1. The new information is described in the green color (the same rule applies below).

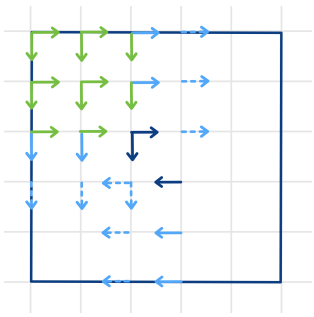


(c) The partial information by adding $f(3,4)_2 = 3$.

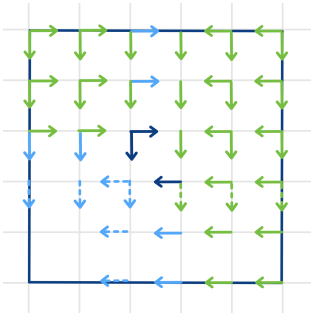


(d) The safe PI function constructed by Algorithm 1.

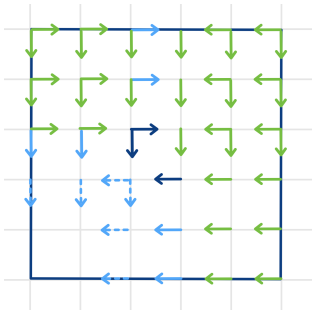
Figure 3 The evolution of PI function when adding $f(3,4)_1 = 4$ and $f(3,4)_2 = 3$.



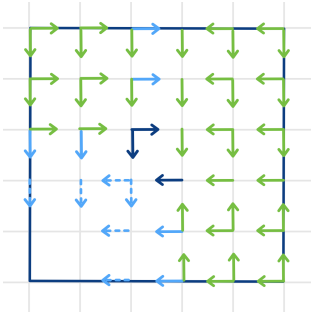
(a) The partial information by adding $f(4,3)_1 = 3$.



(b) The safe PI function constructed by Algorithm 1.



(c) The partial information by adding $f(4,3)_2 = 4$.



(d) The safe PI function constructed by Algorithm 1.

Figure 4 The evolution of PI function when adding $f(4,3)_1 = 3$ and $f(4,3)_2 = 4$.

Note that the function (actually partial function) in Figure 2a does not violate monotonicity. But clearly, no monotone function that is consistent with Figure 2a has a unique fixed point. This is because the partial information derived from $f(3,4) = (4,3)$ and $f(4,3) = (3,4)$ is sufficient to conclude the existence of fixed points in both the bottom left corner and top right corner, as shown in Figure 2b. Observe that if the algorithm $\mathcal{U}$ is not fooled, it could immediately reject the function f and return "the underlying function has multiple fixed points" once it gets the true answer $f(4,3) = (3,4)$.

Perhaps surprisingly, our reduction will modify the answer the algorithm $\mathcal{U}$ gets when querying $(3,4)$, by creating safe PI functions p that satisfy $\texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$ (the formal statement is in Theorem 15).

We show how the PI function evolves step by step in Figure 3 and 4. The figures on the left-hand side are obtained by adding one piece of information (namely, $f(3,4)_1 = 4$, $f(3,4)_2 = 3$, $f(4,3)_1 = 3$, and $f(4,3)_2 = 4$). The figures on the right-hand side are obtained by the Subroutine `Generate-PI-Function`. Note that in the last step after Figure 4b, we will try to add the last piece of information $f(4,3)_2 = 4$. However, since $p(4,3)_2 =\leq$ already, the algorithm $\mathcal{U}$ will get $f|_p(4,3)_2 = 3$.

It is easy to verify that all PI functions of the figures on the right-hand side are safe and satisfy $\texttt{Fix}(f|_p) \subseteq \texttt{Fix}(f)$. In particular, for Figure 4d, every point outside the bottom left corner is certainly not a fixed point of $f|_p$, and the fixed point(s) in the bottom left corner is not affected.

5 Promise Problem versus TFNP Version

The problems $\textsc{Tarski}(n,k)$ and $\textsc{UniqueTarski}(n,k)$ are *promise problems*. In the former, we want to compute a fixed point of the given function under the promise (condition) that it is monotone; in the latter the function is promised to be monotone and have a unique fixed point.

From a promise problem, one can define a total search problem, where on any given arbitrary input one seeks either a desired solution as in the promise problem, or a violation certificate showing that the input does not satisfy the promise. The total search version of the Tarski problem is formally the following search problem.

▶ **Definition 32** (Total search version of $\textsc{Tarski}(n,k)$)**.** *Given a function $f : [n]^k \to [n]^k$, find one of the following:*

- *a point $x \in [n]^k$ such that $f(x) = x$; or*
- *two points $x, y \in [n]^k$ such that $x \preceq y$ and $f(x) \not\preceq f(y)$.*

In the black box setting, the function f is given by a black box (an oracle). In the white box setting, the function f is given by a poly$(k, \log n)$-size circuit C with $k * \lceil \log n \rceil$ input gates and $k * \lceil \log n \rceil$ output gates.

The total search version of Tarski in the white box setting is in TFNP, in fact it is PLS ∩ PPAD. Any algorithm for the total search version of a promise problem (whether in the white box or the black box setting) can be obviously used also to solve the promise problem, so the total version is always at least as hard as the promise problem. In general the converse may not hold, since in the total search version, the algorithm is not allowed to simply fail if the input does not satisfy the promise, but it must provide a violation certificate (and in general the complexity of the total problem may depend on the type of certificate that is required). In the case of the Tarski problem in the black box setting however it is easy to see that the total version is no harder than the promise problem. This is because of the following property.

▶ **Lemma 33.** *Let $Q = \{q^1, \ldots, q^m\}$ be a set of query points in $[n]^k$ and $A = \{a^1, \ldots, a^m\}$ the corresponding answers of the black box. There is a monotone function f that is consistent with all the answers (i.e such that $f(q^i) = a^i$ for all $i \in [m]$) if and only if there is no pair i, j such that $q^i \preceq q^j$ and $a^i \not\preceq a^j$.*

Proof. If there is a pair i, j such that $q^i \preceq q^j$ and $a^i \not\preceq a^j$ then clearly there is no monotone function f that is consistent with the answers. Suppose now that there is no such pair. Define the function f as follows: For every point $x \in [n]^k$ and every coordinate i, set $f(x)_i = \min\{a^j_i \mid x \preceq q^j\}$; if the set on the right-hand side is empty then set $f(x)_i = n$. We have to show that f is monotone and is consistent with the answers.

Consider any two points $x \preceq y$ and any coordinate i. Then $y \preceq q^j$ implies $x \preceq q^j$, thus $f(x)_i = \min\{a^j_i \mid x \preceq q^j\} \leq f(y)_i = \min\{a^j_i \mid y \preceq q^j\}$. Therefore, f is monotone.

By the definition of f, for any query point q^t and coordinate i, $f(q^t)_i = \min\{a^j_i \mid q^t \preceq q^j\} \leq a^t_i$. If $f(q^t)_i < a^t_i$ then there is another query point q^j such that $q^t \preceq q^j$ and $a^j_i < a^t_i$, hence $a^t \not\preceq a^j$. ◀

▶ **Corollary 34.** *In the black-box setting, suppose that $\textsc{Tarski}(n,k)$ (the promise problem) can be solved in $q(n,k)$ queries and $t(n,k)$ time, then total search version of $\textsc{Tarski}(n,k)$ can be solved in $q(n,k)$ queries and $O(t(n,k) + q(n,k)^2 \cdot k)$ time.*

Proof. Run the algorithm for the promise problem. Either the algorithm will find a fixed point within the query and time complexity of the promise problem, or two of the query points provide a violation certificate. ◀

We showed that TARSKI(n,k) reduces to UNIQUETARSKI(n,k) with the same query complexity. Therefore, we have.

▶ **Corollary 35.** *Any black-box algorithm for* UNIQUETARSKI(n,k) *(the promise problem) can be used to solve also the total search version of* TARSKI(n,k) *with the same query complexity.*

We can define a total search version of UNIQUETARSKI(n,k) that includes as a possible answer also a violation certificate of uniqueness. One way to define it is as follows.

▶ **Definition 36** (Total search version of UNIQUETARSKI(n,k))**.** *Given a function* $f : [n]^k \to [n]^k$*, find one of the following:*

- *a point* $x \in [n]^k$ *such that* $f(x) = x$*; or*
- *two points* $x, y \in [n]^k$ *such that* $x \preceq y$ *and* $f(x) \not\preceq f(y)$*; or*
- *two points* $x, y \in [n]^k$ *such that* $x \preceq f(x)$*,* $y \succeq f(y)$ *and* $x \not\preceq y$*.*

In the black box setting, the function f *is given by a black box (an oracle). In the white box setting, the function* f *is given by a poly*$(k, \log n)$*-size circuit* C *with* $k * \lceil \log n \rceil$ *input gates and* $k * \lceil \log n \rceil$ *output gates.*

Note that if f is monotone and $x \preceq f(x)$ then f has a fixed point in $\mathcal{L}_{x,n^k}$, and if $y \succeq f(y)$ then f has a fixed point in $\mathcal{L}_{1^k,y}$. If $x \not\preceq y$ then $\mathcal{L}_{1^k,y}$ and $\mathcal{L}_{x,n^k}$ are disjoint, and hence f has at least two fixed points. Clearly, the total search version of TARSKI(n,k) is at least as hard as that of UNIQUETARSKI(n,k), both in the white box and the black box setting, since the latter includes one more option for an acceptable output. It is not much harder however. Let T-TARSKI(n,k) and T-UNIQUETARSKI(n,k) denote the total search versions of the two problems, as defined above.

▶ **Theorem 37.** *If* T-UNIQUETARSKI(n,k) *can be solved in* $q(n,k)$ *queries in the black box setting, then* T-TARSKI(n,k) *can be solved in* $q(n,k)$ *queries. If* T-UNIQUETARSKI(n,k) *can be solved in time* $t(n,k)$ *in the black box (respectively, white box) setting, then* T-TARSKI(n,k) *can be solved in time* $O(t(n,k) * (k \cdot \log n))$ *in the black box (resp. white box) setting.*

Proof. The statement in the first sentence follows from Corollary 35. Next, we show the statement in the second sentence.

Given a black-box or white-box algorithm $\mathcal{U}$ for T-UNIQUETARSKI(n,k), the algorithm for T-TARSKI(n,k) in the same setting is as follows. Use the algorithm $\mathcal{U}$ to find a solution of T-UNIQUETARSKI(n,k). If the solution is a fixed point (i.e., a point $x \in [n]^k$ such that $f(x) = x$) or a violation certificate of monotonicity (i.e., two points $x, y \in [n]^k$ such that $x \preceq y$ and $f(x) \not\preceq f(y)$) then we are done, since they are also solution of T-TARSKI(n,k). Otherwise, we find a solution that is a violation certificate of uniqueness (i.e., two points $x, y \in [n]^k$ such that $x \preceq f(x)$, $y \succeq f(y)$ and $x \not\succeq y$). Then there exists i such that $x_i > y_i$, which means either $x_i > n/2$ or $y_i \leq n/2$. If $x_i > n/2$, then we shrink the search space to $\mathcal{L}_{x,n^k}$ and recursively call $\mathcal{U}$ to find a solution in $\mathcal{L}_{x,n^k}$; If $y_i \leq n/2$, then we shrink the search space to $\mathcal{L}_{1^k,y}$ and recursively call $\mathcal{U}$ to find a solution in $\mathcal{L}_{1^k,y}$. The function f may map a point q in the reduced space to a point outside the space; in that case the point q together with either the top or the bottom point of the reduced space form a violation certificate for monotonicity. In the black box setting, if the algorithm ever queries such a point q then we immediately get a violation of monotonicity and can terminate. In the white box setting,

when we recurse to the reduced space, we replace the circuit for f with a modified circuit for a function f' which restricts the coordinates of the output point to lie in the reduced space. When the recursive call returns a solution to T-TARSKI for the reduced space, i.e. either a fixed point x of f' or a pair of points x, y that certify that f' is not monotone, then we test if f and f' have the same value on these points. If they do, then they constitute a solution for f in the original space; if one of them does not, then that point with the bottom or the top element provide a certificate for the violation of monotonicity of f.

The search space goes down by a factor of two after each call of $\mathcal{U}$. So after at most $k \cdot \log n$ many rounds, we can find a solution of T-TARSKI(n, k). ◀

6 Discussion and Open Problems

Our results resolve an open question in [4] and could potentially shed new light on the upper bounds and lower bounds of the query complexity of TARSKI(n, k). As we showed, TARSKI(n, k) is no harder, with respect to query complexity, than the special case of monotone functions that have a unique fixed point in the lattice, and even further, have a unique fixed point in every slice of the lattice. There is a lot of structure in such monotone functions. In a function f with a unique fixed point, the least fixed point and the greatest fixed point coincide. There is a path connecting the bottom element 1^k of the lattice with the top element n^k, the fixed point lies on this path, and the function f on all points in this path point in the direction of the fixed point. The same structure holds for every slice if the function has a unique fixed point on all slices. This structure may well be useful in helping to design an algorithm with low complexity. On the lower bound side, it may also provide a useful framework; indeed the lower bound constructions for two dimensions in [4] use this structure. Can we use uniqueness to improve the bounds on the query complexity of Tarski?

A second question concerns the time complexity of the algorithms in the black box setting. Our reduction involves the maintenance of a partial information function p that is defined over the whole lattice. A straightforward implementation would take of course exponential time. Note however that we do not need to compute p on the whole domain; we only need to be able to compute p on demand on specific points, namely the query points of the Unique Tarski algorithm. Is it possible to implement the algorithm so that it runs in polynomial time in the number of queries? More generally, does the black-box time complexity of TARSKI(n, k) reduce also to that of UNIQUETARSKI(n, k)?

Regarding the white-box complexity, we know that the total search version of TARSKI(n, k) is in PLS ∩ PPAD [4] and thus by the results of [5, 8], it is in the classes CLS (Continuous-Local-Search) and EOPL (End-of-Potential-Line). Is the total search version of UNIQUETARSKI in the class UEOPL (Unique-EOPL) [6]? Is it hard for UEOPL?

References

1 Xi Chen and Yuhao Li. Improved upper bounds for finding tarski fixed points. In *Proceedings of the 23rd ACM Conference on Economics and Computation*, pages 1108–1118, 2022.

2 Anne Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992.

3 Chuangyin Dang, Qi Qi, and Yinyu Ye. Computational models and complexities of tarski's fixed points. Technical report, Stanford University, 2011.

4 Kousha Etessami, Christos Papadimitriou, Aviad Rubinstein, and Mihalis Yannakakis. Tarski's theorem, supermodular games, and the complexity of equilibria. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

5 John Fearnley, Paul W. Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: CLS = PPAD ∩ PLS. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 46–59. ACM, 2021.

6 John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. Unique end of potential line. *J. Comput. Syst. Sci.*, 114:1–35, 2020.

7 John Fearnley, Dömötör Pálvölgyi, and Rahul Savani. A faster algorithm for finding tarski fixed points. *ACM Transactions on Algorithms (TALG)*, 18(3):1–23, 2022.

8 Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. Further collapses in TFNP. In *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 33:1–33:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

9 Massimo Marinacci and Luigi Montrucchio. Unique tarski fixed points. *Math. Oper. Res.*, 44(4):1174–1191, 2019. doi:10.1287/moor.2018.0959.

10 Paul Milgrom and John Roberts. Rationalizability, learning, and equilibrium in games with strategic complementarities. *Econometrica: Journal of the Econometric Society*, pages 1255–1277, 1990.

11 L. Shapley. Stochastic games. *Proc. Nat. Acad. Sci.*, 39(10):1095–1100, 1953.

12 Alfred Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific journal of Mathematics*, 5(2):285–309, 1955.

13 Donald M Topkis. Equilibrium points in nonzero-sum n-person submodular games. *Siam Journal on control and optimization*, 17(6):773–787, 1979.

14 Donald M Topkis. *Supermodularity and Complementarity*. Princeton University Press, 1998.

A Distribution Testing Oracle Separating QMA and QCMA

Anand Natarajan
Massachusetts Institute of Technology, Cambridge, MA, USA

Chinmay Nirkhe
IBM Quantum, Cambridge, MA, USA

Abstract

It is a long-standing open question in quantum complexity theory whether the definition of *non-deterministic* quantum computation requires quantum witnesses (QMA) or if classical witnesses suffice (QCMA). We make progress on this question by constructing a randomized classical oracle separating the respective computational complexity classes. Previous separations [3, 13] required a quantum unitary oracle. The separating problem is deciding whether a distribution supported on regular un-directed graphs either consists of multiple connected components (yes instances) or consists of one expanding connected component (no instances) where the graph is given in an adjacency-list format by the oracle. Therefore, the oracle is a distribution over n-bit boolean functions.

2012 ACM Subject Classification Theory of computation → Quantum computation theory

Keywords and phrases quantum non-determinism, complexity theory

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.22

Related Version *Full Version*: arXiv:2210.15380 [22]

Acknowledgements We thank Srinivasan Arunachalam, Andrew Childs, Elizabeth Crosson, Yi-Kai Liu, Aram Harrow, Zhiyang He, Robin Kothari, William Kretschmer, Yupan Liu, Kunal Marwaha, Mehdi Soleimanifar, Umesh Vazirani, and Elizabeth Yang for helpful discussions. Some of the early ideas of this result were developed while Chinmay Nirkhe was at the University of California, Berkeley. This work was partially completed while both authors were participants in the Simons Institute for the Theory of Computing program *The Quantum Wave in Computing: Extended Reunion*.

1 Introduction

There are two natural *quantum* analogs of the computational complexity class NP. The first is the class QMA in which a quantum polynomial-time decision algorithm is given access to a $\mathsf{poly}(n)$ *qubit* quantum state as a witness for the statement. This class is captured by the QMA-complete local Hamiltonian problem [17] in which the quantum witness can be interpreted as the ground-state of the local Hamiltonian. The second is the class QCMA in which the quantum polynomial-time decision algorithm is given access instead to a $\mathsf{poly}(n)$ *bit classical* state. While it is easy to prove that $\mathsf{QCMA} \subseteq \mathsf{QMA}$ as the quantum witness state can be immediately measured to yield a classical witness string, the question of whether $\mathsf{QCMA} \stackrel{?}{=} \mathsf{QMA}$, first posed by Aharonov and Naveh [4], remains unanswered. If $\mathsf{QCMA} = \mathsf{QMA}$, then every local Hamiltonian would have an efficient classical witness of its ground energy; morally, this can be thought of as an efficient classical description of its ground state. The relevance of local Hamiltonians to condensed matter physics makes this question a central open question in quantum complexity theory [2].

Because $\mathsf{P} \subseteq \mathsf{QCMA} \subseteq \mathsf{QMA} \subseteq \mathsf{PSPACE}$, any unconditional separation of the two complexity classes would imply $\mathsf{P} \neq \mathsf{PSPACE}$ and seems unlikely without remarkably ingenious new tools. A more reasonable goal is an oracle separation between the two complexity classes. The first oracle separation, by Aaronson and Kuperberg [3], showed that there exists a black-box unitary problem for which quantum witnesses suffice and yet no polynomial sized classical witness and algorithm can solve the problem with even negligible success probability. A second black-box separation was discovered a decade later by Fefferman and Kimmel [13]. The Fefferman and Kimmel oracle is a completely positive trace perseving (CPTP) map called an "in-place" permutation oracle. Both oracles [3, 13] are inherently quantum[1]. Whereas, the "gold-standard" of oracle separations – namely black-box function separations (also known as classical oracle separations) – only require access to a *classical function* that can be queried in superposition[2].

1.1 Graph oracles

The major result of this work is to prove that there exists a distribution over black-box function problems separating QMA and QCMA. Each black-box function corresponds to the adjacency list of a $N \stackrel{\text{def}}{=} 2^n$ vertex constant-degree colored graphs[3] $G = (V, E)$. Roughly speaking, a graph is a YES instance if the second eigenvalue of its normalized adjacency matrix is 1 (equivalently, if it has at least two connected components) and a graph is a NO instance if it second eigenvalue is at most $1 - \alpha$ for some fixed constant α (equivalently, the graph has one connected component and is expanding). We call this problem the *expander distinguishing problem*.

Distribution oracles

A distribution over functions (equivalently, a distribution over graphs) is a YES instance if it is entirely supported on YES graphs and a distribution over functions is a NO instance if it is entirely supported on NO graphs.

In this work, we construct, for every n, families of YES and NO distributions over graphs such that following hold for the promise problem of distinguishing a graph sampled from a YES distribution from a graph sampled from a NO distribution.

1. There is a QMA proof system that solves this problem, where the verifier runs in quantum polynomial time and has black-box query access to the sampled graph, and the honest prover's (quantum) witness depends only on the distribution, not on the specific sample.
2. No QCMA proof system can solve this problem, provided the prover's (classical) witness is only allowed to depend on the distribution, and not on the sample.

Our work is not the first to consider oracles that sample from distributions over functions. The in-place oracle separation of [13] between QMA and QCMA used oracles that sampled random permutations. For a somewhat different problem, of separating bounded-depth

[1] It might be reasonable to wonder if the unitary oracles can be converted into classical oracles by providing oracle access to the exponentially long classical descriptions of the respective matrices. This is not known to be true because it is unclear how to use access to the classical description to solve the QMA problem.

[2] One reason this model is natural is that if we were given a circuit of size C to implement this classical function, then we would automatically get a quantum circuit of size C to implement the oracle, simply by running the classical circuit coherently. This is not true for the "in-place" permutation oracle model, assuming that one-way functions exist.

[3] A similar problem was previously conjectured to be an oracle separation for these complexity classes by Lutomirski [21].

Table 1 List of known oracle separations.

Authors	Separating black box object	Proof techniques used
Aaronson & Kuperberg [3]	n-qubit unitaries	Adversary method
Fefferman & Kimmel [13]	n-qubit CPTP maps	Combinatorial argument, Adversary method
This work	Distributions over n-bit boolean functions	Combinatorial argument, Adversary method, Polynomial method
Conjectured	n-bit boolean function	?

quantum-classical circuits, [8] introduced a related notion called a "stochastic oracle" – the main difference between this and our model is that a stochastic oracle resamples an instance every time it is queried.

Comparison with previous oracle separations between QMA and QCMA

Table 1 summarizes our work in relation to previous oracle separations. In terms of results, we take a further step towards the standard oracle model – all that remains is to remove the randomness from our oracle. In terms of techniques, we combine the use of counting arguments and the adversary method from previous works with a BQP lower bound for a similar graph problem, due to [6]. This lower bound was shown using the polynomial method. We view the judicious combination of these lower bound techniques – as simple as it may seem – as one of the conceptual contributions of this paper.

Intuition for hardness

The expander distinguishing problem is a natural candidate for a separation between QMA and QCMA because it is an "oracular" version of the sparse Hamiltonian problem, which is complete for QMA [10, Problem H-4]. To see this, we recall some facts from spectral graph theory. The top eigenvalue of the normalized adjacency matrix A for regular graphs is always 1 and the uniform superposition over vertices is always an associated eigenvector. If the graph is an expander (the NO case of our problem), the second eigenvalue is bounded away from 1, but if the graph is disconnected (the YES case of our problem), then the second eigenvalue is exactly 1. Thus, our oracle problem is exactly the problem of estimating the minimum eigenvalue of $\mathbb{I} - A$ (a sparse matrix for a constant-degree graph), on the subspace orthogonal to the uniform superposition state. Viewing $\mathbb{I} - A$ as a sparse Hamiltonian, we obtain the connection between our problem and the sparse Hamiltonian problem.

One reason to show oracle separations between two classes is to provide a *barrier* against attempts to collapse the classes in the "real" world. We interpret our results as confirming the intuition that any QCMA protocol for the sparse Hamiltonian must use more than just black-box access to entries of the Hamiltonian: it must use some nontrivial properties of the ground states of these Hamiltonians. In this sense, it emulates the original quantum adversary lower bound of [9] which showed that any BQP-algorithm for solving NP-complete problems must rely on some inherent structure of the NP-complete problem as BQP-algorithms cannot solve unconstrained search efficiently.

Naturalness of the randomized oracle model

Some care must be taken whenever one proves a separation in a "nonstandard" oracle model – see for instance the "trivial" example in [1] of a randomized oracle separating MA_1 from MA. We believe that our randomized oracle model is natural for several reasons. Firstly, as mentioned above, randomization was used in the quantum oracle of [13] for essentially the

same reason: to impose a restriction on the witnesses received from the prover. Secondly, it is consistent with our knowledge that our oracle separates QMA from QCMA even when the randomness is removed (and indeed we conjecture this is the case, as described below.) Thirdly, the randomization still gives the prover access to substantial information about the graph: in particular, the prover knows the full connected component structure of the graph. As we show, this information is enough for the prover to give a *quantum* witness state, that in the YES case convinces the verifier with certainty. Our result shows that even given full knowledge of the component structure, the prover cannot construct a convincing classical witness – we believe this sheds light on how a QMA witness can be more powerful than a QCMA witness.

1.2 Overview of proof techniques

Quantum witnesses and containment in oracular QMA

A quantum witness for any YES instance graph is any eigenvector $|\xi\rangle$ of eigenvalue 1 that is orthogonal to the uniform superposition over vertices. The verification procedure is simple: project the witness into the subspace orthogonal to the uniform superposition over vertices, and then perform one step of a random walk along the graph, by querying the oracle for the adjacency matrix in superposition. Verify that the state after the walk step equals $|\xi\rangle$. This is equivalent to a 1-bit phase estimation of the eigenvalue. If a graph is a NO instance, then there does not exist any vector orthogonal to the uniform superposition (the unique eigenvector of value 1) that would pass the previous test.

Whenever, the graph has a connected component of $S \subsetneq V$, then an eigenvector orthogonal to the uniform superposition of eigenvalue 1 exists. When $|S| \ll N$, this eigenvector is very close to $|S\rangle$, the uniform superposition over basis vectors $x \in S$. Notice that this state only depends on the connected component S and not the specific edges of the graph. Furthermore, the state $|S'\rangle$ for any subset S' that approximates S forms a witness that is accepted with high probability.

Lower bound on classical witnesses

The difficulty in this problem lies in proving a *lower bound* on the ability for classical witnesses to distinguish YES and NO instances. To prove a lower bound, we argue that any quantum algorithm with access to a polynomial length classical witness must make an exponential number of (quantum) queries to the adjacency list of the graph in order to distinguish YES and NO instances. This, in turn, lower bounds the time complexity of any QCMA algorithm distinguishing YES and NO instances but is actually slightly stronger since we don't consider the computational complexity of the algorithm between queries.

Proving lower bounds when classical witnesses are involved is difficult because the witness could be based on any property of the graph. For example, the classical witness could describe cycles, triangles, etc. contained in the graph – while it isn't obvious why such a witness would be helpful, proving that any such witness is insufficient is a significant challenge. One way to circumvent this difficulty is to first show a lower bound *assuming* some structure about the witness[4], and then "remove the training wheels" by showing that the assumption holds for any good classical witness.

[4] Assuming structure about a witness is a common technique in theoretical computer science and in particular lower bounds for classical witnesses of quantum statements. For example, lower bounds against natural proofs [19]. Another example is the NLTS statement [7] which is about lower bounds for classical witnesses for the ground energy of a quantum Hamiltonian of a particular form: constant-depth quantum circuits.

Lower bound against "subset witnesses"

One structure we can assume is that the witness only depends on the set of vertices contained in the connected component S. This is certainly the case for the ideal quantum witness state. Our result shows that any polynomial-length witness only depending on the vertices in S requires an exponential query complexity to distinguish YES and NO instance graphs.

The starting point for this statement is the exponential query lower bound *in the absence of a witness* (i.e. for BQP) for the expander distinguishing problem proven by Ambainis, Childs and Liu [6], using the polynomial method. In [6], the authors define two distributions over constant-degree regular colored graphs: the first is a distribution P_1 over random graphs with overwhelming probability of having a second normalized eigenvalue at most $1 - \epsilon_0$. The second is a distribution P_ℓ over random graphs with overwhelming probability of having ℓ connected components. Since, almost all graphs in P_1 are NO graphs and all graphs in P_ℓ are YES graphs, any algorithm distinguishing YES and NO instances must be able to distinguish the two distributions. We first show that a comparable query lower bound still holds even when the algorithm is given a witness consisting of polynomially many random points F from any one connected component.

Next, we show that if there were a QCMA algorithm where the optimal witness depends only on the set of vertices S in one of the connected components, by a counting argument, there must exist a combinatorial *sunflower* of subsets S that correspond to the same witness string. A *sunflower*, in this context, is a set of subsets such that each subset contains a core $F \subset V$ and every vertex of $V \setminus F$ occurs in a small fraction of subsets. This implies that there exists a BQP algorithm which distinguishes YES instances corresponding to the sunflower from all NO instances. Next, we show using an adversary bound [5], a quantum query algorithm cannot distinguish the distribution of YES instances corresponding to the sunflower from the uniform distribution of YES instances such that the core F is contained in a connected component (the ideal sunflower).

This indistinguishability, along with the previous polynomial method based lower bound, proves that QCMA algorithm – whose witness only depends on the vertices in the connected component – for the expander distinguishing problem must make an exponential number of queries to the graph.

Removing the restriction over witnesses

Our proof, thus far, has required the restriction that the witness only depends on the vertices in the connected component. In some sense, this argues that there is an oracle separation between QMA and QCMA if the prover is restricted to being "near-sighted": it cannot see the intricacies of the edge-structure of the graph, but can notice the separate connected components of the graph. If the near-sighted prover was capable of sending quantum states as witnesses, then she can still aid a verifier in deciding the expander distinguishing problem, whereas if she could only send classical witnesses, then she cannot aid a verifier.

It now remains to remove the restriction that the witness can only depend on the vertices in the connected component. We do this by introducing *randomness* into the oracle, precisely designed to "blind" the prover to the local structure of the graph. In the standard oracle setting, the verifier and prover both get access to an oracle $x \in \{0,1\}^N$, and the prover provides either a quantum witness, $|\xi(x)\rangle \in (\mathbb{C}^2)^{\otimes \mathsf{poly}(n)}$ or a classical witness, $\xi(x) \in \{0,1\}^{\mathsf{poly}(n)}$. The verifier then runs an efficient quantum algorithm V^x which takes as input $|\xi(x)\rangle$ (or $\xi(x)$, respectively) and consists of quantum oracle gates applying the unitary transform defined as the linear extension of

$$|i\rangle \mapsto (-1)^{x_i} |i\rangle \text{ for } i \in [N]. \tag{1}$$

We now modify this setup slightly. Instead of a single oracle x, we consider a distribution $\mathcal{B}$ over oracles. The prover constructs a quantum witness $|\xi(\mathcal{B})\rangle$ (or a classical witness $\xi(\mathcal{B})$, respectively) based on the distribution $\mathcal{B}$. The verifier then samples a classical oracle $x \leftarrow \mathcal{B}$ from the distribution, and then runs the verification procedure V^x which takes as input $|\xi(\mathcal{B})\rangle$ (or $\xi(\mathcal{B})$, respectively) and applies quantum oracle gates corresponding to x. The success probability of the verifier is taken over the distribution $\mathcal{B}$ and the randomness in the verification procedure.

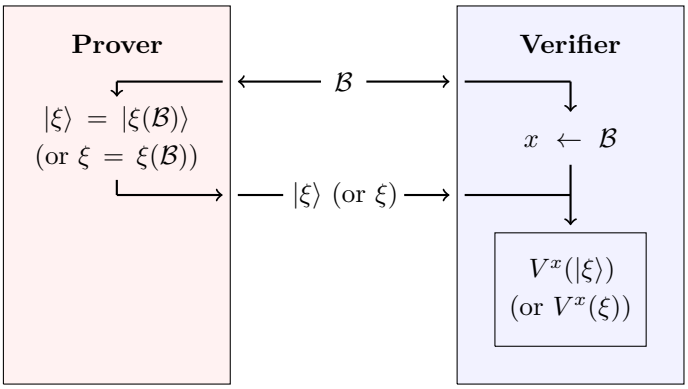


Figure 1 *Cartoon of interaction between Prover and Verifier for a distribution over classical boolean functions.*

From our previous observations, graphs with the same connected component S have the same ideal witness state. So, if the distribution $\mathcal{B}$ is supported on all graphs with the same connected component S, then the ideal witness state suffices. Furthermore, in the case of the classical witness system, the witness can only depend on S and the previously stated lower bound applies. This motivates the oracle problem of distinguishing distributions, marked either YES or NO, over 2^n bit strings (or equivalently n-bit functions).

1.3 Statement of the result

▶ **Theorem 1.** *For every sufficiently large integer n that is a multiple of 200, there exist distributions over 100-regular 100-colored graphs on $N = 2^n$ vertices labeled either* YES *or* NO *such that*

- *Each* YES *distribution is entirely supported on* YES *instances of the expander-distinguishing problem and, likewise, each* NO *distribution is entirely supported on* NO *instance of the expander-distinguishing problem.*
- *There exists a* poly(n) *time quantum algorithm V_q taking a witness state $|\xi\rangle$ as input and making $O(1)$ queries to the quantum oracle such that*

1. *For every* YES *distribution $\mathcal{B}$, there exists a quantum witness $|\xi\rangle \in (\mathbb{C}^2)^{\otimes n}$ such that*

$$\mathop{\mathbf{E}}_{x \leftarrow \mathcal{B}} \mathbf{Pr}[V_q^x(|\xi\rangle) \text{ accepts}] = 1. \tag{2}$$

2. *For every* NO *distribution $\mathcal{B}$, for all quantum witnesses $|\xi\rangle \in (\mathbb{C}^2)^{\otimes n}$,*

$$\mathop{\mathbf{E}}_{x \leftarrow \mathcal{B}} \mathbf{Pr}[V_q^x(|\xi\rangle) \text{ accepts}] \leq 0.01. \tag{3}$$

- *Any quantum algorithm V_c accepting a classical witness of length $q(n)$ satisfying the following two criteria either requires $q(n)$ to be exponential or must make an exponential number of queries to the oracle*[5].
  1. *For every* YES *distribution $\mathcal{B}$, there exists a classical witness $\xi = \xi(\mathcal{B}) \in \{0,1\}^{q(n)}$*

$$\mathop{\mathbf{E}}_{x \leftarrow \mathcal{B}} \mathbf{Pr}[V_c^x(\xi) \text{ accepts}] \geq 0.99. \tag{4}$$

  2. *For every* NO *distribution $\mathcal{B}$, for all classical witnesses $\xi \in \{0,1\}^{q(n)}$,*

$$\mathop{\mathbf{E}}_{x \leftarrow \mathcal{B}} \mathbf{Pr}[V_c^x(\xi) \text{ accepts}] \leq 0.01. \tag{5}$$

Although our main theorem is formulated as a query lower bound, it can be converted to a separation between the relativized classes of QMA and QCMA via a standard diagonalization argument. Similarly, it was pointed out to us [14] that it proves a separation between the relativized classes of BQP/qpoly and BQP/poly, following the technique of [3].

1.4 Implications and future directions

There are several future questions raised by this work that we find interesting:

Oracle and communication separations

The most natural question is, of course, whether the oracle's randomness can be removed to obtain a separation in the standard model. We conjecture that our problem yields such a separation, but a new technique seems necessary to prove it. See Section 9 for more details on the technical barriers to derandomizing our construction.

Another natural question is to show a *communication complexity* separation between QMA and QCMA. This has been shown for one-way communication by Klauck and Podder [18] but their problem does not yield a separation for two-way communication. Could our query separation be lifted to the communication world by use of the appropriate gadget?

The class QMA(2) is another relative of QMA which is perhaps even more enigmatic than QCMA. In QMA(2), the witness state is promised to be an unentangled between the first and second half of the qubits. We do not even know of a quantum (unitary) oracle separation between QMA(2) and QMA, nor do we have a natural candidate problem. Could we at least formulate such a candidate by considering "oracular" versions of QMA(2)-complete problems, in analogy to what we do in this work for QCMA.

Search-to-decision

In [15], Irani, Natarajan, Nirkhe, Rao and Yuen studied the complexity of generating a witness to a QMA problem (equivalently, generating a ground state of a local Hamiltonian) when given *oracle access* to a QMA oracle. This paradigm, called *search-to-decision*, is commonplace in classical complexity theory (for example, P, NP, MA, etc. all have search-to-decision reductions) and yet [15] gives evidence that QMA likely does not exhibit a search-to-decision reduction. They prove this by showing an oracle relative to which QMA search-to-decision reductions are provably impossible. The oracle used is identical to that of Aaronson and

[5] We leave it open whether the completeness-soundness gap in this lower bound can be improved to an inverse polynomial. Naive gap amplification for QCMA do not work in the distribution testing oracle setting; see the note below Definition 4 for details.

Kuperberg [3] to separate QMA and QCMA. [15] acknowledge this noncoincidence and conjecture whether *any* QMA and QCMA separating oracle yields a QMA search-to-decision impossibility result. Similar to the reasons for why the gold-standard of oracle separation between QMA and QCMA is a n-bit boolean function, the ideal oracle for proving QMA search-to-decision impossibility is also a n-bit boolean function. Does the oracle presented here also yield a search-to-decision impossibility?

Implications for Quantum PCPs

The quantum PCP conjecture [4] is one of the biggest open questions in quantum complexity theory. In a recent panel [24] on the quantum PCP conjecture and the NLTS theorem [7], an interesting question was posed of whether MA or QCMA (lower or upper) bounds can be placed on the complexity of the promise-gapped local Hamiltonian problem. We recommend [23] for an introduction to the subject. Because the oracle presented in this result corresponds to a sparse Hamiltonian with a problem of deciding if the second eigenvalue of the Hamiltonian is 1 or $< 1-\alpha/d = 1-\Omega(1)$, one might wonder if this provides oracular evidence that quantum PCPs are at least QCMA-hard. Unfortunately, to the best of our knowledge, this is not a reasonable conclusion. While we give evidence that the promise-gapped *sparse* Hamiltonian problem is likely QCMA-hard, the reduction from the sparse Hamiltonian problem to the local Hamiltonian problem does not imply that the promise-gapped local Hamiltonian problem is likely QCMA-hard. The only algorithm known for checking a witness for the sparse Hamiltonian problem is Hamiltonian simulation on the witness which is not a local algorithm.

Connections to Stoquastic Hamiltonians

Since the oracles studied in this work correspond to the adjacency lists of graphs, they can be viewed as sparse access to a Hamiltonian H which is the Laplacian of a graph (recall that if the adjacency matrix is A, then the Laplacian is $\mathbb{I} - A/d$). Such Hamiltonians have a special structure not present in general Hamiltonians: they are *stoquastic*, meaning that the off-diagonal entries are nonpositive. The local Hamiltonian (LH) problem for stoquastic Hamiltonians is significantly easier than the general LH problem, and in some cases is even contained in MA as shown by Bravyi and Terhal [12]. It is worth noticing why this is not in tension with our result – in particular, why this does not imply that our oracle problem is contained in oracular MA.

- Crucially, the MA-containment for stoquastic LH holds *only* for the ground state: this is because of the Perron-Frobenius theorem, which implies that ground states of such Hamiltonians have nonnegative coefficients. However, in our case, we want the first excited state: the state of minimum energy for H restricted to the subspace orthogonal to the uniform superposition. It was shown by [16] that all excited state energies are QMA-hard to calculate for a stoquastic Hamiltonian.
- The MA containment also uses the locality of the Hamiltonian, which in turn imposes a strong structure on the adjacency matrix of the graph. The random graphs we consider will not have this structure. (While it was shown by [11] showed an AM algorithm for calculating the ground energy *stoquastic and sparse* Hamiltonians, again this does not apply to higher excited states.)
- At an intuitive level, in graph language, the LH problem for stoquastic Hamiltonians is to find a component of the graph where the average value of some *potential function* (given by the diagonal entries of H) is minimized. An MA verifier can solve this by executing a

random walk, given the right starting point by Merlin. In contrast, our problem is to determine whether the graph as a whole is connected – a global property which an MA verifier cannot determine.

2 Organization of the paper

The remainder of the paper is the proof of Theorem 1. The proof is divided into smaller components and these intermediate results are joined together in Section 8. In Section 3, we state some basic definitions and formally define the expander distinguishing problem. In Section 4, we describe the distributions over graphs that constitute YES and NO instances. In Section 5, we prove that there is an efficient QMA algorithm for the expander distinguishing problem. In particular, there is a single quantum witness that serves all the graphs in each of the YES distributions. In Section 6, we use the adversary method and counting arguments to prove that any QCMA algorithm for the expander distinguishing problem for the constructed distributions implies a BQP algorithm for distinguishing YES instances with a connected component corresponding to an ideal sunflower from a generic NO instance. In Section 7, we argue using the polynomial method that such an algorithm is impossible without an exponential query complexity. In Section 9, we present some concluding remarks about our construction and its relation to other notions of computational complexity.

3 Preliminaries

3.1 Notation and quantum information basics

We will assume that the reader is familiar with the basics of quantum computing and quantum information. We will use $N \stackrel{\text{def}}{=} 2^n$ throughout this paper and we will only consider graphs of N vertices. The adjacency list of a d-regular d-colored graph on N vertices takes dnN bits to describe. For any m, we abbreviate the set of integers $\{1, 2, \dots m\}$ as $[m]$. For a set $A \subseteq [N]$, we will use $|A\rangle$ to denote the state $\frac{1}{\sqrt{A}} \sum_{j \in A} |j\rangle$, the subset state corresponding to A. Unless otherwise, specified we assume $\|\cdot\|$ is the Euclidean norm $\|\cdot\|_2$ for a vector, and the spectral norm for a matrix, which is the largest singular value.

3.2 Expander graphs

▶ **Definition 2.** *A graph G is a* spectral α-expander *(equiv. is α-expanding) if the second highest eigenvalue λ_2 of the normalized adjacency matrix of G satisfies $\lambda_2 \leq 1 - \alpha$. We say that a connected component S of the graph is α-expanding if the restricted graph to the vertices of S is α-expanding.*

▶ **Lemma 3.** *Let G be a d-regular α-expander. Consider the random walk that starts in any distribution over the vertices, and at each time step, stays in place with probability $1/2$, and moves along an edge of the graph with probability $1/2$. Then for any vertex v, after ℓ steps, the probability $\mathbf{Pr}[v]$ that the walk is in v satisfies*

$$\left|\mathbf{Pr}[v] - \frac{1}{N}\right| \leq \left(1 - \frac{\alpha}{2}\right)^{\ell}. \tag{6}$$

In particular, when $\ell = O(c \log N/\alpha)$ we can get the RHS to be $1/N^c$.

Proof. Proof is available in the full version [22]. ◀

3.3 Non-deterministic oracle problems

▶ **Definition 4** (Quantum oracle problems). *For a n-bit boolean function $\mathcal{O}$, we say an* oracle *decision problem $\mathcal{L}^{\mathcal{O}}$ is in $\mathsf{QMA}^{\mathcal{O}}(\epsilon)$ if there exists a uniform family of quantum circuits $A^{\mathcal{O}}$ such that*

1. *For every* YES *instance $\mathcal{O}$, there exists a quantum state $|\xi\rangle$ of $\mathsf{poly}(n)$ qubits such that $A^{\mathcal{O}}(|\xi\rangle)$ accepts with probability $\geq 1-\epsilon$.*
2. *For every* NO *instance $\mathcal{O}$, for all quantum states $|\xi\rangle$ of $\mathsf{poly}(n)$ qubits, $A^{\mathcal{O}}(|\xi\rangle)$ accepts with probability $\leq \epsilon$.*

$\mathsf{QCMA}^{\mathcal{O}}(c,s)$ is defined similarly, except the state $|\xi\rangle$ is promised to be classical. The classes $\mathsf{QMA}^{\mathcal{O}}$ and $\mathsf{QCMA}^{\mathcal{O}}$ are defined as $\mathsf{QMA}^{\mathcal{O}}(1/3)$ and $\mathsf{QCMA}^{\mathcal{O}}(1/3)$, respectively.

We note that due to parallel repetition, $\mathsf{QMA}^{\mathcal{O}}(\epsilon = \frac{1}{2} - 1/\mathsf{poly}(n)) = \mathsf{QMA}^{\mathcal{O}} = \mathsf{QMA}^{\mathcal{O}}(\epsilon = 2^{-\mathsf{poly}(n)})$. Likewise, for $\mathsf{QCMA}^{\mathcal{O}}$. This justifies removing the constant ϵ from the definition. We now define the same problem for oracles equaling distributions over n-bit boolean functions.

▶ **Definition 5** (Random classical oracles). *A random oracle $\mathcal{R}$ is a distribution over classical oracles $\{\mathcal{O}\}$. We say an* oracle *decision problem $\mathcal{L}^{\mathcal{R}}$ is in $\mathsf{QMA}^{\mathcal{R}}(\epsilon)$ if there exists a uniform family of quantum circuits $A^{\mathcal{O}}$ such that*

1. *For every* YES *instance $\mathcal{R}$, there exists a quantum state $|\xi\rangle$ of $\mathsf{poly}(n)$ qubits such that*

$$\mathop{\mathbf{E}}_{\mathcal{O}\in\mathcal{R}} \mathbf{Pr}\left[A^{\mathcal{O}}(|\xi\rangle) \text{ accepts}\right] \geq 1-\epsilon. \tag{7}$$

2. *For every* NO *instance $\mathcal{O}$, for all quantum states $|\xi\rangle$ of $\mathsf{poly}(n)$ qubits,*

$$\mathop{\mathbf{E}}_{\mathcal{O}\in\mathcal{R}} \mathbf{Pr}\left[A^{\mathcal{O}}(|\xi\rangle) \text{ accepts}\right] \leq \epsilon. \tag{8}$$

$\mathsf{QCMA}^{\mathcal{R}}(c,s)$ is defined similarly, except the state $|\xi\rangle$ is promised to be classical.

Ideally, we would define the classes $\mathsf{QMA}^{\mathcal{R}}$ and $\mathsf{QCMA}^{\mathcal{R}}$ are defined as $\mathsf{QMA}^{\mathcal{R}}(1/3)$ and $\mathsf{QCMA}^{\mathcal{R}}(1/3)$, respectively. However, the parallel repetition argument for boolean function oracles cannot be extended to distributions over boolean functions. This is because the ϵ error that an algorithm is the expectation of the success probability of the algorithm over the distribution. It is possible that the algorithm runs on every instance in the distribution with error ϵ or it is possible that the algorithm succeeds with 0 error on a $1-\epsilon$ fraction of the distribution and fails on the remaining ϵ fraction. In the first case, the success of the algorithm can be improved with parallel repetition while it cannot in the second case[6].

3.4 Graph oracles

▶ **Definition 6** (Colored Graphs). *Given a d-colored d-regular graph $G=(V,E)$ on N vertices, we say G contains a triple $(j_1,j_2,\kappa) \in V^2 \times [d]$ if the edge (j_1,j_2) exists in G and is colored with color κ.*

[6] We note that this subtlety is overlooked in Fefferman and Kimmel [13] but we believe that their result without parallel repetition is correct. Furthermore, the adversary bounds used in [13] do not address this issue but can be rectified using the adversary bound stated in Theorem 13 which deals with *average-case* distinguishing.

▶ **Definition 7** (Adjacency graph oracles). *Let G be a d-colored d-regular undirected graph. The graph $G = (V, E)$ can be described by an adjacency function*

$$G : V \times [d] \to V \tag{9}$$

where the output of (j, κ) returns the neighbor of j along the edge colored with κ. Quantum access to the function G is provided by the following oracle *unitary:*

$$|j, \kappa, z\rangle \overset{G}{\mapsto} |j, c, z \oplus G(j, \kappa)\rangle \,. \tag{10}$$

We call the function G the adjacency graph oracle *corresponding to G.*

▶ **Definition 8** (Expander distinguishing problem). *The (α, ζ)-expander distinguishing problem is a promise oracle language where the input is an oracle G for a d-colored d-regular undirected graph G on N vertices. The problem is to distinguish between the following two cases, promised that one holds:*

- YES*: the graph G has a connected component S of size at most $|S| \leq \zeta$.*
- NO*: the graph G is an α-expander.*

In this paper, we will think of α as a constant and $\zeta \sim N^{9/10}$. To simplify notation, since the oracles considered in this result always correspond to graphs G, we express the algorithm as $\mathcal{A}^G$ rather than $\mathcal{A}^{\mathcal{O}}$.

4 Random distributions over graphs with many connected components

In this subsection, we describe distributions over graphs where the graphs with high probability consist of ℓ connected components. It should not be surprising that the distribution is almost identical to the distribution used by Ambainis, Childs, and Liu [6] in their proof that the expander distinguishing problem requires an exponential number of quantum queries for any quantum query algorithm in the absence of a proof. This is because we will reduce any QCMA algorithm to an efficient query algorithm for some expander distinguishing problem.

The lower bound in [6] is crucially a lower bound on the *polynomial degree* of any polynomial that distinguishes two graph distributions. From there, it isn't too much to argue that these graph distributions are very close to YES and NO instances as prescribed in the expander distinguishing problem; therefore any algorithm solving the expander distinguishing problem must be able to distinguish these two graph distributions. Our first goal is to amplify the argument of [6] to a more restricted class of graphs.

4.1 Graphs distributions inspired by [6]

The goal of the construction is a distribution which depends on an integer ℓ and a subset $F \subset V$. The integer ℓ will roughly correspond to the number of connected components (henceforth denoted $C_1, \ldots, C_\ell$) in the graph and we insist that $F \subset C_1$. Every $v \in V \setminus F$ appears in each subset C_i with equal probability of $1/\ell$. The actual construction will be slightly more complicated than this but, morally, this is what we hope to achieve from the distribution.

Formal construction

Let N be an integer and for integer $M \geq N$, integer ℓ dividing M and a subset $F \subset V$ define the distribution $P_{M,\ell}(F)$ over graphs on N vertices as follows:

1. Start by constructing a graph G' on M vertices: Partition V' into ℓ equally sized sets of vertices $V_1, \ldots, V_\ell$. On each subset V_k, create a random *colored* subgraph by randomly choosing d perfect matchings (each with a different color $1, \ldots, d$) and taking their union.

2. To construct the graph G on N vertices: We first choose an injective map $\iota : V \hookrightarrow V'$. First, we pick a function $k : V \to [\ell]$. We pick k as a uniformly random function conditioned on the fact that $k(j) = 1$ for each $j \in F$. Let $\iota(j)$ be a random vertex from $V_{k(j)}$ *without replacement* to satisfy injectivity. If all vertices from $V_{k(j)}$ have been selected with replacement, output the graph on N vertices with no edges (i.e. abort).
3. Induce a graph G on V from G' and the map ι – i.e. an edge $(j_1, j_2, \kappa) \in V^2 \times [d]$ exists if $(\iota(j_1), \iota(j_2), \kappa) \in V'^2 \times [d]$ is an edge.
4. For a vertex j and a color κ, if the previous induced edges did not introduce a κ-colored edge from j, then add edge (j, j, κ).
5. The distribution over graphs G is henceforth called $P_{M,\ell}(F)$; when $F = \emptyset$, we write it as $P_{M,\ell}$.

Notationally, for edges $e = (u, v, \kappa) \in G$, we will use $\iota(e) = (\iota(u), \iota(v), \kappa) \in G'$. Furthermore, we will extend ι naturally to subgraphs and subsets of vertices and edges.

▶ Remark 9. For any F, $P_{M,1}(F) = P_{M,1}(\emptyset) \stackrel{\text{def}}{=} P_{M,1}$.

4.2 Setting of constants

The lower bounds we prove for the QCMA algorithm are by no means tight (up to constants). We make no attempt to perfect the choice of constants as our only goal is to prove an exponential lower bound on the size of the any quantum witness or the number of queries required to solve the expander distinguishing problem. For this reason, we pick the following constants:

Chosen constants

The degree of the graph G' is set to be $d = 100$. We assume $\ell = N^{1/10}$, $\gamma = N^{-1/10}$ and $M = (1 + \gamma)N$.

Induced constants

In Definition 8, we define an (α, ζ)-expander distinguishing problem. We will only consider $\alpha = 1/(2 \cdot 10^8)$ (which is a consequence of Lemma 10 and the chosen constants). Notationally, we will use $z \stackrel{\text{def}}{=} N/\ell = N^{9/10}$. We will use $\zeta = (1 + \gamma)z = M/\ell$.

Conventions

Typically, we will assume (for the purposes of contradiction) that $|F| \leq N^{1/100}$ but as that is a term we wish to bound, we explicitly state it each time. Anytime a set S is described, it will be of size ζ, but we will also state this.

4.3 Concentration bounds for random distributions over graphs

We will need the following concentration lemma about the generated distributions. The lemma proves that $P_{M,1}$ is approximately a YES instance and that $P_{M,\ell}(F)$ is approximately a NO instance. Overall, this lemma proves that any algorithm solving the expander distinguishing problem must do very well on identifying the distribution $P_{M,1}$ as a NO instance and identifying the distributions $P_{M,\ell}(F)$ as a YES instance. The proof of this lemma is provided in Appendix of the full version [22].

▶ **Lemma 10** (Adaptation of Lemma 16 of [6]). *Assume $|F| \leq N^{1/100}$. Then with probability at least $\geq 1 - O(N^{-3})$, a graph drawn from distribution $P_{M,\ell}(F)$ consists of exactly ℓ connected components each α-expanding and consisting of between $(1-\gamma)z$ and $(1+\gamma)z$ vertices.*

Likewise, the probability that a graph drawn from the distribution $P_{M,1}$ is α-expanding is $\geq 1 - O(N^{-3})$.

Note that being expanding necessarily implies connectivity. Note that when $F = \emptyset$ or $\ell = 1$, there are simpler proofs with tighter bounds but the bound proven here for the general statement is sufficient for our result.

The second concentration lemma that we will use is that $P_{M,\ell}$ is approximately equal to sampling a set F of size $\leq N^{1/100}$ and then sampling a graph from $P_{M,\ell}(F)$. The proof of this lemma is also provided in Appendix of the full version [22].

▶ **Lemma 11.** *Let m be $\leq N^{1/100}$. Let $\mathcal{D}_1$ be the distribution on pairs (G, F) obtained by sampling $G \sim P_{M,\ell}$, choosing a uniformly random vertex $v \in G$, and then choosing F to be a uniformly random subset of the connected component of G containing v of size m. Let $\mathcal{D}_2$ be the distribution on pairs (G, F) obtained by first choosing F to be a uniformly random subset of V with size m, and then sampling $G \sim P_{M,\ell}(F)$. Then these distribution are close in statistical distance:*

$$\|\mathcal{D}_1 - \mathcal{D}_2\| \leq 3N^{-9/200}. \tag{11}$$

5 QMA protocol

In this section we show that the expander distinguishing problem (over a fixed graph – i.e., no distribution) can be solved with a polynomial number of queries (indeed, with just two queries) if a quantum witness is provided. Our algorithm has the added benefit of being time-efficient, so we have shown that this problem is contained in QMA^G. In Section 8, we prove that there still exists a QMA protocol if we consider distribution oracles.

▶ **Lemma 12.** *There is a QMA^G protocol $\mathcal{A}_{\mathsf{QMA}}$ that solves the (α, ζ)-expander distinguishing problem with the following properties:*

1. ***Query complexity:*** *the algorithm makes two queries to G.*
2. ***Completeness:*** *In the YES case, there exists a witness state that the verifier accepts with certainty.*
3. ***Soundness:*** *In the NO case, no witness state is accepted by probability greater than $1 - \alpha/4$.*
4. ***Nice witnesses:*** *In the YES case, if $S \subsetneq V$ is a connected component of the graph G, then the state*

$$|S\rangle = \frac{1}{\sqrt{|S|}} \sum_{v \in S} |v\rangle$$

is accepted with probability at least $1 - \sqrt{|S|/N}$. In particular, since there exists a connected component of size at most ζ, there is a state of this form that is accepted with probability $1 - \sqrt{\zeta/N}$.

Proof. At a high level, the verifier performs one-bit phase estimation of the adjacency matrix of G on the witness state; this verifies that the witness was an eigenvector of optimal eigenvalue of the graph's adjacency matrix. A complete proof is available in the full version [22]. ◀

6 Adversary method

In this section, we use the adversary method of Ambainis to argue that any successful QCMA algorithm implies a BQP algorithm for distinguishing the distribution $P_{M,\ell}(F)$ and $P_{m,1}$ for $|F| \leq N^{1/100}$. In Section 6.1, we state the adversary method result and in the following sections we prove the statement.

6.1 Ambainis' proof of the adversary method

The adversary method of Ambainis [5] is a convenient way of arguing lower bounds on the query complexity of oracular quantum algorithms. The adversary method lower bounds the complexity of any algorithm which (with high probability) computes $f(a)$ for a function $f : \{0,1\}^N \to \{0,1\}$. The quantum algorithm is allowed access to $a \in \{0,1\}^N$ by a oracle gate O which applies linearly the transform $|i\rangle \mapsto (-1)^{a_i} |i\rangle$ for $i \in \{0,1\}^n$ (here $N = 2^n$). In doing so, the adversary method is a convenient way of producing BQP (query) lower-bounds.

To use it in our distributional setting, we make two modifications to the adversary bound. The first is to relax the notion of correctness. The lower bound of Ambainis is for a lower-bound for any algorithm which, for *each* $a \in \{0,1\}^N$, outputs $f(a)$ correctly with probability $1 - \epsilon$ for $\epsilon < \frac{1}{2}$. We instead consider an *average-case* notion of success in which

$$\mathop{\mathbf{E}}_{a \in \{0,1\}^N} \mathop{\mathbf{Pr}}_{\mathcal{A}} [\mathcal{A}^a \neq f(a)] \leq \epsilon^2. \tag{12}$$

By Markov's inequality, this implies

$$\mathop{\mathbf{Pr}}_{a \in \{0,1\}^N} \left[\mathop{\mathbf{Pr}}_{\mathcal{A}} [\mathcal{A}^a \neq f(a)] \geq \epsilon \right] \leq \epsilon, \tag{13}$$

or in other words, most a are (with high probability) correctly identified.

The second modification is to *restrict* the set of locations that the algorithm is allowed to query the oracle. The reason for this is somewhat subtle. Essentially, the original lower bound of Ambainis was designed for decision problems with deterministic oracles, and relies on constructing a relation between two *disjoint* sets of oracle instances, one consisting only of YES instances and the other only of NO instances. However, in our setting, we are interested in distinguishing two distributions over oracles that may have overlapping support. In order to define disjoint YES and NO sets of instances even when the distributions overlap, we add to each oracle string a a set of flag bits b that indicate which of the two distributions the string a was sampled from. Naturally, any reasonable model cannot permit the algorithm to query the flag bits: otherwise, it would be easy to distinguish even two statistically close distributions with few queries.

More formally, we consider a generalization where the oracle string is a tuple $(a, b) \in \{0,1\}^N \times \{0,1\}^M$ and $f : \{0,1\}^{N+M} \to \{0,1\}$ but the algorithm can only query positions of a. In this model, with the average-case notion of success defined above, we obtain the following adversary lower bound for distributions:

▶ **Theorem 13.** *Let $f : \{0,1\}^{N+M} \to \{0,1\}$ be a function and let $X, Y \subset \{0,1\}^N \times \{0,1\}^M$ be two subsets such that $X \subset f^{-1}(0)$ and $Y \subset f^{-1}(1)$. Let $R \subset X \times Y$ be a relation such that*

1. *For every $x \in X$, let $R_x \subset Y$ equal $R_x = \{y : (x,y) \in R\}$ such that $\underline{m} \leq |R_x| \leq \overline{m}$.*
2. *For every $y \in Y$, let $R_y \subset X$ equal $R_y = \{x : (x,y) \in R\}$ such that $\underline{m'} \leq |R_y| \leq \overline{m'}$.*
3. *For every $x = (a,b) \in X$ and $i \in [N]$, let $\ell_{x,i}$ be the number of $y = (c,d) \in Y$ such that $(x,y) \in R$ and $a_i \neq c_i$. Likewise, for every $y = (c,d) \in Y$ and $i \in [N]$, let $\ell_{y,i}$ be the number of $x = (a,b) \in X$ such that $(x,y) \in R$ and $a_i \neq c_i$. Let $\ell_{\max}$ be the maximum product $\ell_{x,i}\ell_{y,i}$ over $(x,y) \in R$ and $i \in [N]$ such that $a_i \neq c_i$.*

Then any quantum algorithm $\mathcal{A}$ which only queries the first N bits of the oracle and computes f such that

$$\mathop{\mathbf{E}}_{x=(a,b)\in X} \mathop{\mathbf{Pr}}_{\mathcal{A}}[\mathcal{A}^a \neq 0] \leq \epsilon^2 \quad \text{and} \quad \mathop{\mathbf{E}}_{y=(c,d)\in Y} \mathop{\mathbf{Pr}}_{\mathcal{A}}[\mathcal{A}^c \neq 1] \leq \epsilon^2 \tag{14}$$

uses

$$\geq \left(1 - 2\sqrt{\epsilon(1-\epsilon)}\right)\sqrt{\frac{(\underline{m} - 2\epsilon\overline{m})(\underline{m'} - 2\epsilon\overline{m'})}{\ell_{\max}}} \quad \text{queries.} \tag{15}$$

▶ **Corollary 14.** *Let X and Y be two subsets of $\{0,1\}^{N+M}$ satisfying the three conditions listed in Theorem 13. Then, any query algorithm $(1-\delta)$-distinguishing the uniform distributions over X and Y, must use eq. (15) queries for $\epsilon = 2\delta$.*

The proofs of both statements are presented in the Appendix of the full version [22].

6.2 Setup from QCMA algorithm

In this subsection, we show that if there is a QCMA algorithm for solving the *expander distinguishing problem* then there exists a sunflower ❀ (defined below) of YES instances which correspond to the same optimal witness $\mathsf{wt}^\star$. If we hardcode $\mathsf{wt}^\star$ into the QCMA algorithm, we generate a quantum query algorithm that, with no access to a prover, accepts instances corresponding to ❀ and rejects all NO instances.

▶ **Definition 15** (Sunflower). *A collection of subsets ❀ $\subset \binom{V}{\zeta}$ is* (μ, ζ, t)-sunflower *if there exists a subset $F \subset V$ with $|F| \leq t$ satisfying the following two conditions:*

1. *For all $S \in$ ❀, $F \subseteq S$.*
2. *For all $x \in \left(\bigcup_{S\in❀} S\right) \setminus F$, the $\mathop{\mathbf{Pr}}_{S\in❀}[x \in S] \leq \left(\frac{\zeta}{N}\right)^{1-\mu}$.*

We call the set F the core of the sunflower.

YES instances corresponding to subsets

For any graph G and subset S of size ζ, define $G \triangleleft S$ if G has a connected component $C_i \subseteq S$. Let $\mathsf{S}_\triangleleft$ be the set of G such that $G \triangleleft S$. For each subset S of size ζ, define B_S to be the restriction of the distribution $P_{M,\ell}$ to graphs in $\mathsf{S}_\triangleleft$. The intuition is that the ideal witness will be a good witness for B_S since the connected components of $P_{M,\ell}$ are of a size concentrated around z.

There is a small complication, which we address now, in that the distribution $P_{M,\ell}$ is not a uniform distribution over a set of graphs. To rectify this, we can always assume that the oracle corresponding to a graph G sampled to $P_{M,\ell}$ consists of a queryable component corresponding to the adjacency list of G and a non-queryable component corresponding to the random coins r_G that were flipped in order to generate G according to $P_{M,\ell}$. We will also define B_S as the restriction of the extended oracle. Therefore, both $P_{M,\ell}$ and B_S are uniform distributions over some support.

Lastly, the distributions B_S are not exactly YES distributions since their support is not *entirely* on YES graphs of the expander distinguishing problem. However, similar to Lemma 10, we will show that B_S is almost entirely supported on YES graphs. Therefore, it suffices to use B_S as a *proxy* for YES instances until the very end where we handle this subtlety.

▶ **Corollary 16.** *For every graph $G \in \mathsf{S}_{\triangleleft}$ such that G has a connected component C_i with $|C_i| \geq (1-\gamma)z$,*

$$\mathbf{Pr}[\mathcal{A}^G_{\mathsf{QMA}}(|S\rangle) = 1] \geq 1 - 3\sqrt{\gamma}. \tag{16}$$

Proof. Proof is available in the full version [22]. ◀

▶ **Corollary 17.** *Let $M = (1+\gamma)N$ and $\gamma = N^{-1/10}$ and $\ell = N^{-1/10}$. For every S of size $(1+\gamma)z$, the distribution B_S is a* YES *instance and*

$$\mathop{\mathbf{E}}_{G \leftarrow B_S}\left[\mathbf{Pr}\left[\mathcal{A}^G_{\mathsf{QMA}}(|S\rangle) = 1\right]\right] \geq 1 - 3\sqrt{\gamma} - O(N^{-3}). \tag{17}$$

Proof. Proof is available in the full version [22]. ◀

QCMA algorithm implies a quantum low-query algorithm for some sunflower ❀

▶ **Lemma 18.** *For some $\epsilon > 0$, assume there exists a k-query non-deterministic quantum algorithm which accepts a q-length* classical *witness and accepts every distribution B_S for subset S of size ζ with probability $1-\epsilon$ and accepts any* NO *distribution B_{NO} with probability at most ϵ. Then for $\mu > 0$, there exists a $(\mu, \zeta, 2q/(\mu \log \ell))$-sunflower ❀ and a k-query quantum algorithm that accepts every distribution B_S for $S \in$ ❀ and accepts any* NO *distribution B_{NO} with probability at most ϵ.*

Proof. Proof is available in the full version [22]. ◀

6.3 Query lower bound for distinguishing sunflowers and fixed distributions

Let $Ⓕ \stackrel{\text{def}}{=} \binom{V}{\zeta} \cap \{S : F \subseteq S\}$. This is the *ideal sunflower* with a core of F. We will show by an adversary bound that the sunflower ❀ and the ideal sunflower Ⓕ are indistinguishable by quantum query algorithms with few queries. Consider the distribution $H_{❀}$ defined by sampling an $S \in$ ❀ and then sampling a graph from B_S. Similarly, define the distribution $H_{Ⓕ}$ but by first sampling an $S \in Ⓕ$. We want to show that any quantum query algorithm requires exponentially many queries to distinguish $H_{❀}$ and $H_{Ⓕ}$. The main result of this subsection is the following lemma.

▶ **Lemma 19.** *For $\delta < 1/4$, any quantum query algorithm $(1-\delta)$-distinguishing the distributions $H_{❀}$ and $H_{Ⓕ}$ where ❀ is a (μ, ζ, t)-sunflower and F is the corresponding core requires*

$$\geq \frac{1}{2}\left(1 - 2\sqrt{2\delta(1-2\delta)}\right)(1-4\delta) \cdot \sqrt{\left(\frac{N}{\zeta}\right)^{1-\mu}} \text{ queries.} \tag{18}$$

6.3.1 A warmup lemma for distinguishing graphs

The main challenge in proving Lemma 19 is the complicated structure inherent in graphs. However, if we work instead directly with the *sets* S, the problem is much simpler, and was already solved in [13, Lemma 11]. They showed that given membership query access (equivalently, the indicator function for the set), it requires exponentially many quantum queries to distinguish a sample from ❀ from a sample from Ⓕ.

We will work up to the result we wish to prove by gradually adding more structure to the objects being queried until we reach graphs. We will start by working with *permutations* that map the set S to a known set, and show that any algorithm with query access to the permutation and its inverse requires exponentially many queries.

To be precise, let $U = [\zeta]$. Let $\Pi_{❀}$ be the set of all permutations and inverses (π, π^{-1}) such that $\pi(S) = U$ for some $S \in ❀$. Similarly, define $\Pi_{Ⓕ}$. We shall abuse notation and also use $\Pi_{❀}$ and $\Pi_{Ⓕ}$ to refer to the uniform distributions over these sets of permutations. We first claim that no quantum query algorithm can distinguish the distributions $\Pi_{❀}$ and $\Pi_{Ⓕ}$ without an exponential number of queries. Note that the algorithm is allowed to query both the permutation and its inverse[7].

▶ **Lemma 20.** *Any quantum query algorithm $(1-\delta)$-distinguishing the distributions $\Pi_{❀}$ and $\Pi_{Ⓕ}$ where ❀ is a (μ, ζ, t)-sunflower and F is the corresponding core requires*

$$\geq \left(\frac{1}{2} - 2\sqrt{2\delta(1-2\delta)}\right)(1-4\delta) \cdot \sqrt{\left(\frac{N}{\zeta}\right)^{1-\mu}} \text{ queries.} \tag{19}$$

Proof. Proof is available in the full version [22]. ◀

A short corollary of Lemma 20 is that there is a similar query lower bound for distinguishing distributions over graphs. Let $\mathcal{G}$ be a distribution over graphs with a connected component of $U = [\zeta]$. Let $\mathcal{G}_{❀}$ be the distribution over graphs formed by sampling a permutation pair (π, π^{-1}) from $\Pi_{❀}$, a graph G from $\mathcal{G}$ and outputting the graph $\pi^{-1}(G)$. By construction, $\mathcal{G}_{❀}$ is a distribution over graphs with a connected component of S for $S \in ❀$. Likewise, define the distribution $\mathcal{G}_{Ⓕ}$.

▶ **Corollary 21.** *For $\delta < 1/4$, any quantum query algorithm $(1-\delta)$-distinguishing the distributions $\mathcal{G}_{❀}$ and $\mathcal{G}_{Ⓕ}$ where ❀ is a (μ, ζ, t)-sunflower and F is the corresponding core requires*

$$\geq \frac{1}{2}\left(1 - 2\sqrt{2\delta(1-2\delta)}\right)(1-4\delta) \cdot \sqrt{\left(\frac{N}{\zeta}\right)^{1-\mu}} \text{ queries.} \tag{20}$$

Proof. The intuition is that any algorithm $\mathcal{A}$ for distinguishing $\mathcal{G}_{❀}$ and $\mathcal{G}_{Ⓕ}$, can be used as a subroutine in a (not necessarily time-efficient) algorithm $\mathcal{A}'$ for distinguishing $\Pi_{❀}$ and $\Pi_{Ⓕ}$ in twice as many queries. A full proof is available in the full version [22]. ◀

6.3.2 Improving to more general permutations

While Lemma 20 and Corollary 21 are simple enough to prove, they are insufficient at proving indistinguishability for the graph distributions $H_{❀}$ and $H_{Ⓕ}$ defined at the start of this section. This is because, unlike the distribution $\mathcal{G}_{❀}$, the distribution $H_{❀}$ cannot be defined in terms of independently sampling a graph G and a set S. For one, the sizes of the connected components in $H_{❀}$ do not exactly equal z; instead, the concentrate tightly around z. It was precisely the independence of the graphs and sets that made Corollary 21 easy to prove.

To fix the argument, we prove the following variations of Lemma 20 and Corollary 21. For a sunflower ❀ with core F and any k such that $|F| \leq k \leq \zeta$, let $\Pi_{❀}^{(k)}$ be the distribution formed by the following procedure:

1. Sample a set S from ❀.
2. Sample uniformly randomly a subset $C \subset S$ of size k.
3. Sample uniformly randomly a permutation $\pi : V \to V$ such that $\pi(C) = [k]$.
4. Output (π, π^{-1}).

Define the distribution $\Pi_{Ⓕ}^{(k)}$ similarly where we change the first step to sampling from Ⓕ.

[7] This can be equivalently modeled by having a separate in-place oracle for the permutation and its inverse, or having a single "standard" oracle for the permutation.

▶ **Lemma 22.** *Any quantum query algorithm $(1-\delta)$-distinguishing the distributions $\Pi_{❀}^{(k)}$ and $\Pi_{Ⓕ}^{(k)}$ where ❀ is a (μ,ζ,t)-sunflower and F is the corresponding core requires*

$$\geq \left(1-2\sqrt{2\delta(1-2\delta)}\right)(1-4\delta)\cdot\sqrt{\left(\frac{N}{\zeta}\right)^{1-\mu}} \text{ queries.} \tag{21}$$

Proof. This proof is equivalent to that of Lemma 20 except we use $U=[k]$. Note, the listed bound has no dependence on k; this is because $k\leq\zeta$ and we express here the weaker bound with ζ. ◀

Likewise, a short corollary of Lemma 22 is the following. Construct the distribution $\mathcal{G}_{❀}^{(k)}$ by the following procedure:

1. Sample a graph G from the restriction of the distribution $P_{M,\ell}$ to graphs with a connected component of exactly $[k]$.
2. Sample a permutation (π,π^{-1}) from $\Pi_{❀}^{(k)}$.
3. Output $(\pi^{-1}(G),r_G)$ where r_G is the random coin flips that would have generated G when sampling according to $P_{M,\ell}$. The oracle will be divided into a queryable component of $(\pi^{-1}(G))$ and a un-queryable component of r_G.

▶ **Corollary 23.** *For $\delta<1/4$, any quantum query algorithm $(1-\delta)$-distinguishing the distributions $\mathcal{G}_{❀}^{(k)}$ and $\mathcal{G}_{Ⓕ}^{(k)}$ where ❀ is a (μ,ζ,t)-sunflower and F is the corresponding core requires*

$$\geq \frac{1}{2}\left(1-2\sqrt{2\delta(1-2\delta)}\right)(1-4\delta)\cdot\sqrt{\left(\frac{N}{\zeta}\right)^{1-\mu}} \text{ queries.} \tag{22}$$

Proof. The corollary follows from Lemma 22 via a reduction from permutations to graphs exactly as in the proof of Corollary 21 from Lemma 20. ◀

6.3.3 Completing the proof

Proof of Lemma 19. Notice that for any $k\neq k'$, the support of $\mathcal{G}_{❀}^{(k)}$ is disjoint from the support of $\mathcal{G}_{❀}^{(k')}$, and likewise for $\mathcal{G}_{Ⓕ}^{(k)}$ and $\mathcal{G}_{Ⓕ}^{(k')}$. Let us again abuse notation and use $\mathcal{G}_{❀}^{(k)}$ to denote the support of the corresponding distribution. For each k, the lower bound from Corollary 23 is shown via an adversary bound with a relation R_k, and parameters $m,m',\ell_{\max}$, and moreover these parameters are the same for all k. Thus, we may construct a relation R between $\bigcup_k \mathcal{G}_{❀}^{(k)}$ and $\bigcup_k \mathcal{G}_{Ⓕ}^{(k)}$ by simply taking the union $R=\bigcup_k R_k$. This relation maintains the same parameters $m,m',\ell_{\max}$ due to the disjointness of supports for different k. Lastly, notice that $\bigcup_k \mathcal{G}_{❀}^{(k)}$ is equal to the support of $H_{❀}$ as described in the statement of Lemma 19. Likewise, for $H_{Ⓕ}$. Since $H_{❀}$ and $H_{Ⓕ}$ are uniform distributions over their support, by Corollary 14 using the relation R that we have constructed, the distributions are indistinguishable without the stated number of queries. ◀

6.4 Statistical indistinguishability between random distributions

The final step of this section is to show that no algorithm can distinguish the distributions $H_{Ⓕ}$ and $P_{M,\ell}(F)$ with more than a negligible probability. This will be because these distributions are statistically close and this can be proven by a Chernoff tail bound.

▶ **Lemma 24.** *The statistical distance between $H_{Ⓕ}$ and $P_{M,\ell}(F)$ is $O(N^{-3})$.*

Proof. Notice that the distribution $H_{\textcircled{F}}$ is equivalent to sampling a graph from $P_{M,\ell}(F)$ conditioned on consisting of ℓ connected components each with size $\in [(1-\gamma)z, (1+\gamma)z]$. By Lemma 10, with all but $O(N^{-3})$ probability, a graph from $P_{M,\ell}(F)$ satisfies this condition. Therefore, the statistical distance between these distributions is bounded by $O(N^{-3})$. ◀

7 Polynomial method lower bound

In this section, we prove that any quantum query algorithm cannot distinguish the graph distributions $P_{M,1}$ and $P_{M,\ell}(F)$. When $F = \emptyset$, this is equivalent to the problem studied by [6] in their quantum query lower bound:

▶ **Theorem 25** (Restatement of Theorem 2 of [6]). *For any sufficiently small constant $\epsilon_1 > 0$, any* deterministic *quantum query algorithm $\mathcal{A}$ distinguishing the distributions $P_{M,1}$ and $P_{M,\ell}$ for any $1 < \ell < N^{1/4}$ by probability ϵ_1. I.e.*

$$\mathop{\mathbf{E}}_{G \leftarrow P_{M,1}} \left[\Pr_{\mathcal{A}} \left[\mathcal{A}^G = 1 \right] \right] - \mathop{\mathbf{E}}_{G \leftarrow P_{M,\ell}} \left[\Pr_{\mathcal{A}} \left[\mathcal{A}^G = 1 \right] \right] \geq \epsilon_1 \tag{23}$$

must make at least $\Omega(N^{1/4}/\log N)$ queries. Here the Ω notation hides a dependence on ϵ_1.

The proof used in that result is very technical and builds on the polynomial method. Fortunately, we can show our query lower bound via a *reduction* to the [6] result. The reduction requires taking a short walk which mixes well by the expander mixing lemma.

▶ **Lemma 26.** *Suppose there exists some F_0 and a q_1-query quantum algorithm that ϵ_1-distinguishes the distributions $P_{M,1} = P_{M,1}(F_0)$ and $P_{M,\ell}(F_0)$ for $\ell > 1$. Then there exists a q_2-query quantum algorithm that ϵ_2-distinguishes the distributions $P_{M,1}$ and $P_{M,\ell}$ with $q_2 = q_1 + O(N^{3/100})$ and $\epsilon_2 = \epsilon_1 - O(N^{-9/200})$.*

Intuitively, what this lemma says is that the set of points F_0 (which are in the same connected component) is not a helpful witness. Concretely, such a witness is negligibly more helpful than no witness at all. This is because, in the case of $P_{M,1}$ or $P_{M,\ell}$, the connected components are expanding and therefore the verifier can easily select a random subset of the points from a single connected component without any assistance from the prover. This can be shown via an application of the expander mixing lemma. Therefore, if a query algorithm exists for distinguishing $P_{M,1}$ and $P_{M,\ell}(F)$, it can be used as a subroutine for distinguishing $P_{M,1}$ and $P_{M,\ell}$ without any witness.

Furthermore, due to Ambainis, Childs, and Liu [6], we know Theorem 25 – i.e. that distinguishing the distributions without witnesses has a query lower bound. Therefore, the problem has a query lower bound even when a set of points F from a connected component are provided:

▶ **Corollary 27.** *For any F_0 with $|F_0| \leq N^{1/100}$, any sufficiently small constant ϵ_1, and any ℓ with $1 < \ell < N^{1/4}$, any quantum query algorithm to ϵ_1-distinguish $P_{M,1}$ and $P_{M,\ell}(F_0)$ must make $\Omega(N^{1/4}/\log N)$ queries.*

Proof. Suppose an algorithm making $q = o(N^{1/4}/\log N)$ queries existed. Then by Lemma 26 there exists an algorithm making $q' = q + O(N^{3/100}) = o(N^{1/4}/\log N)$ queries that distinguishes between $P_{M,1}$ and $P_{M,\ell}$ as well. However, this is impossible by Theorem 25. ◀

The remainder of this section is the proof of Lemma 26.

Let $\mathcal{A}_0$ be the hypothesized algorithm making q_1 queries to ϵ_1-distinguish $P_{M,1}$ and $P_{M,\ell}(F_0)$. We first claim that for any F with $|F| = |F_0|$, there exists an algorithm $\mathcal{A}_1$ that, given as classical input a list of all the vertices in F, and as oracle input an oracle G where G is a sample from either $P_{M,1}$ or $P_{M,\ell}(F)$, can ϵ_1-distinguish between these two cases using q_1 queries to G. The algorithm $\mathcal{A}_1$ is as follows:

1. Given F, compute a permutation π on V that maps F to F_0. (This step is not efficient in terms of runtime, but makes no queries to the oracle G.)
2. Run $\mathcal{A}_0$ with every query to G replaced by a query to $\pi(G)$. Return the answer given by $\mathcal{A}_0$.

The correctness of the algorithm follows from the fact that π maps the distribution $P_{M,\ell}(F)$ exactly to $P_{M,\ell}(F_0)$. Therefore, for all F such that $|F| = |F_0|$,

$$\mathop{\mathbf{E}}_{G \leftarrow P_{M,\ell}(F)} \left[\mathop{\mathbf{Pr}}_{\mathcal{A}_1}[\mathcal{A}_1(F, G) = 1] \right] - \mathop{\mathbf{E}}_{G \leftarrow P_{M,1}} \left[\mathop{\mathbf{Pr}}_{\mathcal{A}_1}[\mathcal{A}_1(F, G) = 1] \right] \geq \epsilon_1. \tag{24}$$

As this holds for all such F,

$$\mathop{\mathbf{E}}_{F} \mathop{\mathbf{E}}_{G \leftarrow P_{M,\ell}(F)} \left[\mathop{\mathbf{Pr}}_{\mathcal{A}_1}[\mathcal{A}_1(F, G) = 1] \right] - \mathop{\mathbf{E}}_{F} \mathop{\mathbf{E}}_{G \leftarrow P_{M,1}} \left[\mathop{\mathbf{Pr}}_{\mathcal{A}_1}[\mathcal{A}_1(F, G) = 1] \right] \geq \epsilon_1. \tag{25}$$

Next, we will show that the input of F can be removed from the algorithm: given just access to G, it is possible to compute a suitable F without making too many queries to the oracle. Specifically, we define the algorithm $\mathcal{A}_2$ to distinguish between $P_{M,1}$ and $P_{M,\ell}$ given only oracle access to G.

1. For a choice of t to be defined later, construct a set F_1 by starting at a random vertex v_0 and taking a $100t \cdot N^{1/100}$-step random walk along the graph as described in Lemma 3. If $|F_1| \geq |F|$, pick the first $|F|$ points from F_1 as the set F'. If not, output 0 (i.e. abort).
2. Run $\mathcal{A}_1$ on input F' with oracle access to G.

We will argue that for an appropriately chosen t, this algorithm achieves the success probability and query complexity claimed in the theorem. To do so, we will argue in two stages.

1. First, we argue that the distribution of F' chosen by random walk is very close to F' chosen uniformly at random from subsets of a connected component of G. This analysis uses the expander mixing lemma.
2. Second, we argue that the distribution over pairs (G, F') obtained after the first step of $\mathcal{A}_2$ is statistically indistinguishable from the distribution over pairs (G, F) sampled by first choosing a uniformly random $F \subseteq V$ and then choosing a random $G \leftarrow P_{M,\ell}(F)$. This will make use of Lemma 11, shown in the appendix of the full version [22] of this paper. By eq. (25), the algorithm $\mathcal{A}_1$ can ϵ_1-distinguish inputs distributed in this manner, and thus the second step of $\mathcal{A}_2$ can ϵ_2-distinguish inputs of $P_{M,1}$ and $P_{M,\ell}$ for ϵ_2 just slightly smaller than ϵ_1.

In our analysis, we will denote probabilities over the distribution of (G, F') generated by $\mathcal{A}_2$ by $\mathop{\mathbf{Pr}}_{\mathcal{A}_2}[\cdot]$ and probabilities over the distribution of (G, F) obtained by first sampling $F \subseteq V$, and then sampling $G \leftarrow P_{M,\ell}(F)$ by $\mathop{\mathbf{Pr}}_{F \text{ then } G}[\cdot]$. The notation $\mathop{\mathbf{Pr}}_{\text{unif}}[\cdot]$ denotes the distribution over F obtained by first picking a uniformly random vertex v in G, and then picking F to be a uniformly random subset of the connected component of G containing v with size $|F_0|$.

7.1 From random walk sampling to uniform sampling

Henceforth, define *expander walk sampling* as the sampling procedure of selecting a uniformly random vertex as the initial vertex v_1, then subsequently taking t steps of a lazy random walk (as defined in the expander mixing lemma, Lemma 3) to choose v_2, and so forth. In this case, the graph and the integer t will be clear from context.

We start by showing a sequence of claims that establish that if the expander walk sampling procedure for generating F' starts in a connected component C of G with size $|C| = K$ and expansion α, then the distribution over sets F' generated by the random walk is close to uniformly sampling points from C. Our main result here will be Claim 30.

▷ Claim 28. Let $\delta = (1 - \alpha/2)^t$ and let r be a natural number with $rK\delta < 1$. The for any sequence of r vertices $v_1, \ldots, v_r$, the probability $\mathop{\mathbf{Pr}}_{\text{unif}}[\cdot]$ that this sequence was obtained by iid random sampling and the probability $\mathop{\mathbf{Pr}}_{\text{walk}}[\cdot]$ that it was obtained by expander walk sampling differ by

$$\left|\mathop{\mathbf{Pr}}_{\text{unif}}[v_1, \ldots, v_r] - \mathop{\mathbf{Pr}}_{\text{walk}}[v_1, \ldots, v_r]\right| \leq \left(\frac{1}{K}\right)^r \cdot \left(rK\delta + (rK\delta)^2 \frac{1}{1 - rK\delta}\right). \tag{26}$$

Proof. Proof is available in the full version [22]. ◁

The following claim will be used to bound the probability that the expander walk sampling procedure aborts, by instead bounding the probability that iid sampling fails to generate enough distinct points.

▷ Claim 29. The probability that $T \geq 100|F|$ iid samples from C contain fewer than $|F|$ distinct vertices is at most $\exp(-T/16)$.

Proof. Proof is available in the full version [22]. ◁

We now combine these two claims and apply them to our setting. Define the event $[F' \leftarrow G]$ if F' is the set of vertices selected from the graph G. Define the distribution $\mathbf{Pr}_{\text{unif}}[\cdot]$ corresponding to first choosing a connected component C with probability proportional to $|C|$, taking r uniform iid samples from the connected component C and setting F' to be the first $|F|$ distinct sampled points. Likewise, define the distribution $\mathbf{Pr}_{\mathcal{A}_2}$ corresponding to choosing a random vertex v in G, taking C to be the connected component containing v, taking r samples according to an expander random walk in C initialized at v with t steps between samples, and then setting F' to be the first $|F|$ distinct sampled points. Set K to be the maximum size of a connected component in G and let $\delta = (1 + \alpha/2)^t$. Then we have the following distance bound between the distributions.

▷ Claim 30. Suppose r, K, δ are such that $rK\delta \leq 10/11$. For any F' of size $|F'| = |F|$, let $\delta_C(F') = \mathbf{Pr}_{\text{unif}}[F' \leftarrow G] - \mathbf{Pr}_{\mathcal{A}_2}[F' \leftarrow G]$. Then

$$|\delta_G(F')| \leq \left(rK\delta + (rK\delta)^2 \frac{1}{1 - rK\delta}\right) \leq 10rK\delta. \tag{27}$$

Moreover, $\mathop{\mathbf{Pr}}_{\mathcal{A}_2}[\text{abort}] \leq 10rK\delta + \exp(-T/16)$.

Proof. Proof is available in the full version [22]. ◁

7.2 From $\mathbf{Pr}_{\mathcal{A}_2}[\cdot]$ to $\mathbf{Pr}_{F \text{ then } G}[\cdot]$

We will now proceed to the main argument showing that the pairs (G, F) sampled by $\mathcal{A}_2$ are distributed close to the distribution expected by $\mathcal{A}_1$.

G has expanding components with high probability

To start off, first note that by Lemma 10, with probability at least $1 - O(N^{-3})$ a graph drawn from $P_{M,\ell}(F)$, for any F of size $\leq N^{1/100}$, will consist of ℓ connected-components which are α-expanders and have size between $[(1-\gamma)z, (1+\gamma)z]$. Since ϵ_1 is a constant, for sufficiently large N, we can restrict to the situation that the graph is of this form and account for this factor in the end. Henceforth set $K_0 = (1+\gamma)z$; we are guaranteed that every component has size at most K_0.

Relating the probabilities

In the case that each connected component is an α-expander, observe that the probability of every valid pair (G, F') is approximately a constant p independent of G and F'. Also recall that the event $F' \leftarrow G$ is the event that F' is the set of vertices selected from G. Moreover, recall the distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ from Lemma 11, and notice that $\mathcal{D}_2$ is exactly the distribution $\mathbf{Pr}_{F \text{ then } G}$ defined above. We define

$$\delta_{1,2}(G, F') \stackrel{\text{def}}{=} \mathop{\mathbf{Pr}}_{\mathcal{D}_1}[(G, F')] - \mathop{\mathbf{Pr}}_{\mathcal{D}_2}[(G, F')], \tag{28}$$

$$\delta_G(F') \stackrel{\text{def}}{=} \mathop{\mathbf{Pr}}_{\mathcal{A}_2}[F' \leftarrow G] - \mathop{\mathbf{Pr}}_{\text{unif}}[F' \leftarrow G]. \tag{29}$$

We will now start with the $\mathbf{Pr}_{\mathcal{A}_2}$ distribution and bound its distance from $\mathbf{Pr}_{F \text{ then } G}$.

$$\mathop{\mathbf{Pr}}_{\mathcal{A}_2}[(G, F')] = \mathop{\mathbf{Pr}}_{P_{M,\ell}}[G] \cdot \mathop{\mathbf{Pr}}_{\mathcal{A}_2}[F' \leftarrow G] \tag{30}$$

$$= \mathop{\mathbf{Pr}}_{P_{M,\ell}}[G] \cdot \left(\mathop{\mathbf{Pr}}_{\text{unif}}[F' \leftarrow G] + \delta_G(F')\right) = \mathop{\mathbf{Pr}}_{\mathcal{D}_1}[(G, F')] + \delta_G(F') \cdot \mathop{\mathbf{Pr}}_{P_{M,\ell}}[G] \tag{31}$$

$$= \mathop{\mathbf{Pr}}_{\mathcal{D}_2}[(G, F')] + \delta_{1,2}(G, F') + \delta_G(F') \cdot \mathop{\mathbf{Pr}}_{P_{M,\ell}}[G] \tag{32}$$

$$= \mathop{\mathbf{Pr}}_{F \text{ then } G}[(G, F')] + \delta_{1,2}(G, F') + \delta_G(F') \cdot \mathop{\mathbf{Pr}}_{P_{M,\ell}}[G]. \tag{33}$$

We may now bound the total variational distance between the two sides.

$$\left\| \mathop{\mathbf{Pr}}_{\mathcal{A}_2}[\cdot] - \mathop{\mathbf{Pr}}_{F \text{ then } G}[\cdot] \right\| = \frac{1}{2} \sum_{(G,F')} \left| \mathop{\mathbf{Pr}}_{\mathcal{A}_2}[(G, F')] - \mathop{\mathbf{Pr}}_{F \text{ then } G}[(G, F')] \right| \tag{34}$$

$$\leq \frac{1}{2} \sum_{(G,F')} \left(|\delta_{1,2}(G, F')| + \mathop{\mathbf{Pr}}_{P_{M,\ell}}[G] \cdot |\delta_G(F')| \right) \tag{35}$$

$$\leq \|\mathcal{D}_1 - \mathcal{D}_2\| + \frac{1}{2} \binom{K_0}{|F'|} \cdot \max_{G,F'} |\delta_G(F')| \tag{36}$$

$$\leq 3N^{-9/200} + K_0^{|F|} \cdot (10 r K_0 \delta) \tag{37}$$

$$= 3N^{-9/200} + ((1+\gamma)z)^{|F|+1} \cdot (10 \cdot (100|F|) \cdot (1-\alpha/2)^t) \tag{38}$$

$$\leq 3N^{-9/200} + \left(2N^{9/10}\right)^{N^{1/100}+1} \cdot 1000 N^{1/100} \cdot (1-\alpha/2)^t \tag{39}$$

$$= 3N^{-9/200} + 2000 \cdot 2^{N^{0.01}} \cdot N^{0.9N^{0.01}+0.91} \cdot (1-\alpha/2)^t. \tag{40}$$

A total distance bound of $O(N^{-9/200})$ can be achieved if

$$2^{N^{0.01}} \cdot N^{0.9N^{0.01}+0.91} \cdot (1-\alpha/2)^t \leq N^{-9/200} \tag{41}$$

$$N^{0.01} + (0.9N^{0.01} + 0.91) \cdot \log N + t \cdot \log(1-\alpha/2) \leq -\frac{9}{200} \log N \tag{42}$$

$$\left(N^{0.01}\left(\frac{1}{\log N}+0.9\right)+0.955\right)\frac{\log N}{\log(1/(1-\alpha/2))}\leq t. \tag{43}$$

So setting $t=\Theta(N^{0.02})$ is sufficient.

Given this choice of t, let us know calculate the chance that the sampling of F' aborts. By Claim 30, this is at most $10rK\delta+\exp(-T/16)=O(N^{-9/200})+\exp(-100N^{0.01}/16)=O(N^{-9/200})$

Thus, the total error probability of $\mathcal{A}_2$ equals the error probability of $\mathcal{A}_1$ up to

$$\underbrace{O(N^{-3})}_{\text{Sample a bad graph}}+\underbrace{O(N^{-9/200})}_{\text{Changing }\mathcal{A}_2\text{ to }F\text{ then }G}+\underbrace{O(N^{-9/200})}_{\text{Sampling }F'\text{ aborts}}, \tag{44}$$

yielding $\epsilon_2=\epsilon_1-O(N^{-9/200})$ as claimed in theorem. And the total query complexity assuming not aborting can be calculated as follows. Recall that t was chosen to be $\Theta(N^{0.02})$. The total number of additional queries over $\mathcal{A}_1$ is thus the number of steps in the walk which is $100N^{1/100}\cdot t\leq O(N^{0.03})$. Thus, this algorithm has total query complexity $q+O(N^{0.03})$ and distinguishes with probability ϵ_2 as claimed.

8 Wrapping up the proof of Theorem 1

First, we need to note that the distributions B_S and $P_{M,1}$ which we used as proxies for YES and NO instances are not fully supported on YES and NO instance graphs, respectively. However, they are very close. For every $S\subset[N]$ of size ζ, let $\widetilde{B}_S$ be the restriction of the distribution B_S (defined in Section 6.2) to graphs with ℓ connected components each consisting of between $(1-\gamma)z$ and $(1+\gamma)z$ vertices. By Corollary 17, the statistical distance between B_S and $\widetilde{B}_S$ is $O(N^{-3})$. The YES instances for Theorem 1 are the $\{\widetilde{B}_S\}$.

We consider a single NO instance of $\widetilde{P}_{M,1}$ where $\widetilde{P}_{M,1}$ is the restriction of $P_{M,1}$ to graphs which are α-expanders. The statistical distance between these two distributions is $O(N^{-3})$ by Lemma 10.

Furthermore, we can verify that the supports of $\widetilde{P}_{M,1}$ and $\widetilde{B}_S$ are far apart in Hamming distance. Consider graphs G_1 and G_ℓ from either support, respectively. Consider a connected component C from G_ℓ. In the graph G_ℓ, all the edges on C stay within G_ℓ, but since G_1 is an α-expander and also a $1/10^2$-edge expander (see proof of Lemma 10)), then in G_1, a $\geq 1/10^4$ fraction of the edges emanating from C leave C. As this holds for all components C since $|C|\ll N/2$, then the Hamming distance between the adjacency lists of G_1 and G_ℓ is $\Omega(N)$. As this holds for all graphs G_1 and G_ℓ, then the Hamming distance bound between the supports hold.

8.1 QMA algorithm

For completeness, from Corollary 17, we know that the algorithm $\mathcal{A}_{\mathsf{QMA}}$ with witness state $|S\rangle$ answers distribution $\widetilde{B}_S$ with probability at least $\geq 1-O(N^{-1/20})$. For soundness, from Lemma 10, we know that $\widetilde{P}_{M,1}$ is an $1/(2\cdot 10^8)$-expander with probability $\geq 1-O(N^{-3})$. Therefore, by Lemma 12, the algorithm $\mathcal{A}_{\mathsf{QMA}}$ accepts with probability at most

$$\leq 1-\frac{1}{4}\cdot\frac{1}{(2\cdot 10^8)}+O(N^{-3})\leq 1-\frac{1}{9\cdot 10^8}. \tag{45}$$

By parallel repetition $9\cdot 10^6=O(1)$ times, we yield a quantum algorithm with ≤ 0.01 soundness.

8.2 QCMA algorithm

We argue now that any QCMA algorithm with completeness 0.99 and soundness 0.01 either requires an exponentially long proof or an exponential number of quantum queries. This is done by arguing that any algorithm with a short proof and few queries cannot have such a large completeness and soundness gap. Assume, therefore, that there exists a QCMA algorithm with a

$$q \leq \frac{n \cdot N^{1/100}}{2000}\text{-bit proof and } f = O(N^{1/50}) \text{ quantum queries} \tag{46}$$

and a completeness and soundness gap of ≥ 0.98. By Lemma 18, there exists a

$$\left(\frac{1}{100}, \zeta, \frac{2000q}{n}\right)\text{-sunflower } ❀ \tag{47}$$

with core F and a f-query deterministic quantum algorithm $\mathcal{A}$ that accepts each distribution $\widetilde{B}_S$ for $S \in ❀$ with probability ≥ 0.99 and accepts $\widetilde{P}_{M,1}$ with at most ≤ 0.01 probability. It also accepts that accepts each distribution B_S for $S \in ❀$ with probability $\geq 0.99 - O(N^{-3})$ and accepts $P_{M,1}$ with at most $\leq 0.01 + O(N^{-3})$ probability. Then with the assumed number of queries, we can apply Lemma 19 with $\delta = 1/10$ to argue that $\mathcal{A}$ must accepts the distribution H_{Ω_F} with probability $\geq 0.09 - O(N^{-3})$. Next, by Lemma 24, $\mathcal{A}$ must accept the distribution $P_{M,\ell}(F)$ with probability $\geq 0.09 - 2 \cdot O(N^{-3}) \geq 0.08$. We conclude by applying Corollary 27. Therefore, $\mathcal{A}$ must accept the distribution $P_{M,1}$ with probability > 0.02, a contradiction.

9 Concluding remarks

9.1 Relation to the Fefferman and Kimmel [13] construction

One can think of the result stated in this work as applying the QCMA lower bounding techniques developed by Fefferman and Kimmel [13] to the expander distinguishing problem originally studied by Ambainis, Childs, and Liu [6].

At a high level, in the *in-place permutation oracle* QMA and QCMA separation of [13], the goal was to distinguish between permutations $\pi : [N] \to [N]$ such that $\pi^{-1}([\sqrt{N}])$ is mostly (2/3) supported on odd numbers from permutations mostly supported on even numbers. The original idea in Fefferman and Kimmel was that if the oracle π was provided as a classical oracle (an Nn-bit list $[\pi(1), \pi(2), \ldots, \pi(N)]$) then the subset state $|\xi_{\text{ideal}}\rangle = |\pi^{-1}([\sqrt{N}])\rangle$ would be a good quantum witness. By measuring the last qubit of a witness $|\xi\rangle$, the verifier can decide if the set $\pi^{-1}([\sqrt{N}])$ is supported mostly on either odd numbers or even numbers. What remains to verify is that the witness $|\xi\rangle$ provided is indeed $|\xi_{\text{ideal}}\rangle$. The hope would be to use the oracle for π to verify the statement as the sate $|[\sqrt{N}]\rangle$ can be easily verified by measuring in the Hadamard basis.

However, due to the index-erasure problem, a classical oracle for verifying that $|\xi\rangle = |\xi_{\text{ideal}}\rangle$ would need to allow implementation of both π and π^{-1}. However, if the oracle π^{-1} is provided, then there is a BQP algorithm for this problem. Simply, pick a random $j \in [\sqrt{N}]$ and then check if $\pi^{-1}(j)$ is odd or even. The solution in [13] was to define the oracle instead as an "in-place oracle" for π, meaning a unitary defined as $\sum_j |\pi(j)\rangle\langle j|$. Then the verifier can verify that $|\xi\rangle = |\xi_{\text{ideal}}\rangle$ and yet the BQP algorithm no longer holds.

Fefferman and Kimmel had to make one more modification to prove a QMA and QCMA oracle separation: they considered distributions over in-place oracles which mapped to the same ideal quantum witness $|\xi_{\text{ideal}}\rangle$. This was because it seems to be beyond current techniques

to prove classical lower bounds without forcing a large structured set of permutations to all share the same witness – otherwise, for all we know, there might be a mathematical fact about permutations which yields a short classical certificate for any individual permutation. So the oracle is defined as a distribution over unitaries – i.e. a completely positive trace preserving (CPTP) map.

Notice that this work takes much inspiration form [13]; the quantum witnesses for both our work and [13] are subset states and we also consider distributions over oracles with the same (or similar) ideal quantum witness. This is because we are unsure how to prove that there is no property of a specific regular graph which yields a short classical witness. We elaborate on why such an impossibility result is hard to prove in the next subsection. What our result principally improves on is that the underlying oracle can be a classical string instead of a unitary.

9.2 Difficulties in proving stronger statements

Recall that our QMA upper bound does not require the setup of distributions over oracles – it was only included to prove the QCMA lower bound. How much harder is it (or is it even possible) to prove a QCMA lower bound without considering distributions?

As pointed out to us by William Kretschmer [20], if one considers *average-case* algorithms instead of *worst-case* algorithms, then this problem is $\in \mathsf{RNP}^G$, the average-case analog of NP^G. This is because the average-case version of the expander distinguishing problem is to distinguish the distributions $P_{M,\ell}$ and $P_{M,1}$. And there is a simple randomized algorithm for this problem with a classical witness. Let us recall that for a d-regular graph, the expected number of triangles in a connected component is $\Theta(d^3)$ independent of the number of vertices in the component. A similar analysis can be done for $P_{M,\ell}$ and $P_{M,1}$, to show that a random graph from $P_{M,\ell}$ has $\Theta(\ell d^3)$ triangles whereas $P_{M,1}$ has $\Theta(d^3)$ triangles. Therefore, a *classical* witness for the statement that the graph (with high probability) is drawn from $P_{M,\ell}$ (instead of $P_{M,1}$) is a list of $100 \cdot \Theta(d^3)$ triangles from the graph. This witness is easily verifiable and correctly distinguishes with high probability.

Notice that this RNP^G algorithm does not solve the expander distinguishing problem in the worst-case; since graphs exist in both distributions which are triangle-free (with constant probability). Furthermore, it cannot distinguish the distributions considered in Theorem 1 because the proof relies on finding triangles which is property of the graph not deducible from only knowing the connected components.

But it does highlight a principal roadblock in extending Theorem 1 to distinguishing oracles that are not distributions. It is entirely possible that there exists a property of graphs revealed by looking at the edges that distinguishes graphs with many connected components from graphs with a single expanding connected component. To the best of our knowledge, we do not know of any such property but proving that none exist is beyond the techniques shown here.

Lastly, if we consider the expander distinguishing problem when in the YES case we are promised that every connected component has size at most $0.99N$, then this problem is in coAM^G. When the graph is a NO instance, the verifier can select two random points and the prover can always find a path of length $O(\log N) = O(n)$ between the two. However, when the graph is disconnected and no component is too big, with probability $\geq 1/50$, no path exists.

Therefore, our constructed oracle very finely separates the classes QMA and QCMA in the sense that small perturbations of the problem might be very easy.

References

1 Scott Aaronson. On perfect completeness for QMA. *Quantum Information & Computation*, 9(1):81–89, 2009. arXiv:0806.0450.

2 Scott Aaronson. Open problems related to quantum query complexity. *ACM Transactions on Quantum Computing*, 2(4), December 2021. doi:10.1145/3488559.

3 Scott Aaronson and Greg Kuperberg. Quantum versus classical proofs and advice. In *Twenty-Second Annual IEEE Conference on Computational Complexity (CCC'07)*, pages 115–128, 2007. doi:10.1109/CCC.2007.27.

4 Dorit Aharonov and Tomer Naveh. Quantum NP – A survey, 2002. arXiv:arXiv:quant-ph/0210077.

5 Andris Ambainis. Quantum lower bounds by quantum arguments. *J. Comput. Syst. Sci.*, 64(4):750–767, June 2002. doi:10.1006/jcss.2002.1826.

6 Andris Ambainis, Andrew M. Childs, and Yi-Kai Liu. Quantum property testing for bounded-degree graphs. In Leslie Ann Goldberg, Klaus Jansen, R. Ravi, and José D. P. Rolim, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 365–376, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

7 Anurag Anshu, Nikolas Breuckmann, and Chinmay Nirkhe. NLTS hamiltonians from good quantum codes, 2022. arXiv:2206.13228.

8 Atul Singh Arora, Alexandru Gheorghiu, and Uttam Singh. Oracle separations of hybrid quantum-classical circuits. *CoRR*, 2022. arXiv:2201.01904.

9 Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh V. Vazirani. Strengths and weaknesses of quantum computing. *SIAM J. Comput.*, 26(5):1510–1523, 1997. doi:10.1137/S0097539796300933.

10 Adam D. Bookatz. QMA-complete problems. *CoRR*, 2012. doi:10.48550/arXiv.1212.6312.

11 Sergey Bravyi, David P. Divincenzo, Roberto Oliveira, and Barbara M. Terhal. The complexity of stoquastic local hamiltonian problems. *Quantum Info. Comput.*, 8(5):361–385, May 2008.

12 Sergey Bravyi and Barbara Terhal. Complexity of stoquastic frustration-free hamiltonians. *SIAM J. Comput.*, 39(4):1462–1485, November 2009.

13 Bill Fefferman and Shelby Kimmel. Quantum vs. classical proofs and subset verification. In Igor Potapov, Paul G. Spirakis, and James Worrell, editors, *43rd International Symposium on Mathematical Foundations of Computer Science, MFCS 2018, August 27-31, 2018, Liverpool, UK*, volume 117 of *LIPIcs*, pages 22:1–22:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.MFCS.2018.22.

14 Honghao Fu. Personal Communication, October 2022.

15 Sandy Irani, Anand Natarajan, Chinmay Nirkhe, Sujit Rao, and Henry Yuen. Quantum search-to-decision reductions and the state synthesis problem, 2021. arXiv:2111.02999.

16 Stephen P Jordan, David Gosset, and Peter J Love. Quantum-merlin-arthur-complete problems for stoquastic hamiltonians and markov matrices. *Physical Review. A*, 81(3), March 2010. doi:10.1103/PHYSREVA.81.032331.

17 A.Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003. doi:10.1016/S0003-4916(02)00018-0.

18 Hartmut Klauck and Supartha Podder. Two results about quantum messages. In *International Symposium on Mathematical Foundations of Computer Science*, pages 445–456. Springer, 2014.

19 Dexter Kozen. Lower bounds for natural proof systems. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, SFCS '77, pages 254–266, USA, 1977. IEEE Computer Society. doi:10.1109/SFCS.1977.16.

20 William Kretschmer. Personal Communication, July 2022.

21 Andrew Lutomirski. Component mixers and a hardness result for counterfeiting quantum money, 2011. arXiv:1107.0321.

22 Anand Natarajan and Chinmay Nirkhe. A distribution testing oracle separation between QMA and QCMA. *CoRR*, 2023. arXiv:2210.15380.

23 Chinmay Nirkhe. *Lower bounds on the complexity of quantum proofs*. PhD thesis, EECS Department, University of California, Berkeley, November 2022. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2022/EECS-2022-236.html.

24 Chinmay Nirkhe. NLTS Hamiltonians from codes, 2022. Simons Institute for the Theory of Computing Quantum Colloquium. Panel Umesh Vazirani, Dorit Aharanov, Matthew Hastings, Anand Natarajan, and Chinmay Nirkhe. URL: https://simons.berkeley.edu/events/quantum-colloquium-nlts-hamiltonians-codes.

Translationally Invariant Constraint Optimization Problems

Dorit Aharonov
Department of Computer Science and Engineering, Hebrew University, Jerusalem, Israel

Sandy Irani
Department of Computer Science, University of California Irvine, CA, USA
The Simons Institute for the Theory of Computing, University of California Berkeley, CA, USA

Abstract

We study the complexity of classical constraint satisfaction problems on a 2D grid. Specifically, we consider the computational complexity of function versions of such problems, with the additional restriction that the constraints are *translationally invariant*, namely, the variables are located at the vertices of a 2D grid and the constraint between every pair of adjacent variables is the same in each dimension. The only input to the problem is thus the size of the grid. This problem is equivalent to one of the most interesting problems in classical physics, namely, computing the lowest energy of a classical system of particles on the grid. We provide a tight characterization of the complexity of this problem, and show that it is complete for the class $\mathrm{FP}^{\mathrm{NEXP}}$. Gottesman and Irani (FOCS 2009) also studied classical constraint satisfaction problems using this strong notion of translational-invariance; they show that the problem of deciding whether the cost of the optimal assignment is below a given threshold is NEXP-complete. Our result is thus a strengthening of their result from the decision version to the function version of the problem. Our result can also be viewed as a generalization to the translationally invariant setting, of Krentel's famous result from 1988, showing that the function version of SAT is complete for the class $\mathrm{FP}^{\mathrm{NP}}$.

An essential ingredient in the proof is a study of the computational complexity of a gapped variant of the problem. We show that it is NEXP-hard to approximate the cost of the optimal assignment to within an additive error of $\Omega(N^{1/4})$, where the grid size is $N \times N$. To the best of our knowledge, no gapped result is known for CSPs on the grid, even in the non-translationally invariant case. This might be of independent interest. As a byproduct of our results, we also show that a decision version of the optimization problem which asks whether the cost of the optimal assignment is odd or even is also complete for $\mathrm{P}^{\mathrm{NEXP}}$.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases Constraint satisfaction, Tiling, Translational-invariance

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.23

Related Version *Full Version*: `https://arxiv.org/abs/2209.08731`

Acknowledgements We are grateful to the Simons Institute for the Theory of Computing, at whose program on the "The Quantum Wave in Computing" this collaboration began.

1 Introduction

More than half a century ago, Cook and Levin inaugurated the field of NP-completness. The fact that the Constraint Satisfaction Problem (CSP) is NP-complete has been the cornerstone of our understanding and approach to important optimization problems arising in countless applications. However, the importance of CSP and its NP-completeness stems not only from its central role in studying the complexity of optimization problems; in fact, the computational complexity of CSP is of major importance to physics as well.

In classical many body physics, the most basic notion is the local Hamiltonian, which expresses the total energy of a system of particles. It turns out that this local Hamiltonian can be viewed as a CSP. The energy of the system is written in such a Hamiltonian as the

sum of terms, each of which describes the energy interaction between constant-sized clusters of particles. These terms can be viewed as local constraints and finding whether the lowest energy state of such a system is below a given threshold or above it, is a special case of CSP. The classical local Hamiltonian problem was famously shown to be NP-complete in many cases by Barahona and others [6, 13], and the understanding of its complexity for a variety of Hamiltonians such as the Ising model, the Potts model, and more, has had a fundamental impact in several research areas in physics, including statistical mechanics and mathematical physics.

The theory of NP-completeness also has a natural generalization to the quantum setting. Like classical NP-completeness, the study of quantum NP-completeness has had a tremendous impact on our understanding of the relevant field in physics, namely condensed matter physics. More specifically, the Cook-Levin theorem was generalized by Kitaev [14] roughly 25 years ago to show that the following problem is QMA-complete: Given a local Hamiltonian with quantum energy interactions describing the energy in a quantum many-body system, decide whether the ground energy is above some value or below another. This problem had been intensely studied in recent years [14, 9, 16, 2]; importantly, over the past two decades, the study of local Hamiltonians and their computational complexity has led to the birth of a new field called "Hamiltonian complexity", which studies problems related to condensed matter physics, through the computational lens [9]. Both in classical statistical mechanics as well as in condensed matter and many-body quantum physics, the importance of the computational perspective has become one of the fundamental underpinnings of research today.

From the physics point of view, however, the general CSP setting commonly studied in computer science, in which each constraint is specified separately and independently based on the particular optimization problem of interest, seems quite contrived. By and large, physicists study local Hamiltonians, be them classical or quantum, in a *translational-invariant* (TI) setting. In this scenario, the particles are located at the vertices of a geometric lattice and all the terms acting on adjacent pairs of particles along a particular dimension are *the same*. Quantum and classical Hamiltonians are used to model the energy interactions of particles in a material. If the material is uniform, then it is natural that the energy interaction between particles would be the same throughout. Thus, in order for a theory of CSPs to be relevant in physics, it must consider this translational-invariance requirement, which introduces many new complexity challenges.

The model most relevant to physics is TI in a very strong sense: the dimension of the individual particles and the Hamiltonian term acting on each pair of adjacent particles in a lattice are fixed parameters of the problem. When considering finite systems, the only input is an integer N indicating the size of the system. This set-up corresponds to the fact that in physics, different Hamiltonians represent completely different physical systems. For example, studying the ground energy (or some other quantity) in the so-called AKLT model [1] is considered to be a completely different problem than studying the same quantity in, say, the Ising model[1].

Important progress on the computational complexity of translationally-invariant CSPs has been made in recent years. In particular, Gottesman and Irani [12] studied TI CSPs both in the classical and in the quantum setting, and showed hardness results in both cases. Since the size of the grid can be given by logarithmically many bits, and there is no other input, one encounters an exponential factor compared to the standard version of CSP. Thus

[1] Some of the recent work on the quantum TI local Hamiltonian problem [7] adopt a weaker notion in which the input also includes the Hamiltonian term that is applied to each pair of particles, allowing the Hamiltonian to be tuned to the size of the system. This model is mainly considered in quantum Hamiltonian complexity, but have not been a topic of study in physics.

the results in [12] show NEXP- and QMA_{EXP}-completeness for the classical and quantum variants of the problem, respectively. A tightly related line of work studies TI infinite systems [3, 21, 8] and considers computability and computational complexity in that domain, namely in the so-called thermodynamic limit. Although the focus in the current work is on finite systems, constructions for finite systems have played an important role for the results in the thermodynamic limit. In particular, all the results in [3, 21, 8] use a finite construction layered on top of a certain type of aperiodic tiling of the infinite grid.

However, all the results mentioned above have studied a *decision version* of the CSP problem. In contrast, in classical as well as in quantum physics, when considering the local Hamiltonian, the main problem of interest is the problem of finding the lowest possible energy for the Hamiltonian over all possible states – namely, the ground energy, which is one of the most important notions in physics. Thus, when considering CSPs with a physics motivation in mind, it seems that the *function version* of the CSP is a more relevant version of the problem than the decision version. In this setting, the question is not whether all constraints can be satisfied, but what is the *maximum number* of constraints that can be satisfied by any assignment, or, in a weighted variant, what is the cost of the optimal assignment which minimizes the weighted sum of violated constraints. We note that computing the cost of the optimal solution is in fact also the more natural version of CSPs in many combinatorial applications (to give just two examples, max-cut and max independent set).

What is known about the computational complexity of the function version of CSPs? In 1988, Krentel [15] proved that the function problem for constraint satisfaction is FP^{NP}-complete. Krentel's proof is significantly more involved technically than that of the Cook-Levin's theorem which characterizes the complexity of the decision variant of CSPs. In stark contrast to the theory of decision problems and NP-completeness, the function version of CSP seems to have received significantly less attention in the TCS literature.

In particular, to the best of our knowledge, the computational complexity of function CSPs in the TI setting, has remained open. In this paper we provide a tight characterization of its complexity, and show that the function version of TI CSP on a 2-dimensional grid is complete for FP^{NEXP}. This result thus strengthens Krentel's construction for general CSPs to apply even TI systems for two and higher dimensions. The result is also a generalization of Gottesman-Irani who prove hardness for 2D TI systems for the standard decision problem, where one only needs to determine if the ground energy is below a given threshold. One of the key technical challenges in our result is to effectively create large ($\Theta(N^{\epsilon})$) costs on an $N \times N$ grid using only two constant-sized terms which apply one in the horizontal and one in the vertical direction. Thus, as a stepping stone to the more complex result for the function version of TI 2D CSPs, we show a fault-tolerant result which we believe is of interest on its own, namely that it is NEXP-complete to even approximate the ground energy of 2D TI CSPs to within an additive $\Theta(N^{1/4})$.

2 Problem Definitions, Results and Main Challenges

It is most convenient to present our results using the language of the weighted tiling problem, where we focus here on the two dimensional case[2]. In this tiling problem, one is asked to tile an $N \times N$ 2D grid with a set of 1×1 tiles. The tiles come in different colors and only some pairs of colors can be placed next to each other in either the horizontal or vertical directions. More precisely, a set of tiling rules $\mathcal{T}$ is a triple (T, δ_H, δ_V), where T is a finite set of tile *types*

[2] Our version of tiling is equivalent to the more common Wang tiles [19].

$T = \{t_1, \ldots, t_d\}$, and δ_H and δ_V are functions from $T \times T$ to $\mathbb{Z}$. For $(t, t') \in T \times T$, $\delta_h(t, t')$ is the cost of putting a tile of type t immediately to the left of a tile of type t' and $\delta_v(t, t')$ is the cost of putting a tile of type t immediately above a tile of type t'. Let $\lambda_0(\mathcal{T}(N))$ be the minimum cost of tiling an $N \times N$ grid with tiling rules $\mathcal{T}$. The goal is to tile the grid with minimal total cost. Note that this problem is directly analogous to a classical Hamiltonian in 2D. We first define a function version of the problem.

▶ **Definition 1.** $\mathcal{T}$-FWT (Function Weighted Tiling)
Input: *An integer N specified with $\lfloor \log N + 1 \rfloor$ bits*
Output: $\lambda_0(\mathcal{T}(N))$

▶ **Theorem 2. (Main)** *There exists a set of tiling rules $\mathcal{T}$ such that $\mathcal{T}$-FWT is FP^{NEXP}-complete.*

We note that the fact that the function problem is complete for 2D immediately implies that it is complete for any grid of dimension at least 2 since the 2D construction can be embedded into a higher dimensional grid. The 1D CSP case is poly-time computable using dynamic programming.

The upper bound in Theorem 2 is easy: it can be achieved by binary search with access to an oracle for the decision problem. For the lower bound, one encounters a challenge. The reduction must encode in the tiling rules the computation of a polynomial time TM (TM) with access to a NEXP oracle. If an instance given to the oracle is a *yes* instance, the computation of the verifier can be encoded into the tiling rules. However *no* instances cannot be directly verified in this way. Krentel's proof that the function problem of weighted SAT is $\mathrm{FP}^{\mathrm{NP}}$-complete [15] overcomes this challenge; let us recall it and then explain the problem in carrying it over to the TI setting. Krentel uses an accounting scheme [15, 17] that applies a cost to every string z representing guesses for the sequence of responses to all the oracle queries made. The accounting scheme needs to ensure that the minimum cost z is equal to the correct sequence of oracle responses, $\tilde{z}$. *yes* and *no* guesses are treated differently, due to the fact that the verifier can check *yes* instances (and thus incorrect *yes* guesses can incur a very high cost), but *no* guesses, cannot be directly verified. In Krentel's scheme, *no* guesses incur a more modest cost, whether correct or not, and their cost must decrease exponentially. This is because the oracle queries are adaptive; an incorrect oracle response could potentially change all the oracle queries made in the future and so it is important that the penalty for an incorrect guess on the i^{th} query is higher than the cost that could potentially be saved on all future queries. The weights on clauses that implement this accounting scheme are multiplied by a large power of two to ensure that they are the dominant factor in determining the optimal assignment.

The difficulty in applying Krentel's accounting scheme in the TI setting is that the costs must grow with the size of the input. Therefore, it is not possible to apply the costs directly into the tiling rules which are of fixed constant size. A natural attempt to circumvent the problem is to assign the required large penalty by many tiles, each of which would acquire a constant penalty; however, the problem in implementing this approach is that Cook-Levin type reductions from computations to tilings are very brittle, as a single error can potentially derail the entire computation. For example, imagine inserting a row that does not have a TM head. There will be a single fault where the head disappears from one row to the next, but every row thereafter will contain the unchanging contents of the TM tape without a head to execute a next step. This imposes a challenge since when enforcing large costs by using many tiles, or constraints, we need to make sure that many of these constraints are indeed violated in order to incur the required large penalty.

We provide a construction which circumvents this issue by exhibiting some *fault tolerance* properties. We thus prove what can be viewed as a gapped version or a hardness of approximation result, which is then a natural stepping stone to implementing the more intricate function required in Krentel's accounting scheme. To this end we define an approximation version of weighted tiling:

▶ **Definition 3.** $(\mathcal{T}, f)$-GWT (Gapped Weighted Tiling)
Input: *An integer N specified with $\lfloor \log N + 1 \rfloor$ bits. Two integers a and b such that $b - a \geq f(N)$.*
Output: *Determine whether $\lambda_0(\mathcal{T}(N)) \leq a$ or $\lambda_0(\mathcal{T}(N)) \geq b$.*

▶ **Theorem 4.** *There exists a set of tiling rules $\mathcal{T}$ such that $(\mathcal{T}, f)$-GWT is NEXP-complete for a function $f(n) = \Omega(N^{1/4})$.*

This shows that it is NEXP-hard to even approximate the cost of the optimal tiling to within an additive error that is $\Omega(N^{1/4})$. This can be viewed as a gapped version of the results of [12]; the proof constructs a reduction mapping the computation into a tiling such that even in the presence of $O(N^{1/4})$ faults, the computation encoded by the tiling is able to proceed and produce approximately correct results.

Theorem 4 is of potential interest on its own. It might resemble a PCP type result, but the model we consider differs from the standard PCP setting in two ways: the first is that the underlying graph is a grid, rather than a graph with much higher connectivity, and the second is translational-invariance. It is not possible to obtain a hardness of approximation result with an additive error that is linear in N (as one has in the PCP theorem) on any finite dimensional lattice because such graphs do not have the necessary expansion properties. For example, in 2D, one could divide the grid into $b \times b$ squares for $b = \Theta(\sqrt{\log N})$ and solve each square optimally in polynomial time. The resulting solution would be within an additive $N/\sqrt{\log N}$ of the optimal solution. To the best of our knowledge, no gapped version was proven before for CSP problem set on a constant dimensional grid, even without the TI restriction.

Finally, our results provide tight characterizations of the complexity of the following decision problem;

▶ **Definition 5.** $\mathcal{T}$-PWT (Parity Weighted Tiling)
Input: *An integer N specified with $\lfloor \log N + 1 \rfloor$ bits*
Output: *Determine whether $\lambda_0(\mathcal{T}(N))$ is odd or even.*

The proof is very similar to the proof of Theorem 2. The result on Parity Weighted Tiling illustrates that decision problems related to CSP can be complete for an oracle class just like the function problem. The crucial difference between the threshold decision problem (is the cost of the optimal solution less than t?) which is NEXP-complete and the parity problem which is $\mathrm{P}^{\mathrm{NEXP}}$-complete is that the parity problem still seems to require determining the optimal cost. This seems to make the characterization of its complexity as challenging as for the function version of the problem.

Organization. We next proceed to an overview of the proofs. We start with the setup of tiling rules and layers in Section 3. Overviews of the proofs of the Theorems are given in subsection 4. We end with related work and open questions in Section 5. The complete proofs are given in the full version of the paper [4].

3 Tiling Rules and Layers

We assume that there is a special tile denoted by □ which must be placed around the perimeter of the grid to be tiled. Moreover, no □ tile can be placed in the interior of the grid. We will return later to enforcing this condition in the context of the different problems. The tiles on the interior will be composed of multiple layers where each layer has its own set of tile types. A tile type for an internal tile in the overall construction is described by a tile type for each of the layers.

For ease of exposition, we allow our tiling rules to also apply to local *squares* of four tiles. This can easily then be translated to two-local constraints on tiles, as in our definition of the tiling problem. This simple transition is described in more detail in the full version. For the remainder of the paper our tiling rules include constraints on local squares of four tiles, as well as pairs of horizontal tiles.

If the four tiles in a square are all interior tiles, then each possible pattern of four square tiles within a layer will be designated as legal or illegal. The overall cost of placing four interior tiles in a local square together will be function of whether the square for each layer is legal or illegal. For the Gapped Weighted Tiling, the cost will be just the number of layers for which the square pattern is illegal. For the Function Weighted Tiling and Weighted Tiling Parity, illegal squares at different layers will contribute different amounts to the cost.

In general, a no-cost tiling of each Layer represents a computational process where each row represents the state of a TM. The computation reverses direction from one layer to the next. The rows of a tiling of an $N \times N$ grid will be numbered r_0 through r_{N-1} from bottom to top. When referring to the rows in a particular layer, we will exclude the border rows and order the rows according to the computation direction. So the first row of Layer 1, which proceeds from bottom to top, is row r_1 and the last row of Layer 1 is r_{N-2}. Layer 2 proceeds from top to bottom, so the first row for Layer 2 is r_{N-2} and the last row is r_1.

For the most part, the rules governing the tiling apply to the tile types within each individual layer. The different layers only interact at the lower and upper border of the grid. This is how the output of one process (on Layer i) is translated into the input for the next process (on Layer $i+1$). For example, a square may be illegal if the two lower tiles are □ □, and the two upper tiles violate certain constraints between the Layer i and Layer $i+1$ types. Some of the layers will also have additional constraints on which tiles can be next to each other in the horizontal direction. Each type of violated constraint is given a name described below.

▶ **Definition 6** (Faults in a Tiling). *An occurrence of any of the illegal patterns described in the constructions is called a* fault. *A tiling with no faults, will correspond to a* fault-free *computation.*

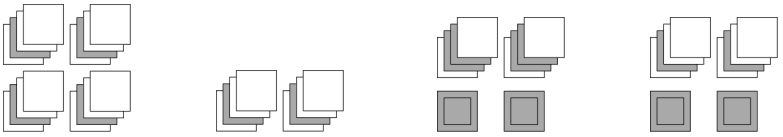
There will be some additional costs (described later) associated with a computation ending in a rejecting state. These are not considered faults because they can happen in correct computations. Figure 1 illustrates the different types of tiling constraints.

Illegal Computation Squares: For each layer, every pattern of four tile types will be designated as a *legal computation square* or an *illegal computation square*. In general, these rules enforce that the tiling within the layer represents a consistent execution of a TM. The full version of the paper [4] gives a set of rules to translate the rules of a TM into legal and illegal computation squares.

Illegal Pairs: Some of the layers will have additional constraints on which tiles can be placed next to each other in the horizontal direction. Each ordered pair of tiles types for that layer will be designated as a *legal pair* or an *illegal pair*.

Illegal Initialization Squares: For each layer, there are also some initialization rules that constrain the initial configuration of the TM. If the layer runs bottom to top, then these rules apply to r_0, which consists of all $\square$ tiles, and the first row of the layer. For example, if tile t_1 can not be immediately to the left of t_2 in the first row of Layer i, then the square with $\square$ $\square$ directly below t_1 t_2 is an *illegal initialization square* for Layer i. If the TM for the layer runs top to bottom, then the square with $\square$ $\square$ directly above t_1 t_2 in Layer i is illegal.

Illegal Translation Squares: Finally, we add rules that control how the last row of Layer i is translated to the first row of Layer $i+1$. If Layer i runs top to bottom, then the rules apply to rows r_0 and r_1. For example, if tile t in Layer i cannot be translated to t' in Layer $i+1$, then any square with a $\square$ directly below a tile whose Layer i type is t and whose Layer $i+1$ type is t' would be illegal. The translation rules can also apply to pairs of adjacent tiles. E.g., it could be illegal to have a square whose bottom two tiles are $\square$ $\square$ and whose top two tiles have t_1 t_2 in Layer i and t_3 t_4 in Layer $i+1$.



(a) Computation. **(b)** Illegal pairs. **(c)** Translation. **(d)** Initialization.

Figure 1 Interior tiles have four layers. Border tiles have one layer and are labeled with the $\square$ symbol. (a) An illegal computation square for Layer 2. The constraint applies to the four tile types for Layer 2 shown in gray. (b) An illegal pair for Layer 2. The constraint applies to the two adjacent tile types for Layer 2 shown in gray. (c) An illegal translation square from Layer 2 to Layer 3. The constraint applies to two border tiles and the tile types for Layers 2 and 3 for the other two interior tiles. (d) An illegal initialization square for Layer 2. The constraint applies to two border tiles and the Layer 2 tile types for the other two interior tiles.

4 Overview of Proofs

Thoeorem 4: Gapped Weighted Tiling. Recall that the standard encoding of a TM into tiling rules is very brittle in that a single fault can derail the entire computation. The most straight forward way to overcome this is using a construction which embeds many repetitions of the computation, so that many faults would be required to derail a large number of those computations. Multiple computations thus need to be set up and initiated, using a single faulty TM with TI rules. In our construction, this is achieved by a first stage of the computation (implemented in Layer 1, as we describe below), which, roughly, creates intervals in the top row of Layer 1, such that the independent repetitions of the computations will occur in different strips on the grid; the boundaries of the strips are determined by those intervals. The difficulty is how to implement the initial set up using a single TM, in a fault tolerant way. We now describe the details.

The tiling rules for the first two layers, as well as the reduction mapping x to N are independent of the language $L \in$ NEXP, the language we are reducing from. Let V denote the exponential time verifier for L. In general tiles will be either *tape* tiles which encode a single symbol from the TM's tape alphabet or *head* tiles which encode both the state of the TM as well as the current tape symbol to which the head is pointing.

The TM computation represented in Layer 1 starts with two non-blank symbols and proceeds to write a sequence of intervals on the tape, where an interval is a sequence of B symbols bracketed on either side by a *delimiter* tile from the set $\{X, \overline{X}, \triangleleft, \triangleright\}$. The TM just repeatedly executes a single loop which we refer to as the *Outer Loop*. In one iteration of the Outer Loop, an additional B symbol is inserted into every interval and a new interval with no B's in the middle is added to the right end of the non-blank symbols. In a fault-free execution of the TM, after m iterations of the loop, there are $m+1$ intervals. The number of symbols in each interval (including the delimiter tiles on either end) is $m+2, m+1, m, \ldots, 2$. For $m = 4$, the row should look like:

□	$(q/\triangleleft)$	B	B	B	X	B	B	X	B	X	$\triangleright$	#	#	$\cdots$	#	□

When the top row of Layer 1 is translated to Layer 2, the head tile for the Layer 1 TM is translated to a tape tile (so the state information is lost) and a head tile is inserted on the left end of every interval. For example, an interval $X\ B\ B\ B\ \cdots\ B\ X$ at the end of Layer 1 is translated to $X\ (q_s/S)\ B\ B\ \cdots\ B\ T\ X$ in the first row of Layer 2. In Layers 2 and 3 the sizes and locations of the intervals do not change within a row unless the interval contains an illegal square. Thus, a single interval over all the rows of Layer 2 forms a vertical strip of tiles, and a separate, independent computation takes place within each strip. See Figure 2 for an example. Once the intervals are created on Layer 1, each computation on Layers 2 and 3 is fault-free unless the strip contains an illegal square. Thus, the number of illegal squares is at least the number of strips that fail to complete their computation correctly.

In Layer 2, the computation is just a binary counter TM that continually increments a binary counter. All the strips that do not contain an illegal square will have the same string x represented in the final row of Layer 2. The string x then serves as the input to the computation in Layer 3. The binary counter TM in Layer 2 runs for exactly $N-3$ steps. The reduction is the function that maps x to N, where the string x is written on the tape of a binary counting TM after $N-3$ steps. The full version [4] gives an exact formula mapping x to N and shows that the value of the number represented by the string x is $\Theta(N)$, the dimension of the grid. The idea of using a binary counting TM to translate the size of the grid to a binary input for a computation was used previously in [12]. Although since the construction in [12] had a gap of 1, only a single execution of the verifier was needed. Since we are trying to produce a gap of $f(N)$, we need at least $f(N)$ separate computations each of which simulates the verifier on input x.

Each interval $X\ (q_s/S)\ x\ B\ \cdots\ B\ T\ X$ is translated unchanged to Layer 3. The computation in each strip in Layer 3 simulates the verifier on input x using a witness that is guessed in the tiling. There is a final cost for any rejecting computation. If $x \in L$, it will be possible to tile each strip at 0 cost. If $x \notin L$, every strip will contain an illegal square or will incur a cost for the correct rejecting computation. Thus, the gap is essentially created by these parallel computations, each of which contributes a constant cost if $x \notin L$.

Since the sizes of the intervals go down to 0, some of the intervals will be too narrow to complete the computation in either Layers 2 or 3. If the head ever hits the right end of its interval, it transitions to an infinite loop, causing no additional cost. A standard padding argument (provided in detail in the full version [4]) guarantees that an interval need only be $\Theta(N^{1/4})$ wide to complete the computations in Layers 2 and 3. The analysis of Layer 1 then needs to guarantee that despite the faults, there will be sufficiently many sufficiently wide intervals.

The main challenge in the proof is in making the computation in Layer 1 fault-tolerant, meaning that each illegal pair or square cannot derail the computation too much. The horizontal rules in Layer 1 are critical for enforcing that this cannot happen. We show that a row in the tiling that has no illegal pairs corresponds to a sensible configuration of the TM.

In particular such a row has exactly one head tile that lies in between the $\triangleleft$ and $\triangleright$ tiles. Note that faults can still alter the computation in potentially strange ways. Nonetheless, we also show that starting from a row with no illegal pairs, the Layer 1 TM will be able to make progress, and after a sequence of fault-free steps (corresponding to a sequence of rows containing no illegal squares), the computation will perform a complete iteration of the loop. Since the number of illegal pairs and squares is bounded by $O(N^{1/4})$, there are enough complete iterations of the loop to ensure that the last row of Layer 1 has enough intervals that are wide enough to complete the computations in Layers $2, 3$.

By far the most technically involved part of the paper is the analysis of Layer 1. All of the results make use of a tight characterization of the difference between the final row in Layer 1 of a fault-free tiling and the final row of a tiling with faults. In fact, the result on Gapped Weighted Tiling could be established with looser bounds, but we provide the analysis once in a form that can be used for all the results in the paper. Section 4 describes more fully how this tight characterization is accomplished.

Theorem 2: Weighted Tiling Function. The hardness reduction for Function Weighted Tiling reduces from an oracle class. The function f is computed by a polynomial time TM M with access to an oracle for language $L' \in$ NEXP. Let V denote the exponential time verifier for L'. Using a standard padding argument (see for example Lemma 2.30 from [3]) we can assume that for a constant c of our choice, for every $|x| = n$, there is a $\overline{n} \leq cn$, such that the size of $f(x)$ is at most $\overline{n}$, and M makes at most $\overline{n}$ oracle calls to L'. Let z denote an $\overline{n}$-bit string denoting the responses to the oracle queries made on input x. With x and z fixed, the set of inputs to the oracle $(o_1, \ldots, o_{\overline{n}})$ is also determined. $V(o_j)$ is an indicator function denoting whether o_j is in L'. Note that since L' is in NEXP, if $V(o_j) = 1$, there exists a witness that will cause the verifier to accept and if $V(o_j) = 0$, V will always reject regardless of the witness. Define:

$$\mathcal{C}(x, z) = \sum_{j=1}^{\overline{n}} \left[(1 - z_j) \cdot 2^{\overline{n}-j} + z_j \cdot (1 - V(o_j)) \cdot 2^{\overline{n}}\right] \tag{1}$$

Let $f(x, z)$ be the output of TM M on input x with oracle responses z. Note that since $|f(x, z)| \leq \overline{n}$, $f(x, z) \leq 2^{\overline{n}}$. The construction will ensure that the minimum cost tiling for a particular x and z will be $2^{\overline{n}+5}\mathcal{C}(x, z) + 2^3 \cdot f(x, z)$. Note that $\mathcal{C}(x, z)$ represented in the $\overline{n}$ high-order bits of the cost has the necessary structure where the costs for a *no* oracle response decrease exponentially in j, the index of the oracle query. The cost for a *yes* guess will be 0 if the input to the oracle o_j is in fact in L' (i.e., $V(o_j) = 1$) and will be a very large cost of $2^{2\overline{n}+5}$ if o_j is not in L'. This function will guarantee that the overall cost is minimized when z is the correct string of oracle responses. In addition, the low order bits encode the output of the function $f(x, z)$. So if the minimum cost tiling can be computed, this will correspond to $f(x)$, which is $f(x, \bar{z})$, where $\bar{z}$ is the string of correct oracle responses. The factor of 8 ensures that even if the minimum cost is off by ± 3, the value of $f(x)$ can still be recovered.

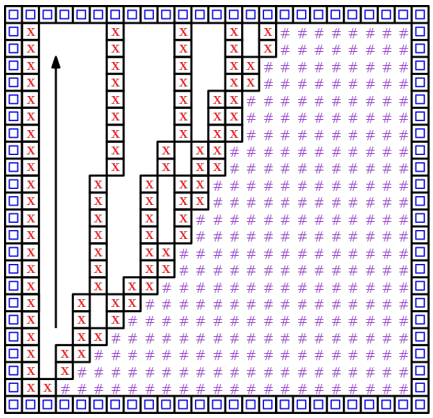
So far what we have described just implementing the original accounting scheme devised by Krentel. The challenge is to implement this cost function in 2D with TI terms. Note that since the tiling rules are fixed parameters of the problem, it is not possible to encode the cost function directly into the penalty terms. As with the Gapped Weighted Tiling problem the function is collectively computed by a set of parallel processes within each strip created by the intervals from Layer 1. However, instead of a threshold function which is either $+f(N)$ or 0, the parallel processes must collectively compute the more intricate function described above, which requires that the individual processes have some additional information.

We will describe first what happens in a fault-free computation (with no illegal pairs or computation squares) and then describe how fault-tolerance is enforced and proven. A schematic view of the construction ins given in Figure 2. The construction for Layers 1 and 2 are exactly the same as for the Gapped Weighted Tiling problem. Layer 1 creates a set of intervals. We definite the function $\mu(N)$ to denote the number of intervals on the tape if the TM for Layer 1 executes $N-3$ steps. If after $N-3$ steps, the computation just happens to finish at the end of an execution of the Outer Loop then the intervals have sizes (from left to right) $\mu(N)+1, \mu(N), \ldots, 2$. If the computation finishes in the middle of an execution of the Outer Loop, the actual sequence of interval sizes will be close to $\mu(N)+1, \mu(N), \ldots, 2$. The largest interval could have size $\mu(N)+2$ and there may be a couple missing values in the range where the current interval is being increased. A complete description of the possible deviations is given in the full version [4]. $\mu(N)$ is $\Theta(N^{1/4})$ and we show using a standard padding argument that for the constant c of our choice, all of the computations require at most $c\mu(N)$ space. This allows us to establish that at least half of the intervals will be large enough to complete the required computations.

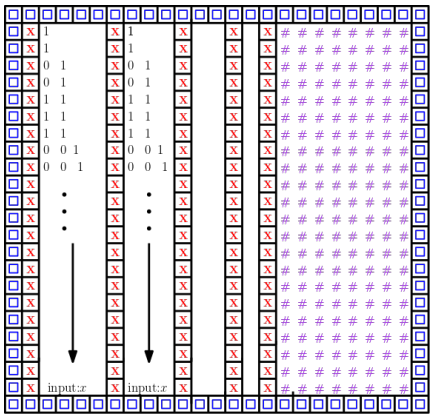
As in the previous construction, Layer 2 then executes a binary counting TM which results in the string x written to the left of each interval which is large enough to complete the computation. Note that Layer 1 is a *global* TM which executes a single process across the entire grid, while Layer 2 represents *local* computations within each strip. When x is translated from Layer 2 to Layer 3 it is augmented with a guess string z for the oracle queries. x. However, there is no guarantee that the guess for each interval is the same. Note that z can be arbitrary but it must be consistently the same for each interval. Layer 3 then executes a global TM which imposes a high penalty if the z strings in each strip are not all the same. This penalty is higher than the cost function for any z, so the lowest cost tiling will correspond to a configuration in which each strip has the same x and z.

Finally, in Layer 4, there is a local computation in each interval, each of which makes a $+1$ or 0 contribution towards the overall cost. The computation within each interval requires a unique tag in order to determine which term of the cost it will contribute to. The tag comes from the size of the interval. The computation begins with counting the number of locations in the interval. This can be accomplished by having the head shuttle back and forth between the two ends of the interval implementing both a unary and binary counter until the unary counter extends across the entire interval. The head returns to the left end of the interval and begins the next phase of the computation. Since the size of an interval is at most $O(N^{1/4})$ this phase of the computation will take at most $O(N^{1/2})$ steps.

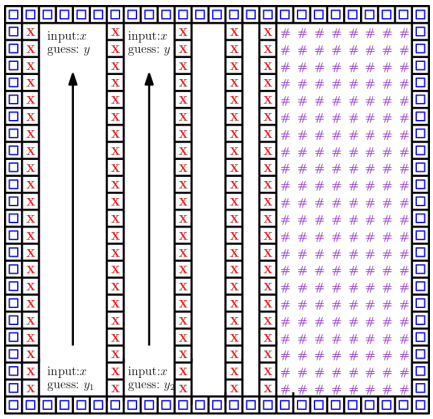
Now each computation has the same pair (x, z) and a its own integer r indicating the size of the interval. From x, the values of N and $\mu(N)$ can be determined. In a fault-free computation, the sizes of the intervals will decrease from left to right. Moreover, all interval sizes are in the set $\{\mu(N)+2, \mu(N)+1, \ldots, 2\}$ with at most one missing value from that set and at most two duplicates. Thus, the value $\mu(N)-r+2$, will be an almost unique identifier for each interval, starting with 0 or 1 on the left and increasing to the right. Using this tag, each interval determines which portion of the cost it will contribute to. The number of intervals assigned to compute a particular term in the cost will depend on the value of the term since each interval can contribute at most 1 to the overall cost. If an interval is assigned to check a *yes* guess ($z_k = 1$) the computation uses x and z to determine the k^{th} input to the oracle o_k, guesses a witness and simulates V on input o_k with the guessed witness. There is a cost of $+1$ if V rejects and 0 if V accepts. If o_k is in fact in L', there is a witness which will allow for a zero cost tiling withing that interval. If $o_k \notin L$, then every witness will lead to a $+1$ cost. Thus, the optimal set of witnesses will result in the minimum



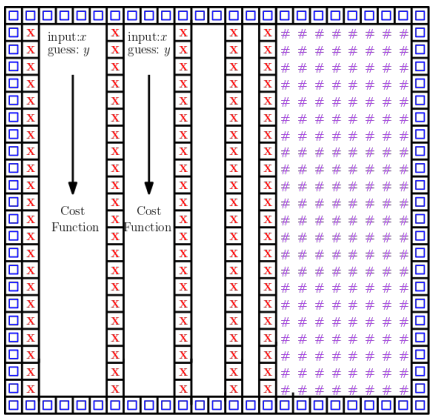
(a) In Layer 1 a single fault-tolerant TM creates the intervals that mark off the width of each strip where independent computations will take place in subsequent layers. The top row of Layer 1 is mapped onto the top row of Layer 2. The computation proceeds upwards from bottom to the top.



(b) In Layer 2, an independent computation takes place in each strip. The computation proceeds from top to bottom. Each strip executes a binary counter TM, so the height of the square is translated into a string x, which serve as the input to the computational problem.



(c) The bottom row of Layer 2 containing the string x is translated to Layer 3. In addition the tiling contains a guess y for the responses to the oracle query. Layer 3 then executes a global TM that proceeds from bottom to top. The computation results in a high cost if the guess strings y in each strip are not all the same.



(d) The independent computations in each strip collectively incur a total cost of $2^{\overline{n}+5}\mathcal{C}(x,z) + 2^3 \cdot f(x,z)$, where $\mathcal{C}(x,z)$ is denoted in Equation (1).

Figure 2 Schematic image showing the four layers in the construcion.

value for $2^{\overline{n}+5}\mathcal{C}(x,z)$. In addition, exactly $2^3 f(x,z)$ of the intervals will just transition to the rejecting state, incurring a cost of $+1$. The total cost due to those intervals is $2^3 f(x,z)$. For the remaining intervals, no cost is incurred.

The cost of a computation fault (illegal pair or square) is a constant that is larger than the cost of ending in a rejecting computation. Therefore, for each independent computation (in Layers 2 and 4) the optimal tiling will correspond to a correct computation which may or may not incur a cost for ending in a rejecting state. Technically, the most challenging part of the proof is to show that the process on Layer 1 which creates the intervals is fault-tolerant. The proof for Function Weighted Tiling requires stronger conditions than for Gapped Weighted

Tiling since we not only have to show that there are a large number of large intervals at the end of Layer 1 but we need to establish that the sequence of interval sizes is close to what one would have in a fault-free computation. To this end, we use a potential function A which captures how much a sequence of interval sizes $(s_1, s_2, \ldots, s_m)$ deviates from the expected sequence $(m+1, m, m-1, \ldots, 2)$. The main part of the proof is to show that each illegal square or pair can cause the value of A to increase by at most a constant amount. At the end of Layer 1, the ideal sequence of interval sizes is $(\mu(N)+1, \mu(N), \ldots, 2)$. Every interval size that is missing from the actual sequence of interval sizes has caused A to increase by at least a fixed amount which in turn corresponds to faults incurred in the computation. Thus, we show that it is more cost-effective to complete the computation correctly (and not incur the higher cost of a fault) and incur the smaller potential cost of a rejecting computation.

The most important measure of progress of the tiling/computation in Layer 1 is the number of times the encoded TM completes an iteration of the Outer Loop in which the size of every interval increases by 1 and a new interval of size 2 is added. Faults can potentially cause an iteration of the Outer Loop to take longer as they may force the head to shuttle back and forth more times which in turn could result in fewer iterations. Even in a fault-free computation, the number of steps per iteration increases with each iteration because there are more intervals. The analysis in the full version provides a lower bound on the number of times the loop is completed in relation to the number of completed loops in a fault-free computation. The proof is a delicate inductive argument which uses the fact that the increase in the running time of a loop is not accelerated too much with each additional fault.

Proof Overview for Parity Weighted Tiling. The proof for parity weighted tiling is very similar to the function problem. Suppose that a language $L \in \mathrm{P}^{\mathrm{NEXP}}$ is computed by a TM M with access to an oracle for $L' \in \mathrm{NEXP}$. Let $M(x, z)$ be the indicator function that is 0 if $M(x, z)$ accepts and 1 if $M(x, z)$ rejects. The overall cost computed by the collective computations is: $4\mathcal{C}(x, z) + M(x, z)$. The left-most interval computes $M(x, z)$ and results in a $+1$ cost in the case that M rejects. The remaining intervals which collectively compute Krentel's cost function all impose costs of $+2$ or 0. Thus the expression $4\mathcal{C}(x, z) + M(x, z)$ will guarantee that the minimum $\mathcal{C}(c, z)$ corresponds to the correct guess $\bar{z}$. Furthermore, the rightmost bit will be $M(x, z)$ which will cause the minimum cost to be odd or even, depending on whether M accepts.

5 Discussion, Related Work, and Open Problems

Despite the fact that the function version of classical local-Hamiltonians describes the task of the computational (classical) physicist much more naturally than decision problems, complexity of function problems was hardly studied even in the non-TI setting, in the literature of classical theory of computer science.

Recently, related results were discovered in the domain of quantum computational complexity. In particular, in [3], Aharonov and Irani use a construction for the function version of (finite) quantum local Hamiltonian as a component for a hardness result for the infinite 2D grid. More specifically, they prove that the problem of estimating the ground energy of a local Hamiltonian on a finite 2D grid, is hard for $\mathrm{FP}^{\mathrm{NP}}$. Importantly, their results do not imply the hardness result presented in this paper, and it seems impossible to extend their proof to deduce the classical hardness result of Theorem 2. Like [3] we implement Krentel's cost function using a fixed Hamiltonian term, but since their construction is quantum (as opposed to the classical construction in this paper), they are able to prove the

result using a completely different set of tools which do not carry over to the classical case. In quantum constructions, the lowest energy is an eigenvalue of a general Hermitian matrix and the matrix can be constructed to fine tune the ground energy to an inverse polynomial precision. In classical constructions, the total energy will be a sum over terms where each term is chosen from a constant-sized set of values determined by the finite horizontal and vertical tiling rules. This allows far less control in the classical setting over the precision of the minimum cost tiling.

Incidentally, note that the results for the quantum case proven in [3] are not tight, which follows from the fact that they use a quantum construction to obtain hardness for $\mathrm{FP}^{\mathrm{NEXP}}$, a classical complexity class. It seems challenging to make the characterization tight in the quantum case. In contrast to the class NP, the class QMA is a class of promise-problems and in simulating a $\mathrm{P}^{\mathrm{QMA}}$ machine, there is no guarantee that the queries sent to the QMA oracle will be valid queries. The cost/energy applied for a particular query will depend on the probability that a QMA verifier accepts on the provided input. If the input is invalid, then the probability of acceptance can be arbitrary. Thus, Krentel's cost function will potentially be an uncontrolled quantity. Typically in a reduction where we want to embed the output of a function into the value of the minimum energy, the low order bits of the energy are used to encode the output of the function. It's not clear how to do this without being able to control the binary representation of the minimum energy. Note that by embedding a classical computation in the Hamiltonian, the issue of invalid queries is circumvented.

Both [3] and [21] study the complexity of computing the ground energy density of infinite TI Hamiltonians to within a desired precision making use of the technique introduced by Cubitt, Prerez-Garcia, and Wolf which embeds *finite* Hamiltonian constructions of exponentially increasing sizes, into the 2D infinite lattice, using Robinson tiles. Robinson tiling rules [18] force an aperiodic structure on the tiling of the infinite plane, with squares of exponentially increasing size. The quantum construction of [3] layers a TI 1D Hamiltonian on top of one of the sides of all the squares. The classical construction of [21] layers a classical finite construction on each square. Neither work obtains tight results due to the same issue with invalid queries, although the two papers compromise in completely different ways. The primary technical innovation introduced in [21] is to devise a more robust version of Robinson tiles which ensures that the lowest energy state corresponds to a correct Robinson tiling, even though the cost of the classical finite construction layered on top may introduce a penalty. If it were possible to obtain an even more robust version of Robinson tiles, one potentially could layer the finite construction from the current paper on top the more robust constructions in the hopes of showing that computing the ground energy density of a classical TI Hamiltonian in the thermodynamic limit is complete for $\mathrm{EXP}^{\mathrm{NEXP}}$ under Karp reductions.

The results in this paper are also related to the work of Ambainis [5] which characterizes the complexity of measuring local observables of ground states of local Hamiltonians (APX-SIM), showing that the problem is complete for $P^{\mathrm{QMA}[\log n]}$. $P^{\mathrm{QMA}[\log n]}$ contains those problems that can be solved by a polynomial time classical TM with access to $O(\log n)$ queries to a QMA oracle. This type of question (determining a property of the ground state) is similar to our classical result about determining whether the cost of the optimal tiling is odd or even. The results on APX-SIM [5, 11, 10] are not hindered by the issue of invalid queries because the quantity being measured is not the actual energy itself. Note that the important point here is the property that distinguishes the state to be measured (minimum energy) is different than the local observable applied to the measured state. By contrast, computing the energy of the lowest energy state appears to be more difficult. The issue of invalid queries appears to be an obstacle, even when the Hamiltonian terms are position-dependent as in the constructions of [11, 10], as well as in the TI constructions in [3, 20].

Finally, it was mentioned earlier that the approximation problem considered here differs from the standard PCP setting in that the underlying graph is a grid and the terms are TI. It remains an open question as to whether there is a family of TI instances of constraint satisfaction on general graphs for which it is hard to estimate the optimal solution to within an additive $\Theta(N)$.

References

1 Ian Affleck, Tom Kennedy, Elliott H. Lieb, and Hal Tasaki. Rigorous results on valence-bond ground states in antiferromagnets. *Phys. Rev. Lett.*, 59:799–802, August 1987. doi:10.1103/PhysRevLett.59.799.

2 Dorit Aharonov, Daniel Gottesman, Sandy Irani, and Julia Kempe. The power of quantum systems on a line. *Communications in Mathematical Physics*, 287(1):41–65, January 2009. doi:10.1007/s00220-008-0710-3.

3 Dorit Aharonov and Sandy Irani. Hamiltonian complexity in the thermodynamic limit. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 750–763. ACM, 2022. doi:10.1145/3519935.3520067.

4 Dorit Aharonov and Sandy Irani. Translationally invariant constraint optimization problems, 2022. arXiv:2209.08731.

5 Andris Ambainis. On physical problems that are slightly more difficult than qma. In *2014 IEEE 29th Conference on Computational Complexity (CCC)*, pages 32–43, 2014. doi:10.1109/CCC.2014.12.

6 F Baharona. On the computational complexity of ising spin glass models. *Journal of Physics A: Mathematical and General*, 15(10):3241–3253, 1982.

7 Johannes Bausch, Toby Cubitt, and Maris Ozols. The complexity of translationally invariant spin chains with low local dimension. *Annales Henri Poincaré*, 18(11):3449–3513, October 2017. doi:10.1007/s00023-017-0609-7.

8 Toby S. Cubitt, David Perez-Garcia, and Michael M. Wolf. Undecidability of the spectral gap. *Nature*, 528(7581):207–211, December 2015. doi:10.1038/nature16059.

9 Sevag Gharibian, Yichen Huang, Zeph Landau, and Seung Woo Shin. Quantum hamiltonian complexity. *Foundations and Trends® in Theoretical Computer Science*, 10(3):159–282, 2015. doi:10.1561/0400000066.

10 Sevag Gharibian, Stephen Piddock, and Justin Yirka. Oracle complexity classes and local measurements on physical hamiltonians. *arXiv*, 2019. arXiv:1909.05981.

11 Sevag Gharibian and Justin Yirka. The complexity of simulating local measurements on quantum systems. *Quantum*, 3:189, September 2019. doi:10.22331/q-2019-09-30-189.

12 Daniel Gottesman and Sandy Irani. The quantum and classical complexity of translationally invariant tiling and hamiltonian problems. *Theory of Computing*, 9(2):31–116, 2013. doi:10.4086/toc.2013.v009a002.

13 Sorin Istrail. Statistical mechanics, three-dimensionality and np-completeness. i. universality of intractability for the partition function of the ising model across non-planar lattices. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA*, pages 87–96, January 2000. doi:10.1145/335305.335316.

14 A. Yu. Kitaev, A. H. Shen, and M. N. Vyalyi. *Classical and Quantum Computation*. American Mathematical Society, USA, 2002.

15 Mark W. Krentel. The complexity of optimization problems. In Alan L. Selman, editor, *Structure in Complexity Theory*, pages 218–218, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

16 R. Oliveira and B. Terhal. The complexity of quantum spin systems on a two-dimensional square lattice. *arXiv*, 2005. arXiv:quant-ph/0504050.

17 Christos H. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.

18 Raphael Robinson. Undecidability and nonperiodicity for the tilings of the plane. *Invent. Math.*, 12:177–209, 1971.

19 Hao Wang. Proving theorems by pattern recognition. *Communications of the ACM*, 3(4):220–234, 1960.

20 James D. Watson, Johannes Bausch, and Sevag Gharibian. The complexity of translationally invariant problems beyond ground state energies. *arXiv*, 2020. `arXiv:2012.12717`.

21 James D. Watson and Toby S. Cubitt. Computational complexity of the ground state energy density problem. *arXiv*, 2021. `arXiv:2107.05060`.

An Exponential Separation Between Quantum Query Complexity and the Polynomial Degree

Andris Ambainis
Faculty of Computing, University of Latvia, Riga, Latvia

Aleksandrs Belovs
Faculty of Computing, University of Latvia, Riga, Latvia

Abstract

While it is known that there is at most a polynomial separation between quantum query complexity and the polynomial degree for *total* functions, the precise relationship between the two is not clear for *partial* functions.

In this paper, we demonstrate an exponential separation between exact polynomial degree and approximate quantum query complexity for a partial Boolean function. For an unbounded alphabet size, we have a constant versus polynomial separation.

2012 ACM Subject Classification Theory of computation → Quantum query complexity

Keywords and phrases Polynomials, Quantum Adversary Bound, Separations in Query Complexity

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.24

Funding This work has been supported by the ERDF project number 1.1.1.5/18/A/020 "Quantum algorithms: from complexity theory to experiment."

Acknowledgements We thank Scott Aaronson for writing the open problem survey [2] which attracted our attention to this problem. We also thank the anonymous reviewers at the CCC conference for their numerous valuable suggestions on the presentation of the paper.

1 Introduction

A polynomial method is an established tool for proving lower bounds for classical [19, 20] and quantum [9] query complexity. In the quantum case, this method is based on an observation that a quantum query algorithm can be turned into an approximating polynomial whose degree is at most twice the query complexity of the algorithm. Showing that a function cannot be approximated by a low-degree polynomial implies that it cannot be solved query-efficiently on a quantum computer. This method was used early on to establish important results like the precise characterisation of quantum query complexity of total symmetric Boolean functions [9] and the optimal lower bound for the collision problem [5]. It was also used recently to prove strong lower bounds on k-distinctness and image size testing [14, 17].

The question of how good this lower bound technique is has gathered attention. Nisan and Szegedy [20] proved that $Q(f) = O\big(\deg(f)^8\big)$ for total Boolean f, where $\deg(f)$ is the exact degree, and $Q(f)$ is the quantum query complexity.[1] This was subsequently improved to $Q(f) = O\big(\deg(f)^4\big)$ (attributed to Nisan and Smolensky in [13]), and $Q(f) = O\big(\deg(f)^3\big)$ by Midrijānis [18].[2] Concerning the approximate degree $\widetilde{\deg}(f)$, Beals, Buhrman, Cleve,

[1] Actually, it was shown that $D(f) = O\big(\deg(f)^8\big)$, where $D(f)$ is the deterministic decision tree complexity. Here we use that $D(f)$ upper bounds approximate (and, actually, even exact) quantum query complexity. Similar comments apply to other upper bounds below.

[2] Stated as $D(f) = O\big(\deg(f)^4\big)$ and $D(f) = O\big(\deg(f)^3\big)$, respectively.

Mosca, and de Wolf [9] showed that $Q(f) = O\big(\widetilde{\deg}(f)^6\big)$ for total Boolean functions.[3] This was improved by Aaronson, Ben-David, Kothari, Rao, and Tal [4] to $Q(f) = O\big(\widetilde{\deg}(f)^4\big)$ for all total Boolean functions.[4]

On the other hand, Ambainis [7] constructed a total Boolean function with superlinear but subquadratic separation between the exact degree $\deg(f)$ and quantum query complexity $Q(f)$. This also implies a similar separation between $\widetilde{\deg}(f)$ and $Q(f)$ as $\widetilde{\deg}(f) \leq \deg(f)$. Aaronson, Ben-David, and Kothari [3] demonstrated an almost quartic separation between $Q(f)$ and $\widetilde{\deg}(f)$ as well as an almost quadratic separation between $Q(f)$ and $\deg(f)$ for a total Boolean function. The former separation is optimal due to the aforementioned result by Aaronson et al. [4].

In order to prove a separation between quantum query complexity and the polynomial degree, one has to use a different tool than the polynomial method to prove lower bounds on quantum query complexity. A popular alternative is the adversary method. Indeed, Ambainis [7] used his, recent at the time, adversary method [6]. Aaronson et al. [3] used their cheat sheet technique, but also relied on the lower bound for the k-sum problem [12], which used the negative-weight adversary [15], as well as other tools.

Note that all these results consider *total* Boolean functions. Up to our knowledge, the question of obtaining a separation between quantum query complexity and the polynomial degree for *partial* functions has not been studied. This is interesting, as partial functions usually allow for much larger separations. This question was raised as an open problem in a recent survey by Aaronson [2].

And indeed, $Q(f)$ versus $\deg(f)$ is not an exception, as we will prove the following two results in our paper (see Section 2.1 for the precise definition of the polynomial degree):

▶ **Theorem 1.** *For every $q \geq n^{12}$, there exists a partial function $f\colon D \to \{0,1\}$ with $D \subseteq [q]^{3n}$ with the following properties:*
- *its exact polynomial degree is at most 9;*
- *its quantum query complexity is $\Omega(n^{1/3})$.*

Take q as the smallest power of 2 exceeding n^{12}. By replacing each variable of the function with $\log q$ Boolean variables, we get the following corollary.

▶ **Corollary 2.** *There exists a family of partial Boolean functions $f\colon D \to \{0,1\}$ with $D \subseteq \{0,1\}^n$ satisfying the following two properties:*
- *its exact polynomial degree is $O(\log n)$;*
- *its quantum query complexity is $\widetilde{\Omega}(n^{1/3})$.*

In Section 3, we formulate the problem and prove the upper bound on its polynomial degree. In Section 4, we prove the lower bound on its quantum query complexity. We also use the adversary method in our proof of the lower bound. The corresponding function is closely related to a function studied previously by Belovs and Rosmanis [11].

Finally, let us note that we have to use the negative-weight formulation of the adversary bound in our separation, and not the easier-to-apply positive-weight, which was used, for instance, by Ambainis in his aforementioned separation [7]. This is because of the result by Anshu, Ben-David, and Kundu [8] stating at most quadratic separation between positive-weight quantum adversary and the polynomial degree even for partial functions.

---

[3] Again, stated as $D(f) = O\big(\widetilde{\deg}(f)^6\big)$. It is also often cited as $D(f) = O\big(Q(f)^6\big)$

[4] Although the result is stated as $D(f) = O\big(Q(f)^4\big)$, what is actually proven in the paper is $D(f) = O\big(\widetilde{\deg}(f)^4\big)$. See also the corresponding cell in Table 1 of the paper.

2 Preliminaries

For a positive integer m, let $[m]$ denote the set $\{1, 2, ..., m\}$. Notation $\mathbb{Z}_m$ denotes the additive group modulo m. For a predicate P, we write 1_P to denote the indicator variable that is 1 is P is true, and 0 otherwise.

For an $X \times Y$-matrix A, $x \in X$, and $y \in Y$, we denote by $A[\![x, y]\!]$ its (x, y)-th entry. For $X' \subseteq X$ and $Y' \subseteq Y$, $A[\![X', Y']\!]$ denotes the corresponding submatrix. Similar notation is also used for vectors. Next, $\|\cdot\|$ denotes the spectral norm (the largest singular value), and $\circ$ denotes the Hadamard (i.e., entry-wise) product of matrices.

We say that a linear operator $A\colon L \to K$ is an *isometry from L' into K* if its coimage is $L' \subseteq L$ and $\|Av\| = \|v\|$ for all $v \in L'$. In other words, all the singular values of A are 1, and the span of its right singular vectors is L'.

2.1 Polynomials

For a (partial) Boolean function $f\colon D \to \{0,1\}$ with $D \subseteq \{0,1\}^n$, a *representing polynomial* is defined as a real multivariate polynomial P in variables $x_1, \ldots, x_n$, treated as elements of $\mathbb{R}$, such that

- $P(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ for every $x \in D$;
- $0 \le P(x_1, \ldots, x_n) \le 1$ for all $x \in \{0,1\}^n$.

The motivation behind this definition is that, as shown in [9], every quantum query algorithm evaluating f in T queries exactly can be turned into a representing polynomial for f of degree at most $2T$.

The notion of representing polynomial can be generalised for functions with larger input alphabets as well. Let $f\colon D \to \{0,1\}$ with $D \subseteq [q]^n$ be a (partial) function with alphabet size q. Then, we can define its representing polynomial (see, e.g., [1]) as a polynomial in nq Boolean variables $1_{x_i=a}$, where i ranges over $[n]$ and a over $[q]$. Namely, it is a real multivariate polynomial P satisfying the following properties:

- $P(1_{x_1=1}, \ldots, 1_{x_n=q}) = f(x_1, \ldots, x_n)$ for every $x \in D$;
- $0 \le P(1_{x_1=1}, \ldots, 1_{x_n=q}) \le 1$ for all $x \in [q]^n$.

The motivation is similar to the Boolean case.

It may seem that the two definitions do not match for $q = 2$, but in this case we have the identity $1_{x_i=2} = 1 - 1_{x_i=1}$, which allows us to remove the variables $1_{x_i=2}$, giving essentially the same definition.

The *exact polynomial degree* of a function f is the minimal degree of its representing polynomial. Similarly, one can define an approximating polynomial and the approximate degree, but we will not need these notions in the paper.

Assume for simplicity that $q = 2^\ell$ is a power of two. In this case, we can convert a function f with domain in $[q]^n$ into a function $\widetilde{f}$ with domain in $\{0,1\}^{n\ell}$ by replacing each $a \in [q]$ by a bit-string $(a_1, \ldots, a_\ell) \in \{0,1\}^\ell$ and each variable $x_i \in [q]$ with ℓ Boolean variables $x_{i,1}, \ldots, x_{i,\ell}$. We have

$$1_{x_i=a} = 1_{x_{i,1}=a_1} 1_{x_{i,2}=a_2} \cdots 1_{x_{i,\ell}=a_\ell}.$$

Therefore, every representing polynomial for f of degree d can be turned into a representing polynomial for the function $\widetilde{f}$ of degree $d\ell$.

2.2 Adversary Bound

In the paper, we only use the (negative-weight) adversary bound for decision problems, which is defined as follows.

Let $f\colon D \to \{0,1\}$ with $D \subseteq [q]^n$. An *adversary matrix* for f is a real $f^{-1}(1) \times f^{-1}(0)$-matrix Γ. For any $j \in [n]$, the $f^{-1}(1) \times f^{-1}(0)$-matrix Δ_j is defined by

$$\Delta_j[\![x,y]\!] = \begin{cases} 0, & \text{if } x_j = y_j; \\ 1, & \text{if } x_j \neq y_j. \end{cases} \tag{1}$$

▶ **Theorem 3** (Adversary bound [15, 16]). *In the above notation, the quantum query complexity of the function f is $\Theta\big(\mathrm{ADV}^{\pm}(f)\big)$, where $\mathrm{ADV}^{\pm}(f)$ is the optimal value of the semi-definite program*

$$\text{maximise} \quad \|\Gamma\| \tag{2a}$$

$$\text{subject to} \quad \|\Delta_j \circ \Gamma\| \leq 1 \qquad \textit{for all } j \in [n]. \tag{2b}$$

Here maximisation is over all adversary matrices Γ for f.

We can choose any adversary matrix Γ and scale it down so that the condition $\|\Delta_j \circ \Gamma\| \leq 1$ holds. Thus, we can use the condition $\|\Delta_j \circ \Gamma\| = \mathrm{O}(1)$ instead of $\|\Delta_j \circ \Gamma\| \leq 1$.

Working with the matrix $\Delta_j \circ \Gamma$ might be cumbersome, so the following trick can be applied. We write $\Gamma \overset{\Delta_j}{\longmapsto} \Gamma_j$ if $\Gamma \circ \Delta_j = \Gamma_j \circ \Delta_j$. In other words, we modify the entries $\Gamma[\![x,y]\!]$ with $x_j = y_j$ to obtain Γ_j. As shown in [16], $\|\Delta_j \circ \Gamma\| \leq 2\|\Gamma_j\|$, hence we can replace $\Delta_j \circ \Gamma$ with Γ_j in (2b).

3 The Problem and the Polynomial Upper Bound

The function for which we give the separation is defined in the following way.

Assume we have $3n$ input variables $x_1, x_2, \ldots, x_{3n} \in [q]$. We treat them as the elements of $\mathbb{Z}_q$. Divide the set of indices into three subsets: $A = \{1, \ldots, n\}$, $B = \{n+1, \ldots, 2n\}$ and $C = \{2n+1, \ldots, 3n\}$. Consider the following system of n^3 linear equations modulo q:

$$x_a + x_b + x_c = r_{a,b,c}, \qquad \text{for } a \in A,\ b \in B \text{ and } c \in C, \tag{3}$$

where $r_{a,b,c} \in \mathbb{Z}_q$ are some fixed values. We call the individual equations in (3) *tripartite equations*, and the whole system of n^3 equations the *tripartite system*.

▶ **Definition 4.** In the *threshold satisfiability problem*, given $x_1, x_2, \ldots, x_{3n}$, the task is to distinguish the following two cases:

- there is no equation satisfied in the tripartite system (3) (negative case); and
- there are exactly n equations satisfied in the tripartite system (3) (positive case).

Note that threshold satisfiability problem depends on parameters $r_{a,b,c}$, which are *not* parts of the input, but specify a particular instance of the problem.

Although the tripartite system (3) has n^3 equations, the largest number of simultaneously satisfiable equations usually is much smaller.

▶ **Proposition 5.** *Assume $q \geq n^{12}$. Then, there exists a choice of $r_{a,b,c} \in \mathbb{Z}_q$ such that, for every input x, less than $4n$ of the equations in (3) are satisfied.*

Proof. This is a simple application of the probabilistic method. In the following, all the probabilities are with respect to the uniform distribution over $r_{a,b,c}$.

Let S be a subset of the equations in (3) of size $4n$. As the number of variables is $3n$, we get that

$$\Pr\big[\text{All the equations in } S \text{ can be satisfied}\big] \leq q^{3n-|S|} = q^{-n}.$$

Therefore, by the union bound, the probability that it possible to satisfy at least $4n$ equations in (3) is at most

$$\binom{n^3}{4n} q^{-n} < n^{12n} q^{-n} \leq 1$$

by our choice of q. ◀

We will call such a choice of the right-hand sides $r_{a,b,c}$ *good*.

▶ **Theorem 6.** *If $r_{a,b,c}$ are good, the exact polynomial degree of the threshold satisfiability function is at most 9.*

Proof. Consider the following function

$$K(x) = \sum_{a\in A,\ b\in B,\ c\in C} 1_{x_a+x_b+x_c=r_{a,b,c}},$$

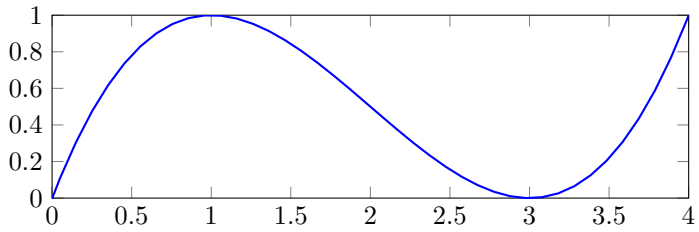
which counts the number of satisfied equations in (3). We have that

$$1_{x_a+x_b+x_c=r} = \sum_{s,t\in\mathbb{Z}_q} 1_{x_a=s} 1_{x_b=t} 1_{x_c=r-s-t},$$

hence, the degree of K is 3. Take the univariate cubic polynomial

$$T(z) = \frac{1}{4}z^3 - \frac{3}{2}z^2 + \frac{9}{4}z.$$

The following is a plot of this polynomial. It has the following properties: $T(0) = 0$, $T(1) = 1$ and $0 \leq T(z) \leq 1$ for all $0 \leq z \leq 4$.



The polynomial $T\left(\frac{K(x)}{n}\right)$ satisfies all the requirements. ◀

4 Quantum Lower Bound

We prove a slightly stronger result, as we prove a lower bound for an easier function. For $s, t \in [n]$, let

$$\mu_{s,t} = \big\{(j,\ n+1+(j+s \bmod n),\ 2n+1+(j+t \bmod n)) \,\big|\, j \in [n]\big\}. \tag{4}$$

Each $\mu_{s,t}$ is a tripartite matching between $A = \{1,\ldots,n\}$, $B = \{n+1,\ldots,2n\}$ and $C = \{2n+1,\ldots,3n\}$. We call it a shifted tripartite matching. We use

$$M = \{\mu_{s,t} \mid s, t \in [n]\}.$$

to denote the set of all shifted tripartite matchings (for a fixed value of n).

▶ **Definition 7.** In the *tripartite shift problem*, given $x_1, x_2, \ldots, x_{3n}$, the task is to distinguish the following two cases:

- there is no equation satisfied in the tripartite system (3) (negative case); and
- there exists a shifted tripartite matching $\mu \in M$ such that an equation $x_a + x_b + x_c = r_{a,b,c}$ from (3) is satisfied if and only if $(a, b, c) \in \mu$ (positive case).

Since each shifted tripartite matching specifies n tripartite equations, the tripartite shift problem is a restriction of the threshold satisfiability problem. Therefore, any lower bound for the former is a lower bound for the latter.

A closely related problem was studied in [11]. It was like in Definition 7, but with the following two modifications:

- all $r_{a,b,c} = 0$; and
- in the positive case, it is not required that $x_a + x_b + x_c \neq r_{a,b,c}$ for $(a, b, c) \notin \mu$.

Therefore, our result is a strengthening of [11]. We obtain a similar lower bound.

▶ **Theorem 8.** *If $q \geq 4n^3$, the quantum query complexity of the tripartite shift problem is $\Omega(n^{1/3})$ for any choice of $r_{a,b,c}$.*

Essentially the same proof goes through. Since the differences are nonetheless substantial, we reproduce the proof in the remaining part of this section.

4.1 Input-Related Sets

We begin with defining some input-related sets. Let

$$\begin{aligned} \widetilde{N} = [q]^{3n} \quad & \text{be the set of all inputs;} \\ N \quad & \text{be the set of negative inputs; and} \\ P \quad & \text{be the set of positive inputs.} \end{aligned}$$

For $(a, b, c) \in A \times B \times C$, let

$$\widetilde{P}^{a,b,c} = \left\{x \in [q]^{\{a,b,c\}} \mid x_a + x_b + x_c = r_{a,b,c}\right\},$$

be the solution set of the corresponding tripartite equation. For $\mu \in M$, let

$$\widetilde{P}^{\mu} = \prod_{(a,b,c)\in\mu} \widetilde{P}^{a,b,c}. \tag{5}$$

In other words, $x \in [q]^{3n}$ belongs to $\widetilde{P}^{\mu}$ if and only if all the equations of the tripartite system (3) with $(a, b, c) \in \mu$ are satisfied. Some of the remaining equations may be satisfied as well. Finally,

$$\widetilde{P} = \bigsqcup_{\mu\in M} \widetilde{P}^{\mu}.$$

We use the disjoint union here because an input $x \in [q]^{3n}$ can belong to several $\widetilde{P}^{\mu}$ at once. We can define $\widetilde{P}$ more precisely as the set of pairs $\left\{(\mu, x) \mid \mu \in M,\ x \in \widetilde{P}^{\mu}\right\}$. We consider P as a subset of $\widetilde{P}$, which is well-defined since $x \in P$ belongs to exactly one $\widetilde{P}^{\mu}$. The reason for introducing the set $\widetilde{P}$ is the decomposition property (5), which P lacks.

As one can guess from the notation, we use $\widetilde{N}$ and $\widetilde{P}$ as proxies for N and P, respectively. We show that their sizes do not differ too much.

▷ Claim 9. Under the assumption $q \geq 4n^3$, we have $|N| \geq 3|\widetilde{N}|/4$ and $|P| \geq 3|\widetilde{P}|/4$.

Proof. We first prove the claim for N and $\widetilde{N}$. Take $x \in \widetilde{N} = [q]^{3n}$ uniformly at random. There are n^3 equations in (3). The probability x satisfies one fixed equation from this list is $1/q$. By the union bound, the probability x satisfies some equation from the list is $n^3/q \leq 1/4$. This proves $|N| \geq 3|\widetilde{N}|/4$.

We can write $P = \bigsqcup_{\mu \in M} P^\mu$, where $P^\mu = P \cap \widetilde{P}^\mu$ is the set of inputs satisfying precisely the equations (3) with $(a,b,c) \in \mu$. We prove $|P^\mu| \geq 3|\widetilde{P}^\mu|/4$, from which the claim follows. To do so, we can use the same reasoning as above, because the probability a uniformly random $x \in \widetilde{P}^\mu$ satisfies a fixed equation from (3) with $(a,b,c) \notin \mu$ is also $1/q$. ◁

4.2 Overview of the Proof

Now let us describe the general structure of the proof. It follows the proof from [11], and is based on the ideas from [10]. The following collection $\alpha = \alpha(\mu, S)$ of real coefficients will be important:

$$\alpha(\mu, S), \text{ where } \mu \in M \text{ and } S \subseteq [3n] \text{ is such that } \big|S \cap \{a,b,c\}\big| \leq 1 \text{ for all } (a,b,c) \in \mu. \quad (6)$$

If S satisfies the condition in (6), we say that S is *good* for μ. We will implicitly assume that $\alpha(\mu, S) = 0$ if S is not good for μ.

Let us define

$$\|\alpha\| = \max_{S \subseteq [3n]} \sqrt{\sum\nolimits_{\mu \in M} \alpha(\mu, S)^2}. \quad (7)$$

And, for $j \in [3n]$, we define the following operation ∂_j on α:

$$\partial_j \alpha(\mu, S) = \begin{cases} \alpha(\mu, S) - \alpha(\mu, S \cup \{j\}), & \text{if } j \notin S; \\ 0, & \text{if } j \in S. \end{cases}$$

For α as in (6), we will define a $\widetilde{P} \times \widetilde{N}$ matrix $G(\alpha)$. It satisfies the following two properties

$$\|G(\alpha)\| = \|\alpha\| \qquad \text{and} \qquad G(\alpha) \overset{\Delta_j}{\longmapsto} G(\partial_j \alpha) \text{ for all } j \in [3n]. \quad (8)$$

The piece of notation $\|\alpha\|$ in (7) was chosen precisely because of the first equation above.

We will construct an explicit α that satisfies the following conditions:

$$\|\alpha\| = n^{1/3} \qquad \text{and} \qquad \|\partial_j \alpha\| = O(1) \text{ for all } j \in [3n]. \quad (9)$$

We define the adversary matrix Γ as $G(\alpha)[\![P, N]\!]$. On the one hand, $\Gamma \overset{\Delta_j}{\longmapsto} G(\partial_j \alpha)[\![P, N]\!]$, which has norm $O(1)$ by the above. On the other hand, $\|G(\alpha)\| = n^{1/3}$, and using that P and N are close to $\widetilde{P}$ and $\widetilde{N}$, respectively, we get that $\|\Gamma\| = \Omega(n^{1/3})$. Theorem 8 follows then from Theorem 3.

4.3 Fourier Basis

We denote $\mathcal{H} = \mathbb{C}^q$ and, for a set T, use notation $\mathcal{H}^T = \mathbb{C}^{[q]^T} = \mathcal{H}^{\otimes T}$. We often write $\mathcal{H}^{a,b,c}$ instead of $\mathcal{H}^{\{a,b,c\}}$ and similarly for related notions.

Let $\chi_0, \ldots, \chi_{q-1}$ be the Fourier basis of $\mathcal{H}$. Recall that it is an orthonormal basis given by $\chi_i[\![j]\!] = \frac{1}{\sqrt{q}} \omega_q^{ij}$, where $\omega_q = \mathsf{e}^{2\pi \mathsf{i}/q}$. The most important of them is

$$\chi_0 = \frac{1}{\sqrt{q}} \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}.$$

The Fourier basis of $\mathcal{H}^T$ is given by tensor products $\chi_s = \bigotimes_{j\in T} \chi_{s_j}$, where each $s_j \in \{0,\dots,q-1\}$. The *support* of χ_s is $\{j \in T \mid s_j \neq 0\}$.

Define two orthogonal projectors in $\mathcal{H}$:

$$\Pi_0 = \chi_0\chi_0^* \qquad \text{and} \qquad \Pi_1 = I - \Pi_0 = \sum_{i=1}^{q-1} \chi_i\chi_i^*.$$

An important relation is

$$\Pi_0 \overset{\Delta}{\longmapsto} \Pi_0 \qquad \text{and} \qquad \Pi_1 \overset{\Delta}{\longmapsto} -\Pi_0, \tag{10}$$

where Δ is as in (1) and acts on the sole variable, and $\overset{\Delta}{\longmapsto}$ is consequently defined as at the end of Section 2.2. For $S \subseteq T$, define the projector Π_S^T in the space $\mathcal{H}^T$ by

$$\Pi_S^T = \bigotimes\nolimits_{j\in T} \Pi_{1_{j\in S}}. \tag{11}$$

Let $\mathcal{H}_S^T$ be its image. It is equal to the span of all the Fourier basis elements of $\mathcal{H}^T$ with support equal to S. For a fixed T, the set of all $\mathcal{H}_S^T$ gives an orthogonal decomposition of $\mathcal{H}^T$.

We have the following properties of Π_S^T. First, from the definition, we get the union property

$$\Pi_S^T \otimes \Pi_{S'}^{T'} = \Pi_{S\cup S'}^{T\cup T'} \tag{12}$$

whenever T and T' are disjoint. Next, by (10), we get the reduction property

$$\Pi_S^T \overset{\Delta_j}{\longmapsto} \begin{cases} \Pi_S^T, & \text{if } j \notin S; \\ -\Pi_{S\setminus\{j\}}^T, & \text{if } j \in S. \end{cases} \tag{13}$$

4.4 The Building Blocks

Now let us describe the building blocks our matrices are comprised of. Assume that $(a,b,c) \in A \times B \times C$, and $S \subset \{a,b,c\}$ is of size $|S| \leq 1$. We define

$$\Psi_S^{a,b,c} = \sqrt{q}\,\Pi_S^{a,b,c}[\![\widetilde{P}^{a,b,c}, [q]^{\{a,b,c\}}]\!]. \tag{14}$$

These are the matrices from (11) with $T = \{a,b,c\}$ whose rows have been restricted to the solution set $\widetilde{P}^{a,b,c}$ of the corresponding tripartite equation. The factor $\sqrt{q}$ is due to normalisation purposes.

▷ Claim 10. The operator $\Psi_S^{a,b,c}$ is an isometry from $\mathcal{H}_S^{a,b,c}$ into $\mathbb{C}^{\widetilde{P}^{a,b,c}}$. Moreover, the operators $\Psi_\emptyset^{a,b,c}$, $\Psi_{\{a\}}^{a,b,c}$, $\Psi_{\{b\}}^{a,b,c}$, and $\Psi_{\{c\}}^{a,b,c}$ have pairwise orthogonal ranges.

Proof. From the definition, it is clear that the coimage of $\Psi_S^{a,b,c}$ is contained in the coimage of $\Pi_S^{a,b,c}$, which is $\mathcal{H}_S^{a,b,c}$. The operators $\Psi_\emptyset^{a,b,c}$, $\Psi_{\{a\}}^{a,b,c}$, $\Psi_{\{b\}}^{a,b,c}$, and $\Psi_{\{c\}}^{a,b,c}$ map the corresponding Fourier basis elements

$$\chi_0 \otimes \chi_0 \otimes \chi_0, \quad \chi_{s_a} \otimes \chi_0 \otimes \chi_0, \quad \chi_0 \otimes \chi_{s_b} \otimes \chi_0, \quad \chi_0 \otimes \chi_0 \otimes \chi_{s_c},$$

where s_a, s_b, s_c are non-zero, into the vectors

$$\begin{aligned} &\sqrt{q}(\chi_0 \otimes \chi_0 \otimes \chi_0)[\![\widetilde{P}^{a,b,c}]\!], \qquad \sqrt{q}(\chi_{s_a} \otimes \chi_0 \otimes \chi_0)[\![\widetilde{P}^{a,b,c}]\!], \\ &\sqrt{q}(\chi_0 \otimes \chi_{s_b} \otimes \chi_0)[\![\widetilde{P}^{a,b,c}]\!], \qquad \sqrt{q}(\chi_0 \otimes \chi_0 \otimes \chi_{s_c})[\![\widetilde{P}^{a,b,c}]\!], \end{aligned} \tag{15}$$

respectively. It remains to prove that all these vectors together form an orthonormal system in $\mathbb{C}^{\widetilde{P}^{a,b,c}}$.

We can identify $x \in \widetilde{P}^{a,b,c}$ with $x \in [q]^{\{a,b\}}$ as the third element x_c is uniquely determined by $x_c = r_{a,b,c} - x_a - x_b$. Therefore, we may treat the vectors from (15) as belonging to $\mathcal{H}^{a,b}$. Under this assumption, the first three vectors in (15) become

$$\chi_0 \otimes \chi_0, \qquad \chi_{s_a} \otimes \chi_0, \qquad \text{and} \qquad \chi_0 \otimes \chi_{s_b}. \tag{16}$$

(Here we used the $\sqrt{q}$ prefactor to compensate for one missing χ_0.) Considering the last vector, its entry corresponding to $x \in \widetilde{P}^{a,b,c}$ is

$$\tfrac{1}{q}\omega_q^{s_c x_c} = \tfrac{1}{q}\omega_q^{s_c(r_{a,b,c}-x_a-x_b)} = \frac{\omega_q^{s_c r_{a,b,c}}}{q}\omega_q^{-s_c x_a - s_c x_b}.$$

Hence, the last vector of (15) becomes

$$\omega_q^{s_c r_{a,b,c}}\, \chi_{-s_c} \otimes \chi_{-s_c}. \tag{17}$$

Clearly, the vectors in (16) and (17) form an orthonormal system. ◁

Now, let $\mu \in M$, and $S \subseteq [3n]$ be good for μ, i.e., $|S \cap \{a,b,c\}| \leq 1$ for every $(a,b,c) \in \mu$. We define the operator

$$\Psi_S^\mu = q^{n/2}\Pi_S^{[3n]}[\![\widetilde{P}^\mu, \widetilde{N}]\!] = \bigotimes_{(a,b,c)\in\mu} \Psi^{a,b,c}_{S\cap\{a,b,c\}}, \tag{18}$$

where the equality follows from the union property (12) and the definition (5) of $\widetilde{P}^\mu$.

▷ Claim 11 (Orthogonal Isometry Claim). The operator Ψ_S^μ is an isometry from $\mathcal{H}_S^{[3n]}$ into $\mathbb{C}^{\widetilde{P}^\mu}$. Moreover, for a fixed μ, the ranges of Ψ_S^μ are pairwise orthogonal.

Proof. This follows from Claim 10 and the second definition of Ψ_S^μ in (18). ◁

Also, from (13) and the first definition of Ψ_S^μ in (18), we get the reduction property

$$\Psi_S^\mu \xmapsto{\Delta_j} \begin{cases} \Psi_S^\mu, & \text{if } j \notin S; \\ -\Psi^\mu_{S\setminus\{j\}}, & \text{if } j \in S. \end{cases} \tag{19}$$

4.5 The Matrix $G(\alpha)$

Now we are able to define the matrix $G(\alpha)$ for α from (6). It is a $\widetilde{P} \times \widetilde{N}$-matrix defined as the vertical stack of matrices

$$G(\alpha) = \begin{pmatrix} G^{\mu_{1,1}}(\alpha) \\ \vdots \\ G^{\mu_{n,n}}(\alpha) \end{pmatrix}, \tag{20}$$

where we have one block

$$G^\mu(\alpha) = \sum_{S\subseteq[3n]} \alpha(\mu, S)\Psi_S^\mu$$

for each shifted tripartite matching $\mu \in M$. Recall that we implicitly assume that $\alpha(\mu, S) = 0$ if S is not good for μ, therefore, the Ψ_S^μ that appear in the latter sum are well-defined (satisfy the conditions above (18)).

For two α and α' as in (6), we can define $\alpha + \alpha'$ element-wise, and $G(\alpha)$ is linear in α: $G(\alpha + \alpha') = G(\alpha) + G(\alpha')$.

▷ Claim 12. Thus defined matrix $G(\alpha)$ satisfies the claims in (8): $\|G(\alpha)\| = \|\alpha\|$ and $G(\alpha) \overset{\Delta_j}{\longmapsto} G(\partial_j \alpha)$ for all $j \in [3n]$.

Proof. Let us start with the first claim. We can write $G(\alpha) = \sum_S G(\alpha|_S)$, where

$$\alpha|_S(\mu, S') = \begin{cases} \alpha(\mu, S'), & \text{if } S' = S \\ 0, & \text{otherwise.} \end{cases} \tag{21}$$

We have that $G^\mu(\alpha|_S) = \alpha(\mu, S)\Psi_S^\mu$. Since by the Orthogonal Isometry Claim 11, each Ψ_S^μ is an isometry from $\mathcal{H}_S^{[3n]}$, we get

$$\big\|G(\alpha|_S)\big\| = \sqrt{\sum\nolimits_{\mu \in M} \alpha(\mu, S)^2}.$$

By the same Claim 11, the ranges and coimages of all $G(\alpha|_S)$ are pairwise orthogonal. Hence, $\|G(\alpha)\| = \max_S \big\|G(\alpha|_S)\big\| = \|\alpha\|$ as defined in (7).

By the reduction property (19), the second claim holds for each block of $G(\alpha)$, that is:

$$G^\mu(\alpha) \overset{\Delta_j}{\longmapsto} G^\mu(\partial_j \alpha).$$

Therefore, it holds for the whole matrix $G(\alpha)$. ◁

4.6 Construction of α

We define

$$\alpha(\mu, S) = \frac{1}{n} \max\Big\{n^{1/3} - |S|, 0\Big\} \tag{22}$$

if S is good for μ. Otherwise, we assume $\alpha(\mu, S) = 0$.

▷ Claim 13. The α as defined in (22) satisfies the conditions in (9): $\|\alpha\| = n^{1/3}$ and $\|\partial_j \alpha\| = O(1)$ for all $j \in [3n]$.

Proof. The value $\|\alpha\| = n^{1/3}$ is attained at $S = \emptyset$.

Now let us prove the second property. It suffices to check that, for all S and $j \in [3n] \setminus S$,

$$\sum\nolimits_{\mu \in M} \big(\alpha(\mu, S) - \alpha(\mu, S \cup \{j\})\big)^2 = O(1). \tag{23}$$

Fix S and j. If $|S| \geq n^{1/3}$, then (23) is zero, so we may assume $|S| \leq n^{1/3}$. There are n^2 choices of $\mu \in M$. They fall into three categories:

- S is not good for μ. Then, $S \cup \{j\}$ is also not good for μ and
$$\alpha(\mu, S) - \alpha(\mu, S \cup \{j\}) = 0.$$
- $S \cup \{j\}$ is good for μ. Then, S is also good for μ, and
$$\big|\alpha(\mu, S) - \alpha(\mu, S \cup \{j\})\big| \leq \frac{1}{n}.$$
(The $\leq$ case may hold for $|S| = \lfloor n^{1/3} \rfloor$. Otherwise, we have equality.)

- S is good for μ, but $S \cup \{j\}$ is not good for μ. In this case, we have an upper bound of

$$\big|\alpha(\mu, S) - \alpha(\mu, S \cup \{j\})\big| = \big|\alpha(\mu, S)\big| \leq n^{-2/3}.$$

Let us estimate for how many μ the third option above holds. This happens only if there is $(a, b, c) \in \mu$ such that $j \in \{a, b, c\}$ and $\big|S \cap \{a, b, c\}\big| = 1$. Assume for concreteness $j = a \in A = \{1, \ldots, n\}$, the other two cases being similar. Let $\mu = \mu_{s,t}$ as defined in (4). We get that the third option holds only if

$$\text{either} \quad n + 1 + (j + s \bmod n) \quad \text{or} \quad 2n + 1 + (j + t \bmod n) \quad \text{belongs to } S.$$

There are at most $|S|n \leq n^{4/3}$ choices of s and t that satisfy the above condition. Therefore,

$$\sum_{\mu \in M} \big(\alpha(\mu, S) - \alpha(\mu, S \cup \{j\})\big)^2 \leq n^2 \cdot \frac{1}{n^2} + n^{4/3} \cdot n^{-4/3} = O(1). \qquad \lhd$$

4.7 Finishing the Proof

As mentioned previously, we define the adversary matrix as

$$\Gamma = G(\alpha)[\![P, N]\!]$$

with the choice of α from (22). Then, $\Gamma \xmapsto{\Delta_j} G(\partial_j \alpha)[\![P, N]\!]$ and for the latter matrix by (8) and (9), we have

$$\big\|G(\partial_j \alpha)[\![P, N]\!]\big\| \leq \big\|G(\partial_j \alpha)\big\| = \|\partial_j \alpha\| = O(1).$$

It remains to lower bound $\|\Gamma\|$. We can write $\alpha = \alpha' + \alpha''$ where $\alpha' = \alpha|_\emptyset$ as in (21) and $\alpha'' = \alpha - \alpha'$. Let u_P, $u_{\widetilde{P}}$, u_N, and $u_{\widetilde{N}}$ denote the uniform unit vectors in $\mathbb{C}^P$, $\mathbb{C}^{\widetilde{P}}$, $\mathbb{C}^N$ and $\mathbb{C}^{\widetilde{N}}$, respectively. That is, $u_P[\![x]\!] = 1/\sqrt{|P|}$ for all $x \in P$, and similarly for other vectors. We have

$$\|\Gamma\| \geq u_P^* \Gamma u_N = u_P^* \big(G(\alpha')[\![P, N]\!]\big) u_N + u_P^* \big(G(\alpha'')[\![P, N]\!]\big) u_N. \tag{24}$$

We bound both terms separately.

By construction, $G(\alpha')$ is an $\widetilde{P} \times \widetilde{N}$ matrix which is a vertical stack of matrices (20), where each block is $n^{-2/3}\Psi_\emptyset^\mu$. Each $\Psi_\emptyset^\mu$ has all its entries equal (to $q^{-5n/2}$). Thus,

$$u_{\widetilde{P}}^* G(\alpha') u_{\widetilde{N}} = \|G(\alpha')\| = \|\alpha'\| = n^{1/3}.$$

Using Claim 9, we get

$$u_P^* \big(G(\alpha')[\![P, N]\!]\big) u_N = \sqrt{\frac{|P| \cdot |N|}{|\widetilde{P}| \cdot |\widetilde{N}|}}\, u_{\widetilde{P}}^* G(\alpha') u_{\widetilde{N}} \geq \frac{3n^{1/3}}{4}. \tag{25}$$

Now consider the second term. First, we have

$$\|G(\alpha'')\| = \|\alpha''\| < n^{1/3}.$$

Next, by Claim 11, the vector $u_{\widetilde{P}}$ is orthogonal to the range of $G(\alpha'')$. Therefore,

$$u_{\widetilde{P}}^* G(\alpha'')[\![\widetilde{P}, N]\!] = 0.$$

We will use the vector $\widetilde{u} = \sqrt{|\widetilde{P}|/|P|}u_{\widetilde{P}}$. It has the property $\widetilde{u}[\![P]\!] = u_P$. Also, by Claim 9, $\big\|\widetilde{u}[\![\widetilde{P} \setminus P]\!]\big\| \leq 1/\sqrt{3}$. Thus,

$$\begin{aligned} u_P^*\big(G(\alpha'')[\![P,N]\!]\big)u_N &= \widetilde{u}^*\big(G(\alpha'')[\![\widetilde{P},N]\!]\big)u_N - \widetilde{u}[\![\widetilde{P}\setminus P]\!]^*\big(G(\alpha'')[\![\widetilde{P}\setminus P,N]\!]\big)u_N \\ &\geq -\big\|\widetilde{u}[\![\widetilde{P}\setminus P]\!]\big\| \cdot \big\|G(\alpha'')[\![\widetilde{P}\setminus P,N]\!]\big\| \geq -n^{1/3}/\sqrt{3}. \end{aligned} \tag{26}$$

Combining Eq. (24), (25) and (26), we get

$$\|\Gamma\| \geq \Big(\frac{3}{4} - \frac{1}{\sqrt{3}}\Big)n^{1/3} = \Omega(n^{1/3}).$$

5 Discussion

The choice of the problem (3) has been chiefly motivated by the availability of a relatively simple lower bound in [11]. In principle, it is possible to analyse other problems. For instance, consider a problem of the form

$$x_a + x_b = r_{a,b}, \qquad \text{for } a \neq b \text{ in } [2n], \tag{27}$$

and take M as consisting of all perfect matchings in $[2n]$.

If $+$ is the bit-wise xor and all $r_{a,b}$ are zeroes, we get the collision problem. The lower bound on its quantum query complexity is $\Omega(n^{1/3})$ and it was obtained by Aaronson and Shi [5] using the polynomial method. We see no reason to expect the non-homogeneous case (with $r_{a,b}$ non-zero) to be any simpler, but the argument of Section 3 shows that it completely breaks down Aaronson's and Shi's lower bound. For the adversary method, which we used in Section 4, there was essentially no difference between the homogeneous case of [11] and the non-homogeneous case of the current paper. The right-hand sides $r_{a,b,c}$ of (3) manifested themselves as the phases in (17), which are irrelevant to the proof. It would be interesting to establish lower bounds for the problem in (27). This would improve the constants in our lower bounds.

Concerning Corollary 2, it is not clear whether $O(\log n)$ can be improved to something better, thus resulting in a superexponential separation.

References

1 Scott Aaronson. Quantum lower bound for the collision problem. In *Proc. of 34th ACM STOC*, pages 635–642, 2002. doi:10.1145/509907.509999.

2 Scott Aaronson. Open problems related to quantum query complexity. *ACM Transactions on Quantum Computing*, 2(4):1–9, 2021. doi:10.1145/3488559.

3 Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proc. of 48th ACM STOC*, pages 863–876, 2016. doi:10.1145/2897518.2897644.

4 Scott Aaronson, Shalev Ben-David, Robin Kothari, Shravas Rao, and Avishay Tal. Degree vs. approximate degree and quantum implications of Huang's sensitivity theorem. In *Proc. of 53rd ACM STOC*, pages 1330–1342, 2021. doi:10.1145/3406325.3451047.

5 Scott Aaronson and Yaoyun Shi. Quantum lower bounds for the collision and the element distinctness problems. *Journal of the ACM*, 51(4):595–605, 2004. doi:10.1145/1008731.1008735.

6 Andris Ambainis. Quantum lower bounds by quantum arguments. *Journal of Computer and System Sciences*, 64(4):750–767, 2002. doi:10.1006/jcss.2002.1826.

7 Andris Ambainis. Polynomial degree vs. quantum query complexity. In *Proc. of 44th IEEE FOCS*, pages 230–239, 2003. doi:10.1109/SFCS.2003.1238197.

8 Anurag Anshu, Shalev Ben-David, and Srijita Kundu. On query-to-communication lifting for adversary bounds. In *Proc. of 36th IEEE CCC*, volume 200 of *LIPIcs*, pages 30:1–30:39, 2021. doi:10.4230/LIPIcs.CCC.2021.30.

9 Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. doi:10.1145/502090.502097.

10 Aleksandrs Belovs and Ansis Rosmanis. On the power of non-adaptive learning graphs. *Computational Complexity*, 23(2):323–354, 2014. doi:10.1007/s00037-014-0084-1.

11 Aleksandrs Belovs and Ansis Rosmanis. Quantum lower bounds for tripartite versions of the hidden shift and the set equality problems. In *Proc. of 13th TQC*, volume 111 of *LIPIcs*, pages 3:1–3:15. Dagstuhl, 2018. doi:10.4230/LIPIcs.TQC.2018.3.

12 Aleksandrs Belovs and Robert Špalek. Adversary lower bound for the k-sum problem. In *Proc. of 4th ACM ITCS*, pages 323–328, 2013. doi:10.1145/2422436.2422474.

13 Harry Buhrman and Ronald de Wolf. Complexity measures and decision tree complexity: a survey. *Theoretical Computer Science*, 288:21–43, 2002. doi:10.1016/S0304-3975(01)00144-X.

14 Mark Bun, Robin Kothari, and Justin Thaler. The polynomial method strikes back: Tight quantum query bounds via dual polynomials. In *Proc. of 50th ACM STOC*, pages 297–310, 2018. doi:10.1145/3188745.3188784.

15 Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proc. of 39th ACM STOC*, pages 526–535, 2007. doi:10.1145/1250790.1250867.

16 Troy Lee, Rajat Mittal, Ben W. Reichardt, Robert Špalek, and Mario Szegedy. Quantum query complexity of state conversion. In *Proc. of 52nd IEEE FOCS*, pages 344–353, 2011. doi:10.1109/FOCS.2011.75.

17 Nikhil S. Mande, Justin Thaler, and Shuchen Zhu. Improved approximate degree bounds for k-distinctness. In *Proc. of 15th TQC*, volume 158 of *LIPIcs*, pages 2:1–2:22, 2020. doi:10.4230/LIPIcs.TQC.2020.2.

18 Gatis Midrijānis. Exact quantum query complexity for total Boolean functions. quant-ph/0403168, 2004.

19 Noam Nisan. CREW PRAMs and decision trees. *SIAM Journal on Computing*, 20(6):999–1007, 1991. doi:10.1137/0220062.

20 Noam Nisan and Mario Szegedy. On the degree of Boolean functions as real polynomials. *Computational Complexity*, 4(4):301–313, 1994. doi:10.1007/BF01263419.

Trade-Offs Between Entanglement and Communication

Srinivasan Arunachalam
IBM Quantum, Almaden, CA, USA

Uma Girish
Princeton University, NJ, USA

Abstract

We study the advantages of quantum communication models over classical communication models that are equipped with a limited number of qubits of entanglement. In this direction, we give explicit partial functions on n bits for which reducing the entanglement increases the classical communication complexity exponentially. Our separations are as follows. For every $k \geq 1$:

$\mathsf{Q}\|^*$ versus $\mathsf{R2}^*$: We show that quantum simultaneous protocols with $\tilde{\Theta}(k^5 \log^3 n)$ qubits of entanglement can exponentially outperform two-way randomized protocols with $O(k)$ qubits of entanglement. This resolves an open problem from [11] and improves the state-of-the-art separations between quantum simultaneous protocols with entanglement and two-way randomized protocols without entanglement [12, 18].

$\mathsf{R}\|^*$ versus $\mathsf{Q}\|^*$: We show that classical simultaneous protocols with $\tilde{\Theta}(k \log n)$ qubits of entanglement can exponentially outperform quantum simultaneous protocols with $O(k)$ qubits of entanglement, resolving an open question from [15, 12]. The best result prior to our work was a relational separation against protocols without entanglement [15].

$\mathsf{R}\|^*$ versus $\mathsf{R1}^*$: We show that classical simultaneous protocols with $\tilde{\Theta}(k \log n)$ qubits of entanglement can exponentially outperform randomized one-way protocols with $O(k)$ qubits of entanglement. Prior to our work, only a relational separation was known [11].

2012 ACM Subject Classification Theory of computation → Communication complexity; Theory of computation → Quantum communication complexity

Keywords and phrases quantum, communication complexity, exponential separation, boolean hidden matching, forrelation, xor lemma

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.25

Related Version *Full Version*: https://arxiv.org/abs/2306.01233

Acknowledgements We thank Vojtech Havlicek and Ran Raz for many discussions during this project. We also thank Ran Raz for feedback on the presentation.

1 Introduction

One of the central goals in complexity theory is to understand the power of different computational resources. In the past four decades, communication complexity has provided a successful toolbox to establish various results in different areas of research in theoretical computer science such as circuit complexity [26, 25], streaming algorithms [24], property testing [6], extension complexity [10], data structures [29], proof complexity [21]. In the standard two-player model of communication complexity introduced by Yao [40] there are two parties Alice and Bob whose goal is to compute a partial function $F : \mathcal{X} \times \mathcal{Y} \to \{-1, 1, \star\}$. Alice receives $x \in \mathcal{X}$ (unknown to Bob) and Bob receives $y \in \mathcal{Y}$ (unknown to Alice) and their goal is to compute $F(x, y)$ for all $(x, y) \in F^{-1}(1) \cup F^{-1}(-1)$, while minimizing the amount of communication. In this setting, there are three models of communication in increasing order of strength:

(i) Simultaneous message passing (SMP) model: Alice and Bob send a message to a referee Charlie, whose goal is to output $F(x, y)$.
(ii) One-way model: Alice sends a message to Bob, whose goal is to output $F(x, y)$.
(iii) Two-way model: Alice and Bob can exchange several rounds of messages and their goal is to output $F(x, y)$.

In all these models, the complexity of the protocol is the total number of bits used to describe the message. It is not hard to see that the communication complexity in model (i) is at least the complexity in model (ii) which in turn is at least the complexity in model (iii).

One variant of these models is when the players are allowed to use *quantum resources*, for instance, the players could send quantum messages or share entanglement. Over the past two decades, several works have established the advantage of quantum over classical communication complexity in various settings. In a sequence of works [8, 7, 34, 15, 14, 27, 13], it has been shown that quantum communication can exponentially outperform classical communication. In particular, a few works [16, 12, 18] have demonstrated communication tasks that are easy to solve in the SMP model if the players share entanglement, however, every interactive randomized protocol without entanglement has exponentially larger cost. This leads to a natural and fundamental question (which has been asked many times before [22, 9, 37, 11]): *How much entanglement do quantum protocols really need?* Given any small-cost quantum protocol, can we simulate it by a small-cost quantum protocol that uses only a small amount of entanglement? Answering this question is one of the central questions in quantum communication complexity; in fact giving *any* upper bound on the number of qubits in a potentially helpful shared state has been open for decades.

A similar question of how much shared *randomness* is necessary in classical communication complexity is well understood. In a famous result, Newman [31] showed that to solve communication tasks on n-bit inputs, with an additive overhead of $O(\log n)$ bits in communication one can assume that the players only have private randomness. Jain et al. [23] showed that blackbox arguments similar to the one in [31] cannot be used to reduce the entanglement in a quantum protocol. Motivated by the question of how much entanglement protocols need, we study a fine-grained variant of this question, which will be the topic of this work.

> *Can we reduce the entanglement in a quantum communication protocol from k qubits to $k/\log n$ qubits using a classical protocol of only polynomially larger cost?*

In this direction, Shi [37] showed that we can remove any amount of entanglement using a classical communication protocol of exponentially larger cost. Subsequently, [11, 22] showed that this exponential blowup is inevitable, in particular they constructed a *relational* problem for which we cannot reduce the entanglement with just a polynomial overhead using one-way communication alone. Their works left open the question of reducing entanglement in a quantum protocol computing a *partial* function, using two-way classical communication between the players.[1]

1.1 Main Result

In this work, we provide a strong negative answer to this question. We give partial functions for which, reducing the entanglement by even a logarithmic factor, increases the communication cost by an exponential factor. To discuss our results, we set up some notation first. Let $\mathsf{R}\|^*$

[1] Relational separations are known as the "weakest" form of separations between communication models. A partial function separation immediately implies a relational separation, however, the converse is false [17].

(resp. $\mathsf{Q}\|^*$) denote the SMP communication model where Alice and Bob share entanglement and send classical (resp. quantum) messages to the referee. Let $\mathsf{R1}^*, \mathsf{R2}^*$ be the one-way and two-way models of classical communication where Alice and Bob share entanglement. The models $\mathsf{R1}$ and $\mathsf{R2}$ are similarly defined with the difference being that Alice and Bob don't share entanglement. The model $\mathsf{Q}\|^{\text{pub}}$ is also defined similarly to $\mathsf{Q}\|^*$ but without entanglement, additionally, the players are allowed public randomness. We first summarize our results informally below. All these results hold for every $k \geq 1$ which is any parameter that is allowed to depend on n.

Our first result shows that for simultaneous quantum protocols, more entanglement cannot be simulated by two-way classical communication with less entanglement (and a polynomial overhead).

▶ **Result 1.** *There is a partial function on $\tilde{O}(kn)$ bits that can be computed in $\mathsf{Q}\|^*$ with $\tilde{O}(k^5 \log^3 n)$ qubits of communication and entanglement, but if the players only share $O(k)$ qubits of entanglement, requires $\Omega(n^{1/4})$ bits of communication in the $\mathsf{R2}^*$ model.*

There are two ways to view this result: (i) It shows that in the rather weak quantum SMP model, reducing the entanglement by a polylogarithmic factor increases the classical communication by an exponential factor, even if Alice and Bob are allowed to interact. This answers an open question in [11]. (ii) This result can also be viewed in the context of quantum versus classical separations in communication complexity. As we mentioned earlier, numerous works [8, 34, 14, 27, 13] have shown that quantum provides exponential savings for partial functions in various settings. The state-of-the-art separations between quantum and classical communication complexity for partial functions are due to [12, 18]; they show separations between $\mathsf{Q}\|^*$ and $\mathsf{R2}$. One drawback of the aforementioned works, in the context of our work, is that the lower bound can only be made to work for protocols where Alice and Bob share $\ll \log n$ qubits of entanglement. We improve upon this by showing separations between $\mathsf{Q}\|^*$ (with more entanglement) and $\mathsf{R2}^*$ (with less entanglement). Our result can thus be seen as the current best-known separation between quantum and classical communication complexity for partial functions. In particular, we give a lower bound technique against $\mathsf{R2}^*$ protocols with $O(\log^c n)$ qubits of entanglement for every $c \in \mathbb{N}$. To the best of our knowledge, there were no known lower bound techniques that distinguished $\mathsf{R2}^*$ (with more entanglement) and $\mathsf{R2}^*$ (with less entanglement) once the number of qubits of entanglement is $\gg \log n$, even for *relational problems*.

Our second result shows that for SMP protocols where the players share entanglement but only send classical messages, entanglement cannot be reduced even by quantum simultaneous protocols or by one-way classical protocols (with a polynomial overhead).

▶ **Result 2.** *There is a partial function on $\tilde{O}(kn)$ bits that can be computed in $\mathsf{R}\|^*$ using $\tilde{O}(k \log n)$ bits of communication and $\tilde{O}(k \log n)$ qubits of entanglement, but if the players share $O(k)$ qubits of entanglement, requires $\Omega(n^{1/3})$ qubits of communication in the $\mathsf{Q}\|^*$ model and $\Omega(\sqrt{n})$ bits in the $\mathsf{R1}^*$ model.*

We remark that the trade-offs obtained in this result are more fine-grained in comparison to Result 1, i.e., our separations hold even if we reduce the entanglement by a $O(\log n)$-factor. Prior to our work, the best known separation between $\mathsf{R}\|^*$ (with more entanglement) and $\mathsf{Q}\|^*$ (with less entanglement) was a relational separation between $\mathsf{R}\|^*$ and $\mathsf{Q}\|^{\text{pub}}$ [15]. Their work left open two questions: (i) Does there exist a *partial function* separating $\mathsf{R}\|^*$ and $\mathsf{Q}\|^{\text{pub}}$? The weaker question of showing a functional separation between $\mathsf{Q}\|^*$ and $\mathsf{Q}\|^{\text{pub}}$ was also open and recently asked by [12]. (ii) Is there a *relational* separation between $\mathsf{Q}\|^*$ (with more

entanglement) and $\mathsf{Q}\|^*$ (with less entanglement)? Our result answers both these questions. Firstly, we prove separations for partial functions improving upon the relational separations; secondly, we also show lower bounds for $\mathsf{Q}\|^*$ with limited entanglement. With regards to separations between $\mathsf{R}\|^*$ (with more entanglement) and $\mathsf{R1}^*$ (with less entanglement), prior to our work these were established in [11, 22], again for relational problems. Gavinsky [11] left open the question of showing a similar separation for *partial* functions and our work resolves this.

In the next two sections, we discuss the problems witnessing these separations followed by the proof sketches. Our first result is based on the Forrelation problem and the second result is based on the Boolean Hidden Matching problem.

1.2 Result 1: Separations based on the Forrelation problem

1.2.1 Problem Definition: The Forrelation Problem

The Forrelation problem was first introduced by Aaronson in the context of query complexity [1] and subsequently has been studied again in the context of separating quantum and classical computation [35, 2]. Variants of the Forrelation problem have been used to show various quantum versus classical separations in communication complexity [18, 3, 36, 19]. The state-of-the-art separations for quantum versus classical communication complexity of partial functions are between $\mathsf{Q}\|^*$ and $\mathsf{R2}$; one such separation is due to [18] and is based on the Forrelation problem, which we define now.

▶ **Definition 1** (Forrelation Function). *Let $n \in \mathbb{N}, n \geq 2$ be a power of two. Let H_n be the (unitary) $n \times n$ Hadamard matrix. For $z_1, z_2 \in \{-1,1\}^{n/2}$, define the* forrelation *function as*

$$\mathrm{forr}(z_1, z_2) = \frac{1}{n}\langle z_2, H_n(z_1)\rangle.$$

Let $\varepsilon \in (0,1]$ be a parameter. We typically set $\varepsilon = \Theta\left(\frac{1}{\log n}\right)$ if it is not specified. We are interested in the communication complexity version of the Forrelation problem defined below.

▶ **Definition 2** (The Forrelation Problem). *In the Forrelation problem, Alice is given $x \in \{-1,1\}^n$, Bob is given $y \in \{-1,1\}^n$. Their goal is to compute $\textsc{forr}(x,y)$ given by*

$$\textsc{forr}(x,y) = \begin{cases} -1 & \mathrm{forr}(x \odot y) \geq \varepsilon/4 \\ 1 & \mathrm{forr}(x \odot y) \leq \varepsilon/8. \end{cases}$$

Here, $\odot$ denotes the pointwise product. Let $k \in \mathbb{N}$ be a parameter satisfying $k = o(n^{1/50})$. We are interested in the XOR of k copies of the Forrelation problem. This problem was first studied in [19] in the context of XOR lemmas.

▶ **Definition 3** ($\oplus^k$-Forrelation Problem). *This problem is the XOR of k independent instances of the Forrelation problem where $\varepsilon = \frac{1}{60k^2 \ln n}$. To be precise, Alice and Bob receive $x = (x^{(1)}, \ldots, x^{(k)})$ and $y = (y^{(1)}, \ldots, y^{(n)})$ where $x^{(i)}, y^{(i)} \in \{-1,1\}^n$ for all $i \in [k]$, and they need to compute*

$$\textsc{forr}^{(\oplus k)}(x,y) = \prod_{i=1}^{k} \textsc{forr}\left(x^{(i)}, y^{(i)}\right).$$

1.2.2 Main Theorem

We now state our main theorem. For $n \in \mathbb{N}$, let $k \in \mathbb{N}$ be a parameter satisfying $k = o(n^{1/50})$.

▶ **Theorem 1.** *The $\oplus^k$-Forrelation problem can be solved with $\tilde{O}(k^5 \log^3 n)$ qubits of communication in the $\mathsf{Q}\|^*$ model if Alice and Bob share $\tilde{\Theta}(k^5 \log^3 n)$ EPR pairs. However, if they share $O(k)$ qubits of entanglement, then this problem requires $\Omega(n^{1/4})$ bits of communication even in the $\mathsf{R2}^*$ model.*

We make a few remarks. Firstly, the upper bound holds provided Alice and Bob share $\tilde{\Theta}(k^5 \log^3 n)$ EPR pairs, however, the lower bound holds for all possible entangled states on $O(k)$ qubits, not necessarily EPR pairs. (See Section 2.2 for a formal description of the models and EPR pairs.) Secondly, although Theorem 1 is stated for bounded-error models, our lower bound also holds for protocols with advantage $2^{-o(k)}$. To prove our lower bound, our main technical contribution is to show a *Fourier growth bound for $\mathsf{R2}^*$ protocols* with limited entanglement.[2] In the following lemma, $O_\ell(t)$ is a shorthand notation for $O(t \cdot 2^{O(\ell)})$.

▶ **Lemma 1.** *Let $C : \{-1,1\}^n \times \{-1,1\}^n \to [-1,1]$ be an $\mathsf{R2}^*$ protocol of cost c where Alice and Bob share an entangled state on at most $2d$ qubits for some parameter $d \in \mathbb{N}$. Let H be the XOR-fiber of C as in Definition 7. Then, for all $\ell \in \mathbb{N}$, we have*

$$L_{1,\ell}(H) \triangleq \sum_{|S|=\ell} \left|\widehat{H}(S)\right| \leq 2^{5d} \cdot O_\ell(c^\ell).$$

We also study lower bounds for the Forrelation problem in the quantum simultaneous model. Prior to our work, there was no partial function separating $\mathsf{Q}\|^*$ with more entanglement and $\mathsf{Q}\|^*$ with less entanglement. In particular, it was unknown whether the Forrelation problem can be solved in the $\mathsf{Q}\|^{\mathrm{pub}}$ model with small cost and no entanglement. Our result shows that this is not the case, and that the Forrelation problem separates $\mathsf{Q}\|^*$ from $\mathsf{Q}\|^{\mathrm{pub}}$, resolving an open problem from [12].

▶ **Theorem 2.** *The Forrelation problem requires $\Omega(n^{1/4})$ qubits of communication in the $\mathsf{Q}\|^{\mathrm{pub}}$ model.*

This is also proved using a Fourier growth bound.

▶ **Lemma 2.** *Let $C : \{-1,1\}^n \times \{-1,1\}^n \to [-1,1]$ be a $\mathsf{Q}\|^{\mathrm{pub}}$ protocol of cost c and let H be its XOR-fiber as in Definition 7. Then, for all $\ell \in \mathbb{N}$, we have*

$$L_{1,\ell}(H) \leq O_\ell\big(c^\ell\big).$$

We remark that our techniques can also show that for the $\oplus^k$-Forrelation problem, if Alice and Bob only share $O(k)$ EPR pairs, they require $\Omega(n^{1/4})$ *qubits* of communication in the $\mathsf{Q}\|^*$ model. We don't show the details of this proof, instead, we prove a stronger result, namely Theorem 3. We give an example of a partial function such that with $\Theta(k \log n)$ EPR pairs, it is solvable in the $\mathsf{R}\|^*$ model with cost $O(k \log n)$, however, with only $O(k)$ qubits of entanglement requires cost $\Omega(n^{1/3})$ even in the $\mathsf{Q}\|^*$ model.

[2] For the introduction, we will loosely say "Fourier growth of communication protocols", when strictly speaking, we are referring to Fourier growth of *XOR-fibers* and other functions associated with communication protocols.

1.3 Result 2: Separations based on Boolean Hidden Matching

1.3.1 Problem Definition: The Boolean Hidden Matching Problem

We first define the Boolean Hidden Hatching problem. The (relational) Hidden Matching problem was first defined by Bar Yossef et al. [4]. The Boolean Hidden Matching problem was defined by Gavinsky et al. [14] in the context of one-way communication complexity and was used to separate the $\mathsf{R}\|^*$ and $\mathsf{R1}$ models. Subsequently this problem and its variants have found several applications, especially in proving streaming lower bounds starting with the seminal work of Kapralov et al. [24]. The Boolean Hidden Matching problem, denoted $\mathrm{BHM}_{m,n}$, is defined as follows. Let $n, m \in \mathbb{N}$ be parameters and $m = \alpha n$ for a small enough constant $\alpha \ll 1$.

▶ **Definition 4** (Boolean Hidden Matching)**.** *Alice gets $x \in \{-1,1\}^n$, Bob gets a matching on $[n]$ with m edges and a string $y \in \{-1,1\}^m$. Their goal is to compute $BHM_{m,n}(x,y,M)$ given by*

$$BHM_{m,n}(x,y,M) = \begin{cases} -1 & \text{if } Mx = \overline{y} \\ 1 & \text{if } Mx = y. \end{cases}$$

Here, we use $Mx \in \{-1,1\}^m$ to denote the vector whose k-th coordinate is $x_{i_k} \cdot x_{j_k}$ for $k \in [m]$, where the edges of M are $(i_1, j_1), \ldots, (i_m, j_m) \in [n]^2$. We also use $\overline{y}$ to denote $-y$.

Below we will be concerned with computing the XOR of k independent copies of $\mathrm{BHM}_{m,n}$.

▶ **Definition 5** ($\oplus^k$-Boolean Hidden Matching Problem)**.** *This problem is the XOR of k independent instances of the Boolean Hidden Matching problem. To be precise, Alice receives $x = (x^{(1)}, \ldots, x^{(k)})$ and Bob receives $y = (y^{(1)}, \ldots, y^{(k)})$ and $M_1, \ldots, M_k$ where $x^{(i)} \in \{-1,1\}^n$, $y^{(i)} \in \{-1,1\}^m$ and M_i is a matching on $[n]$ with m edges for all $i \in [k]$. They need to compute*

$$BHM_{m,n}^{(\oplus k)}(x,y) = \prod_{i=1}^{k} BHM_{m,n}\left(x^{(i)}, y^{(i)}, M_i\right).$$

1.3.2 Main Theorem

We now state our main theorem. Here, $\alpha \ll 1$ is some absolute constant and $k \in \mathbb{N}$ is a parameter, possibly depending on $n \in \mathbb{N}$.

▶ **Theorem 3.** *The $\oplus^k$-Boolean Hidden Matching problem can be solved with $\tilde{O}(k \log n)$ bits of communication in the $\mathsf{R}\|^*$ model if Alice and Bob share $\tilde{\Theta}(k \log n)$ EPR pairs. However, if Alice and Bob only share $O(k)$ qubits of entanglement, then this problem requires*

- *$\Omega(k\sqrt{n})$ bits of communication in the $\mathsf{R1}^*$ model,*
- *$\Omega(kn^{1/3})$ qubits of communication in the $\mathsf{Q}\|^*$ model.*

Similarly to Theorem 1, we make a few remarks. Firstly, the upper bound holds provided Alice and Bob share $\tilde{\Theta}(k \log n)$ EPR pairs, however, the lower bound holds for all possible entangled states on $O(k)$ qubits, not necessarily EPR pairs. Secondly, although Theorem 3 is stated for bounded-error models, our lower bound also holds for protocols with advantage $2^{-o(k)}$. The main technical contribution of this part is to argue that $\mathsf{R1}$ and $\mathsf{Q}\|^{\mathrm{pub}}$ protocols satisfy an XOR lemma with respect to computing the Boolean Hidden Matching problem.

XOR Lemmas

XOR lemmas study the relation between the computational resources of F and the k-fold XOR of F on k independent inputs. In particular, XOR lemmas for communication complexity are of the following format: If cost-t protocols have advantage at most $2/3$ in computing F, then cost-$o(tk)$ protocols have advantage at most $2^{-\Theta(k)}$ in computing the k-fold XOR of F. XOR lemmas provide a framework to construct hard objects in a black-box way and have applications to several areas in theoretical computer science such as one-way functions, pseudorandom generators and streaming algorithms. We prove an XOR lemma for R1 and Q$\|^{\text{pub}}$ protocols with respect to computing the Boolean Hidden Matching problem.

▶ **Lemma 3.** *Let C be any Q$\|^{\text{pub}}$ protocol of cost c. Then its advantage in computing the $\oplus^k$-Boolean Hidden Matching problem is at most $O_k\left(\frac{(c/k)^3}{n}\right)^{k/2} + O_k(n^{-k/2})$.*

▶ **Lemma 4.** *Let C be any R1 protocol of cost c. Then its advantage in computing the $\oplus^k$-Boolean Hidden Matching problem is at most $O_k\left(\frac{(c/k)^2}{n}\right)^{k/2} + O_k(n^{-k/2})$.*

Until very recently [41], we didn't have an XOR lemma for R1 and as far as we are aware we do not have *any* XOR lemmas for the quantum communication model. However we do have direct product and direct sum theorems for classical and quantum communication models (which are strictly weaker than XOR lemmas) and in fact this was used in the prior work of [15, 22]. Our main technical contribution here is an XOR lemma for R1 and Q$\|^{\text{pub}}$ protocols for the Boolean Hidden Matching problem. Only during completion of this project, we were made aware of a recent work by Yu [41] proving an XOR lemma for all constant round classical protocols (hence implying Lemma 4). Given the technicality of his proof, in our paper we present a simple proof for an XOR lemma for the R1 model for the Boolean Hidden Matching problem.

1.4 Proof Sketch

One of the difficulties of proving lower bounds against classical models equipped with entanglement is that these models are quite powerful; using the quantum teleportation protocol, any quantum protocol with q qubits of communication can be classically simulated using q EPR pairs. Thus, all known partial functions that separate quantum and classical communication complexity are easy to classically simulate in the presence of $O(\log^c n)$ EPR pairs for some small constant $c > 0$.

One approach to show a fine-grained separation between protocols with more entanglement and protocols with less entanglement is the following. Consider any communication task F that exhibits an exponential separation between quantum and classical communication complexity. We have many examples of such tasks that are easy with $O(\log n)$ EPR pairs but exponentially harder in the absence of entanglement. Consider the problem of solving k independent and parallel instances of F. Here, the players receive k pairs of inputs (x_i, y_i) and need to compute $F(x_i, y_i)$ *for every* $i \in [k]$. We denote this problem by $F^{(k)}$. The hope is that entanglement obeys a direct sum theorem of sorts, that is, if the players need at least $\Omega(\log n)$ qubits of entanglement to solve the original task, then to solve k independent and parallel instances, they need at least $\Omega(k \log n)$ qubits of entanglement. In particular, we might hope that protocols that compute $F^{(k)}$ using only $O(k)$ qubits of entanglement require exponentially larger cost. There is a way to make this idea work and this was done in [11]. We describe this idea. Assume by contradiction that we have a small-cost protocol computing $F^{(k)}$ using only $O(k)$ qubits of entanglement.

Step 1: Remove entanglement. The first step is to remove all entanglement from this protocol. To do this, we replace the entangled state on $O(k)$ qubits by the maximally mixed state on $O(k)$ qubits. Since the maximally mixed state is unentangled, the resulting protocol effectively uses no entanglement. Furthermore, the mixed state can be viewed as a probability distribution over states, where the original entangled state occurs with probability $2^{-\Theta(k)}$. It follows that this protocol succeeds with probability at least $2^{-\Theta(k)}$.

Step 2: Direct Product Theorems. The second step is to prove a direct product theorem in the absence of entanglement. Direct product theorems in communication complexity are of the following form: If for cost-t protocols, the probability of solving one instance of F is at most $2/3$, then for cost-$o(tk)$ protocols, the probability of solving k parallel and independent instances of F is at most $2^{-\Theta(k)}$. Establishing such theorems is highly non-trivial and for one-way protocols, this was done by [11, 22].

Following this framework, the work of [11] gives examples of relational problems that are easy to solve with $\Theta(k \log n)$ EPR pairs but difficult with only $O(k)$ EPR pairs. One drawback of this approach is that the task $F^{(k)}$ has many output bits, regardless of whether F is a partial function or a relational problem. To get separations for functions with single-bit outputs, we need to modify this approach. We ask the players to solve the XOR of k independent instances of F. Here, the players receive k pairs of inputs (x_i, y_i) and they need to compute $\prod_{i\in[k]} F(x_i, y_i)$. We denote this problem by $F^{(\oplus k)}$. We will show that there is no small-cost protocol solving $F^{(\oplus k)}$ using only $O(k)$ qubits of entanglement. To do this, we assume by contradiction that there exists such a protocol.

Step 1: Remove entanglement. We produce a small-cost protocol for $F^{(\oplus k)}$ that uses no entanglement and has success probability at least $1/2 + 2^{-\Theta(k)}$, i.e., the advantage is at least $2^{-\Theta(k)}$.

Step 2: XOR Lemmas. We establish an XOR lemma for protocols without entanglement. We show that if for cost-t protocols, the probability of solving one instance is at most $2/3$, then for cost-$o(tk)$ protocols, the probability of solving the XOR of k independent instances is at most $1/2 + 2^{-\Theta(k)}$, i.e., the advantage is at most $2^{-\Theta(k)}$.

Together, this would establish the desired result. We now discuss some of the difficulties in executing these steps and present our solutions. We first present the details of step 2 and then step 1.

Details of Step 2. One difficulty with step 2 is that XOR lemmas are stronger than direct product theorems and are thus harder to establish. In this work, we present XOR lemmas that are tailored for particular functions. The functions we will be interested in are the Forrelation problem and the Boolean Hidden Matching problem. For the former problem, XOR lemmas for $\mathsf{R2}$ protocols were established in [19]. For the Boolean Hidden Matching problem, we show an XOR lemma for the $\mathsf{R1}$ and $\mathsf{Q}\|^{\mathrm{pub}}$ models (Lemma 4 and Lemma 3). We now describe this part in more detail.

Let F be the Boolean Hidden Matching problem. One central ingredient in our XOR lemmas for F is the construction of hard distributions. The result of [14] shows hard distributions $\mathcal{Y}$ and $\mathcal{N}$ on the YES and NO instances of F respectively, such that no small-cost protocol can distinguish these distributions with $1/3$ advantage. We produce hard distributions $\mu_{-1}^{(k)}$ and $\mu_1^{(k)}$ for the $F^{(\oplus k)}$ problem such that no small-cost protocol can

distinguish them with advantage $2^{-\Theta(k)}$. To get bounds of the form $2^{-\Theta(k)}$, it turns out that our distributions need to agree on moments of size at most $\Theta(k)$. Motivated by this, we define the following distributions.

$$\mu_1^{(k)} := \frac{1}{2^{k-1}} \sum_{\substack{K\subseteq[k]\\ |K| \text{ is even}}} \mathcal{Y}_K\mathcal{N}_{\overline{K}} \quad \text{and} \quad \mu_{-1}^{(k)} := \frac{1}{2^{k-1}} \sum_{\substack{K\subseteq[k]\\ |K| \text{ is odd}}} \mathcal{Y}_K\mathcal{N}_{\overline{K}},$$

Here, $\mathcal{Y}_K\mathcal{N}_{\overline{K}}$ is a product of k independent distributions, where the i-th distribution is $\mathcal{Y}$ if $i \in K$ and is $\mathcal{N}$. We show that the distributions $\mu_1^{(k)}$ and $\mu_{-1}^{(k)}$ are indeed distributions on the NO and YES instances respectively of the $F^{(\oplus k)}$ problem, furthermore, they agree on all moments of size at most k. To complete the argument, we need to show that small-cost protocols cannot distinguish these distributions with more than $2^{-\Theta(k)}$ advantage. This is fairly technical and involves the use of Fourier analysis. For R1 protocols, the $k = 1$ version of this was proved in [14]. Their work in particular makes use of the level-k inequality. We build on their work for R1 protocols and prove the desired XOR lemma for larger k (Lemma 4). For $\mathsf{Q}\|^{\text{pub}}$ protocols, we are not aware of any works that study the communication complexity of F or that analyze the Fourier spectrum of such protocols, which is a contribution in this paper. In particular we prove Fourier growth bounds for $\mathsf{Q}\|^{\text{pub}}$ protocols (Lemma 2) as well as an XOR lemma for the Boolean Hidden Matching problem (Lemma 3). For these, we make use of a matrix version of the level-k inequality [5].

Details of Step 1. One difficulty with step 1 is that the trick of replacing an entangled state by the maximally mixed state no longer works. It is possible for a protocol to be correct when using a particular entangled state, but wrong for every orthogonal state. In this case, executing the protocol on the maximally mixed state would bias the output towards the wrong answer. Thus, carrying out step 1 is non-trivial and in particular, difficult to do for $\mathsf{R2}^*$ protocols. We take an alternate approach for $\mathsf{R2}^*$ protocols to sidestep this difficulty, which we will describe later. We are able to carry out step 1 for $\mathsf{R1}^*$ and $\mathsf{Q}\|^*$ protocols. Given a cost c protocol for a function in the $\mathsf{R1}^*$ model or $\mathsf{Q}\|^*$ model using at most $2d$ qubits of entanglement, we produce a cost $c + O(d)$ protocol in the R1 or Q‖ model[3] respectively; these protocols use no entanglement and have advantage $2^{-\Theta(d)}$. We now give an illustrative example of this simulation.

Consider a simple $\mathsf{Q}\|^*$ protocol where the entangled state consists of d EPR pairs and Alice and Bob apply a unitary operator to their part of the entangled state and send all their qubits to Charlie. If Alice's and Bob's unitary operators map $|i\rangle$ to $|u_i(x)\rangle$ and $|v_i(y)\rangle$ respectively, then the state received by the referee is the pure state $\sum_{i\in\{-1,1\}^d} |u_i(x)\rangle\,|v_i(y)\rangle$ (ignoring the normalization). We now construct a Q‖ protocol that produces the same state with probability $2^{-\Theta(d)}$, furthermore, Charlie is able to detect when this state was successfully produced. We have Alice and Bob send the pure states $\sum_i |i\rangle\,|u_i(x)\rangle$ and $\sum_j |j\rangle\,|v_j(y)\rangle$ respectively to Charlie. Charlie first projects onto states such that $i = j$ and obtains the pure state $\sum_{i\in\{-1,1\}^d} |i,i\rangle\,|u_i(x)\rangle\,|v_i(y)\rangle$ with probability 2^{-d}. He then applies Hadamard on the first $2d$ qubits and measures. He obtains the outcome $|0^{2d}\rangle$ with probability 2^{-2d} in which the resulting state is the pure state $\sum_i |u_i(x)\rangle\,|v_i(y)\rangle$ as in the original $\mathsf{Q}\|^*$ protocol. We use similar ideas to remove entanglement from arbitrary $\mathsf{Q}\|^*$ protocols. To remove entanglement from an $\mathsf{R1}^*$ protocol, we need to take a different approach which involves Alice sending Bob a random coordinate of a certain density matrix. We omit the details.

[3] We use Q‖ to denote the private-coin version of the $\mathsf{Q}\|^{\text{pub}}$ model.

Alternate Approach to Step 1 for R2* Protocols. We now present an alternative to step 1 for R2* protocols. The idea is to prove Fourier growth bounds for R2* protocols. The results of [19] imply that for protocols whose level-$2k$ Fourier growth is at most α, their advantage in solving the $\oplus^k$-Forrelation problem is at most $\alpha \cdot n^{-k/2} + o(n^{-k/2})$. We directly establish a Fourier growth bound on R2* protocols. In particular, we show that for R2* protocols of communication cost c that use d qubits of entanglement, their level-ℓ Fourier growth is at most $\mathrm{poly}(2^d) \cdot O_\ell(c^\ell)$ (Lemma 1). Choosing $\ell = 2k$, $d = \Theta(k)$ and $c = \Theta(n^{1/4})$ for appropriate constants, we have that the advantage is at most $\mathrm{poly}(2^d) \cdot O_\ell(c^\ell) \cdot n^{-k/2} \ll 1$. This would complete the proof. We now describe how we prove the Fourier growth bound on R2* protocols (Lemma 1).

(1) We first show that if the players share a $2d$-qubit entangled state, then we can decompose the state into a small linear combination of $\mathrm{poly}(2^d)$ many *two*-qubit quantum states that are either unentangled, or locally equivalent to $|\Phi^+\rangle\langle\Phi^+|$, the EPR state. (By locally equivalent we mean that the players can transform one state into the other using local unitaries and no communication.) This gives us the pre-factor of $\mathrm{poly}(2^d)$ in Lemma 1.

(2) We analyze protocols where Alice and Bob share the EPR state and bound the Fourier growth of such protocols. Observe that if they share an unentangled state, the protocol is essentially an R2 protocol and the work of [18] showed Fourier growth for such protocols. We strengthen this result by proving similar Fourier growth for R2* protocols where Alice and Bob share the EPR state. To do this, we study the structure of such protocols. We first show that the expected output of any R2* protocol of cost c where Alice and Bob share the EPR state can be written as

$$C(x,y) = \sum_{z \in \{-1,1\}^c} \alpha_z \cdot \mathrm{Tr}((E_z(x) \otimes F_z(y)) \cdot \rho).$$

where $E_z(x)$ and $F_z(y)$ are positive semidefinite matrices, $\sum_{z\in\{-1,1\}^c} E_z(x) \otimes F_z(y) = \mathbb{I}$, $\alpha_z \in \{-1,1\}$, and $\rho = |\Phi^+\rangle\langle\Phi^+| \otimes |0^{2m}\rangle\langle 0^{2m}|_{AB}$ for some $m \in \mathbb{N}$ that is possibly large. We give some intuition on this expression. The qubits $|0^m\rangle\langle 0^m|_A$ and $|0^m\rangle\langle 0^m|_B$ in ρ correspond to Alice and Bob's private memory respectively and ρ captures the initial state of all the qubits in the system. The matrices $E_z(x) \otimes F_z(y)$ arise out of Alice's and Bob's sequence of quantum operations (i.e., POVMs) in the R2* protocol and the quantity $\mathrm{Tr}(E_z(x) \otimes F_z(y) \cdot \rho)$ captures the probability of the transcript being $z \in \{-1,1\}^c$. The number $\alpha_z \in \{-1,1\}$ is 1 if and only if the transcript z results in the players outputting 1.

We now write out the Fourier expansion of the XOR fiber $H(z) = \mathbb{E}_{x\sim\{-1,1\}^n}[C(x, x+z)]$. Using the convolution property of Fourier coefficients, we can express the Fourier coefficients of $H(z)$ in terms of the Fourier coefficients of $E_z(x), F_z(y)$. In particular, we get

$$\sum_{|S|=\ell} |\widehat{H}(S)| = \sum_{|S|=\ell} \left| \sum_{z\in\{-1,1\}^c} \alpha_z \cdot \mathrm{Tr}\left(\left(\widehat{E_z}(S) \otimes \widehat{F_z}(S)\right) \cdot \rho\right)\right|.$$

We now use the fact that ρ has exactly four non-zero entries. This zeroes out all but four coordinates of $\widehat{E}_z(S) \otimes \widehat{F}_z(S)$ in the above expression. At this point, we use the level-k inequality by Lee [28] to bound each of the coordinates separately in terms of the entries of $E_z(x)$ and $F_z(y)$. Using the fact that $\{E_z(x) \otimes F_z(y)\}_z$ forms a POVM, we can bound the coordinates of these matrices and combine them with the bounds we obtain from the level-k inequality in a nice way.

1.5 Organization

In this version of the paper, we only prove the XOR lemmas for the Boolean Hidden Matching Problem in the R1 and $\mathsf{Q}\|^{\text{pub}}$ models. For more details and the rest of the proofs, see the full version of this paper.

2 Preliminaries

Sets. For $n \in \mathbb{N}$, let $[n] = \{1, \ldots, n\}$. We use $\mathbb{1}$ to denote the indicator function, i.e., for a predicate E, $\mathbb{1}[E]$ is 1 if E is satisfied and 0 otherwise. For a subset $S \subseteq [n]$, we use $\overline{S} := [n] \setminus S$ to denote the complement of S. We denote the $n \times n$ identity matrix by $\mathbb{I}_n$, and we omit the subscript if it is implicit.

Big O Notation. For simplicity in notation, for every $f, g : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ and $\ell \in \mathbb{N}$, we say that $g = O_\ell(f)$ if for some constant $c > 0$, we have $g = O\left(f \cdot 2^{c \cdot \ell}\right)$. We say that $f = \tilde{O}(g)$ (respectively $f = \tilde{\Omega}(g)$) if for some constant $c > 0$, we have $f = O(g \cdot \log^c(g))$ (respectively $f = \Omega(g \cdot \log^{-c}(g))$). We say that $f = \tilde{\Theta}(g)$ if $f = \tilde{O}(g)$ and $f = \tilde{\Omega}(g)$.

Probability Distributions. Let Σ be an alphabet and $\mathcal{D}$ be a probability distribution over Σ. We use $x \sim \mathcal{D}$ to denote x sampled according to $\mathcal{D}$. We use $\text{supp}(\mathcal{D})$ to denote the support of the distribution $\mathcal{D}$. We use $x \sim \Sigma$ to denote a uniformly random sample from Σ. For a function $G : \Sigma \to \mathbb{R}^n$, we use $\mathbb{E}_{x \sim \mathcal{D}}[G(x)]$ to denote the expected value of G when the inputs are drawn according to $\mathcal{D}$. Let $k \in \mathbb{N}$, $S \subseteq [k]$ and $\mathcal{D}, \mathcal{D}'$ be two distributions over Σ. We use $\mathcal{D}_S \mathcal{D}'_{\overline{S}}$ to denote the distribution over Σ^k which is a product of k independent distributions over Σ, where the ith distribution is $\mathcal{D}$ if $i \in S$ and $\mathcal{D}'$ if $i \notin S$ for all $i \in [k]$. For distributions, $\mathcal{D}, \mathcal{D}'$, define the total variation distance as $\|\mathcal{D} - \mathcal{D}'\|_1 := \sum_i |\mathcal{D}(i) - \mathcal{D}'(i)|$.

Norms. Let $k \in \mathbb{N}$. For a vector $v \in \mathbb{R}^n$, we use $\|v\|_k := \left(\sum_{i \in [n]} |v_i|^k\right)^{1/k}$ to denote the ℓ_k-norm of v. For any matrix $M \in \mathbb{R}^{n \times n}$, we use $|M|$ to denote $\sqrt{MM^\dagger}$ and we denote by $\|M\|_1$ the trace norm of M, that is $\|M\|_1 := \text{Tr}(\sqrt{MM^\dagger}) = \text{Tr}(|M|)$. We use $\|M\|_{\text{op}} := \max_{\|v\|_2 = 1}(v^T M v)$ to denote the operator norm of M.

2.1 Quantum information

Quantum States. Let $d \in \mathbb{N}$ and let $\mathcal{H}$ be a Hilbert space of dimension 2^d. This is a vector space defined by the $\mathbb{R}$-span of the orthonormal basis $\{|x\rangle : x \in \{0,1\}^d\}$. We also identify this basis with $\{|i\rangle : i \in [2^d]\}$ using the lexicographic ordering as the correspondence. We use $|0^d\rangle$ to denote the vector $|0, \ldots, 0\rangle$ with d zeroes. Let $\mathcal{P}(\mathcal{H})$ be the set of positive semidefinite matrices in $\mathbb{R}^{2^d \times 2^d}$. Let $\mathcal{S}(\mathcal{H})$ be the set of density operators on $\mathcal{H}$, that is, matrices in $\mathcal{P}(\mathcal{H})$ with trace 1. We typically use ρ and σ to refer to elements of $\mathcal{S}(\mathcal{H})$. The state of a quantum system on d qubits is described by a density operator $\rho \in \mathcal{S}(\mathcal{H})$. For states that are shared between Alice and Bob, we use the subscript A and B on qubits to denote whether Alice or Bob own those qubits.

Quantum State Evolution. Let $\mathcal{H}, \mathcal{H}'$ be Hilbert spaces. The evolution of a quantum state is described by a map $E : \mathcal{S}(\mathcal{H}) \to \mathcal{S}(\mathcal{H}')$ which is CPTP (i.e., completely positive and trace preserving). We use the notation $E(\rho)$ to denote the image of ρ under E. In particular, we will be interested in measurement operators. Any quantum measurement with k outcomes

is specified by k matrices $M_1, \ldots, M_k \in \mathcal{P}(\mathcal{H})$ such that $\sum_{i\in[k]} M_i^\dagger M_i = \mathbb{I}$. The probability of getting outcome $i \in [k]$ is precisely $\mathrm{Tr}(M_i \rho M_i^\dagger)$ and the post measurement state upon obtaining the outcome i is $\frac{M_i \rho M_i^\dagger}{\mathrm{Tr}(M_i \rho M_i^\dagger)}$. We have a correspondence between $\{0,1\}$ and $\{1,-1\}$ defined by $0 \to 1, 1 \to -1$, hence, we will sometimes refer to measurement outcomes "1" and "0" as "-1" and "1" respectively.

Distance between States. Let $\rho, \sigma \in \mathcal{S}(\mathcal{H})$ be density operators. We define the trace distance between ρ and σ to be $\frac{\|\rho-\sigma\|_1}{2}$. We will use the following standard facts about the trace distance. Firstly, the trace distance satisfies triangle inequality. Secondly, the trace distance between ρ and σ is equal to the maximum probability with which these states can be distinguished using any single projective measurement. Thirdly, the following inequality holds as a consequence of the Von-Neumann Inequality.

▶ **Fact 6.** *For any matrices $M, \rho \in \mathbb{R}^{n\times n}$, we have* $\mathrm{Tr}(M\rho) \leq \|M\|_{\mathrm{op}} \cdot \|\rho\|_1$.

2.2 Communication Complexity

The goal in communication complexity is for Alice and Bob to compute a function $F : \mathcal{X} \times \mathcal{Y} \to \{-1, 1, \star\}$. We interpret -1 as a YES and 1 as a NO. We say F is a *total* function if $F(x,y) \in \{-1,1\}$ for all $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, otherwise F is a *partial* function. In this paper we will be mostly concerned with partial functions and denote $\mathsf{dom}(F) = F^{-1}(\{-1,1\})$. Here Alice receives an input $x \in \mathcal{X}$ (unknown to Bob) and Bob receives an input $y \in \mathcal{Y}$ (unknown to Alice) promised that $(x,y) \in \mathsf{dom}(F)$ and their goal is to compute $F(x,y)$ with high probability, i.e., probability at least 2/3. More formally, for any protocol $\mathcal{P}$, we let $\mathrm{cost}(\mathcal{P}, x, y)$ be the communication cost of $\mathcal{P}$ when Alice and Bob are given x, y as inputs. We say that $\mathcal{P}$ computes F, if, for every $(x,y) \in \mathsf{dom}(F)$, the output of the protocol is $F(x,y)$ with probability at least 2/3 (where the probability is taken over the randomness/measurements of the protocol). The communication complexity of F is defined as

$$\min_{\mathcal{P} \text{ computes } F} \max_{(x,y)\in\mathsf{dom}(F)} \mathrm{cost}(\mathcal{P}, x, y).$$

The messages sent are referred to as the transcript of the protocol. We discuss a few models of communication of interest to us.

Simultaneous Message Passing Model. This is a general model of communication called the *simultaneous message passing* (SMP) model. In this model, Alice and Bob each send a single (possibly quantum or randomized) message to a referee Charlie. The goal is for Charlie to output $F(x,y)$ with high probability, i.e., at least 2/3 probability. We measure the cost of a communication protocol by the total number of bits (or qubits) received by Charlie. There are many types of simultaneous protocols.

Quantum versus Classical protocols. We use $\mathsf{R}\|$ to denote the SMP model where the players can only send classical messages to Charlie. We use $\mathsf{Q}\|$ to denote the SMP model where the players can send a quantum message to Charlie.

Public-coin versus Private-coin Protocols. If we allow the players to use public coins, we use the superscript "pub". For instance, $\mathsf{Q}\|^{\mathrm{pub}}$ denotes the public-coin quantum SMP model and $\mathsf{Q}\|$ denotes the private-coin quantum SMP model. Unless otherwise specified, all protocols are private coin protocols.

One-Way Model. In this model, Alice sends a single message to Bob, who should output $F(x, y)$ with probability at least 2/3. The cost of the protocol is the size of message Alice sends. By a classical result of Newman [31], we can assume that all one-way protocols are private-coin protocols with an $O(\log n)$ additive overhead in the communication complexity. We use R1 to denote the one-way model of communication where Alice sends a classical message to Bob.

Two-Way Model. Here Alice and Bob are allowed to exchange messages, and Alice should finally output $F(x, y)$ with probability at least 2/3. The cost of the protocol is the total size of the transcript. As before, by a result of Newman [31], we can assume that all two-way protocols are private-coin protocols with an $O(\log n)$ additive overhead in the communication complexity. We use R2 to denote the model of two-way communication where Alice and Bob exchange classical messages.

We now discuss protocols where Alice and Bob can share an entangled state that is independent of their inputs. One important type of entangled state is the EPR pair: This is the state $|\Phi^+\rangle\langle\Phi^+|$ where $|\Phi^+\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A |0\rangle_B + |1\rangle_A |1\rangle_B)$. Here, the subscript on the qubit denotes which player owns the qubit. The upper bound in all our theorems will be established using quantum protocols where the players share a certain number of EPR pairs. We typically specify the dimension of the state shared by Alice and Bob.

Protocols with entanglement. We use $\mathsf{R}\|^*$ to denote the simultaneous model of communication where Alice and Bob share an entangled state and send classical messages to the referee. The model $\mathsf{Q}\|^*$ is similarly defined, but Alice and Bob can send quantum messages to the referee. We use $\mathsf{R1}^*$ to denote the one-way model where Alice and Bob share entanglement and Alice sends a classical message to Bob. We use $\mathsf{R2}^*$ to denote the two-way model where Alice and Bob share entanglement and the messages are classical. In this model, by the quantum teleportation protocol, the players can exchange a limited number of qubits. Conversely, if the players can exchange a limited number of qubits, then Alice can create EPR pairs by herself and send the corresponding qubits to Bob.

For ease of readability, we summarize all the communication models in the table below.

Table 1 Table of Simultaneous Communication Models.

	Private Coins	Public Coins	Entanglement
Classical Messages	$\mathsf{R}\|\|$	$\mathsf{R}\|\|^{\text{pub}}$	$\mathsf{R}\|\|^*$
Quantum Messages	$\mathsf{Q}\|\|$	$\mathsf{Q}\|\|^{\text{pub}}$	$\mathsf{Q}\|\|^*$

Table 2 Table of One-Way & Two-Way Communication Models.

	One-way Private Coins (≡ Public Coins)	Two-way Private Coins (≡ Public Coins)
Classical Messages	R1	R2
Classical Messages & Entanglement	$\mathsf{R1}^*$	$\mathsf{R2}^*$

2.3 XOR-Fibers of Communication Protocols

▶ **Definition 7.** *Given a communication protocol* $C : \{-1,1\}^n \times \{-1,1\}^n \to [-1,1]$*, the XOR-fiber of* C *is a function* $H : \{-1,1\}^n \to [-1,1]$ *defined at* $z \in \{-1,1\}^n$ *by* $H(z) = \mathbb{E}_{x\sim\{-1,1\}^n}[C(x, x \odot z)]$*, where* $\odot$ *denotes point-wise product.*

The communication complexity of XOR functions are well-studied and have connections to the log-rank conjecture, parity decision trees, lifting theorems and separations between quantum and classical communication complexity [30, 20, 39, 38, 42]. XOR-fibers of communication protocols naturally arise in the study of communication complexity of XOR functions. Although we are not aware of any published works defining the term "XOR-fiber", this concept has been studied in many works, most notably [18] and [33].

2.4 Fourier analysis

Fourier Analysis on the Boolean Hypercube. We discuss some of the basics of Fourier analysis. Let $f : \{-1,1\}^n \to \mathbb{R}$ be a function. The Fourier decomposition of f is

$$f(x) = \sum_{S\subseteq[n]} \widehat{f}(S)\chi_S(x),$$

where $\chi_S(x) = \prod_{i\in S} x_i$. The *Fourier coefficients* of f are defined as $\widehat{f}(S) = \mathbb{E}_{x\sim\{-1,1\}^n}[f(x)\cdot\chi_S(x)]$. For $\ell \in \mathbb{N}$, the level-ℓ Fourier mass of f is denoted by $L_{1,\ell}(f)$ and is defined as follows.

$$L_{1,\ell}(f) = \sum_{|S|=\ell} \left|\widehat{f}(S)\right|$$

By Fourier growth bounds, we typically mean upper bounds on $L_{1,\ell}(f)$. We will need the following technical lemma, often called the level-k inequality.

▶ **Lemma 8** ([28, Lemma 10])**.** *Let* $f : \{-1,1\}^n \to [-1,1]$ *be a function with* $\mathbb{E}_x[|f(x)|] = \alpha$*. Then for every* $\ell \in \mathbb{N}$*,*

$$\sum_{|S|=\ell} \widehat{f}(S)^2 \leq 4\alpha^2 \cdot (2e \cdot \ln(e/\alpha^{1/\ell}))^\ell.$$

Although Lemma 10 in [28] is only stated for functions with range $[0,1]$, the same proof technique can be applied for functions with range $[-1,1]$. We remark that the bound often invoked [32, 14] is with the upper bound of $O(\alpha^2 \cdot \ln^\ell(1/\alpha))$ (i.e., without the $1/\ell$ exponent on α) which only holds for $\ell \leq 2\ln(1/\alpha)$. However this improved upper bound, proven recently in [28], holds for all $\ell \in \mathbb{N}$. This makes our proofs much simpler and saves some logarithmic factors.

Fourier analysis for Matrix-Valued Functions. The Fourier coefficients of a matrix-valued function $f : \{-1,1\}^n \to \mathbb{R}^{m\times m}$ are defined by

$$\widehat{f}(S) := \mathop{\mathbb{E}}_{x\sim\{-1,1\}^n}[f(x)\cdot\chi_S(x)]$$

for all $S \subseteq [n]$. We also use a matrix version of the level-k inequality.

▶ **Lemma 9** ([5, Theorem 7]). *Let $\mathcal{H}$ be a Hilbert space of dimension 2^c and let $f : \{-1,1\}^n \to \mathcal{S}(\mathcal{H})$ be a density-matrix valued function. Then, for any $\ell \in \mathbb{N}$ such that $\ell \leq 2\ln(2)c$,*

$$\sum_{|S|=\ell} \mathrm{Tr}^2\left(|\widehat{\rho}_S|\right) \leq \left((2e\ln 2)\cdot c/\ell\right)^{\ell}.$$

Let $\mathcal{H}'$ be a Hilbert space that contains $\mathcal{H}$ of dimension $c' \geq c$. We can view f as a function $f : \{-1,1\}^n \to \mathcal{S}(\mathcal{H}')$ by simply appending zeroes to the output matrix. Given any $\ell \in \mathbb{N}$ that is possibly larger than c, set $c' := c + \lceil \frac{\ell}{2\ln 2} \rceil$ so that $\ell \leq 2\ln(2)\cdot c'$. Using this setting of parameters and Lemma 9, we have the following corollary.

▶ **Corollary 10.** *Let $\mathcal{H}$ be a Hilbert space of dimension 2^c and let $f : \{-1,1\}^n \to \mathcal{S}(\mathcal{H})$ be a density-matrix valued function. Then, for any $\ell \in \mathbb{N}$,*

$$\sum_{|S|=\ell} \mathrm{Tr}^2\left(|\widehat{\rho}_S|\right) \leq O_\ell\left((c/\ell)^\ell\right) + O_\ell(1).$$

3 XOR Lemma for Q$\|^{\mathrm{pub}}$ for the Boolean Hidden Matching Problem

Our main technical result in this section is an XOR lemma for Q$\|^{\mathrm{pub}}$ protocols with regards to the Boolean Hidden Matching problem.

▶ **Lemma 3.** *Let C be any Q$\|^{\mathrm{pub}}$ protocol of cost c. Then its advantage in computing the $\oplus^k$-Boolean Hidden Matching problem is at most $O_k\left(\frac{(c/k)^3}{n}\right)^{k/2} + O_k(n^{-k/2})$.*

To prove this lemma, we will make use of some properties which are very similar to those proved in [14]. The proofs of the facts are deferred to the full version of the paper. Let $\mathcal{M}$ be the uniform distribution on matchings on $[n]$ of size $m = \alpha n$ and $\mathcal{U}$ be the uniform distribution on $\{-1,1\}^n$.

▶ **Definition 11** (M matches S). *Let $S \subseteq [nk]$ and $M \in \mathrm{supp}(\mathcal{M}^{\otimes k})$. We say that M matches S if M is an induced perfect matching on S. If M matches S, we use $M(S) \subseteq [mk]$ to denote the subset of edges of this induced perfect matching.*

Observe that the map $T = M(S)$ defines a bijection between sets S that are matched by M and subsets $T \subseteq [mk]$. Furthermore, $|T| = |S|/2$ and for any $i \in [k]$, $|T_i|$ is odd if and only if $|S_i|/2$ is odd. We now define some sets that will be important in the proof.

▶ **Definition 12.** *Let $\mathcal{S}_{n,k} := \{S \subseteq [nk] : \forall i \in [k], |S_i|/2$ is an odd integer$\}$ and $\mathcal{T}_{n,k} := \{T \subseteq [mk] : \forall i \in [k], |T_i|$ is an odd integer$\}$. Define $\mathcal{S}^\ell_{n,k} := \{S \in \mathcal{S}_{n,k} : |S| = 2\ell\}$ and $\mathcal{T}^\ell_{n,k} := \{T \in \mathcal{T}_{n,k} : |T| = \ell\}$ for all $\ell \in [mk]$.*

The aforementioned map $T = M(S)$ provides a bijection between sets $S \in \mathcal{S}^\ell_{n,k}$ that are matched by M and sets $T \in \mathcal{T}^\ell_{n,k}$. The following facts can be proved using techniques in [14]. The details are deferred to the full version of the paper.

▶ **Fact 13.** *Let $S \subseteq [nk]$ and $M \in \mathrm{supp}(\mathcal{M}^{\otimes k})$. Then, for any $w \in \{-1,1\}^{mk}$, the quantity*

$$\mathop{\mathbb{E}}_{x\sim\mathcal{U}^{\otimes k}}\left[\mathbb{1}[Mx = w]\cdot\chi_S(x)\right]$$

is non-zero if and only if M matches S. Furthermore, if it is non-zero, it equals $2^{-mk}\cdot\chi_{M(S)}(w)$.

▶ **Fact 14.** *Let $S \subseteq [nk]$ with $|S| = 2\ell$. Then,*

$$\Pr_{M\sim\mathcal{M}^{\otimes k}}[M \text{ matches } S] \leq O_\ell\left(\frac{\ell^\ell}{(nk)^\ell}\right).$$

We now prove our main lemma of this subsection.

Proof of Lemma 3. We assume that $(c/k)^{3/2} \leq \tau \cdot n^{1/2}$ for some small constant $\tau > 0$, otherwise the statement of the lemma is vacuously true. We will construct distributions on the YES and NO instances of the $\oplus^k$-Boolean Hidden Matching problem such that no small cost $\mathsf{Q}\|$ protocol can distinguish them with considerable advantage. Consider the following two distributions.

- $\mathcal{N}$ is a distribution on NO-instances of $\mathrm{BHM}_{m,n}$: A random sample of $\mathcal{N}$ is of the form (x, M, y) where $x \sim \mathcal{U}$, $M \sim \mathcal{M}$ and $y := Mx$.
- $\mathcal{Y}$ is a distribution on YES-instances of $\mathrm{BHM}_{m,n}$ defined similarly to $\mathcal{N}$ except that $y := \overline{Mx}$.

Define two distribution $\mu_1^{(k)}, \mu_{-1}^{(k)}$ on inputs to the $\oplus^k$-Boolean Hidden Matching problem as follows.

$$\mu_1^{(k)} := \frac{1}{2^{k-1}} \sum_{\substack{K\subseteq[k]\\ |K| \text{ is even}}} \mathcal{Y}_K\mathcal{N}_{\overline{K}} \quad \text{and} \quad \mu_{-1}^{(k)} := \frac{1}{2^{k-1}} \sum_{\substack{K\subseteq[k]\\ |K| \text{ is odd}}} \mathcal{Y}_K\mathcal{N}_{\overline{K}}. \tag{1}$$

Recall that $\mathcal{Y}_K\mathcal{N}_{\overline{K}}$ is a product of k independent distributions, where the i-th distribution is $\mathcal{Y}$ if $i \in K$ and is $\mathcal{N}$ if $i \notin K$. Clearly $\mu_{-1}^{(k)}$ and $\mu_1^{(k)}$ are distributions on the YES and NO instances respectively of the $\oplus^k$-Boolean Hidden Matching problem.

Consider any $\mathsf{Q}\|$ protocol with c qubits of communication and let $\mathcal{H}$ be a Hilbert space of dimension 2^c. Such a protocol can be described by density matrices $\rho(x) \in \mathcal{S}(\mathcal{H})$ and $\sigma_M(y) \in \mathcal{S}(\mathcal{H})$ for every $x \in \{-1,1\}^{nk}, y \in \{-1,1\}^{mk}$ and $M \in \mathrm{supp}(\mathcal{M}^{\otimes k})$. The state received by Charlie on these inputs is precisely $\rho(x) \otimes \sigma_M(y)$. We will show that the trace distance between the states $\mathbb{E}_{(x,M,y)\sim\mu_1^{(k)}}[\rho(x) \otimes \sigma_M(y)]$ and $\mathbb{E}_{(x,M,y)\sim\mu_{-1}^{(k)}}[\rho(x) \otimes \sigma_M(y)]$ is at most $O_k\left(\frac{(c/k)^{3k/2}}{n^{k/2}}\right)$. Since the trace distance measures the maximal distinguishing probability between the two states, this, along with Yao's principle would complete the proof. Towards this, define

$$\Delta := \mathop{\mathbb{E}}_{(x,M,y)\sim\mu_1^{(k)}}[\rho(x) \otimes \sigma_M(y)] - \mathop{\mathbb{E}}_{(x,M,y)\sim\mu_{-1}^{(k)}}[\rho(x) \otimes \sigma_M(y)].$$

Using the definition of $\mu_1^{(k)}$ and $\mu_{-1}^{(k)}$ in Eq. (1), we have

$$\Delta \triangleq \sum_{K\subseteq[k]} \frac{(-1)^{|K|}}{2^{k-1}} \cdot \mathop{\mathbb{E}}_{\substack{x\sim\mathcal{U}^{\otimes k}\\ M\sim\mathcal{M}^{\otimes k}}} \left[\rho(x) \otimes \sigma_M\left(\overline{(Mx)}_K (Mx)_{\overline{K}}\right)\right].$$

We introduce a variable $w \in \{-1,1\}^{mk}$ to represent Mx so that

$$\Delta = \sum_{\substack{w\in\{-1,1\}^{mk}\\ K\subseteq[k]}} \frac{(-1)^{|K|}}{2^{k-1}} \cdot \mathop{\mathbb{E}}_{\substack{x\sim\mathcal{U}^{\otimes k}\\ M\sim\mathcal{M}^{\otimes k}}} \left[\rho(x) \otimes \sigma_M\left(\overline{w}_K w_{\overline{K}}\right) \cdot \mathbb{1}[Mx = w]\right].$$

We expand $\rho(x)$ in the Fourier Basis to obtain

$$\Delta = \sum_{\substack{w\in\{-1,1\}^{mk}\\ K\subseteq[k]\\ S\subseteq[nk]}} \frac{(-1)^{|K|}}{2^{k-1}} \cdot \mathop{\mathbb{E}}_{\substack{x\sim\mathcal{U}^{\otimes k}\\ M\sim\mathcal{M}^{\otimes k}}} \left[\widehat{\rho}(S) \otimes \sigma_M\left(\overline{w}_K w_{\overline{K}}\right) \cdot [\mathbb{1}[Mx = w] \cdot \chi_S(x)]\right].$$

Consider the term $\mathop{\mathbb{E}}_{x\sim\mathcal{U}^{\otimes k}}[\mathbb{1}[Mx=w]\cdot\chi_S(x)]$. By Fact 13, this term is non-zero if and only if M matches S, in which case the term evaluates to $2^{-mk}\cdot\chi_{M(S)}(w)$. Substituting this in the equation above, we have that Δ equals

$$\sum_{\substack{w\in\{-1,1\}^{mk}\\K\subseteq[k]\\S\subseteq[nk]}}\frac{(-1)^{|K|}}{2^{k-1}}\mathop{\mathbb{E}}_{M\sim\mathcal{M}^{\otimes k}}\left[\widehat{\rho}(S)\otimes\sigma_M\left(\overline{w}_K w_{\overline{K}}\right)\cdot 2^{-mk}\cdot\chi_{M(S)}(w)\cdot\mathbb{1}[M\text{ matches }S]\right]. \tag{2}$$

We now expand $\sigma_M\left(\overline{w}_K w_{\overline{K}}\right)$ in the Fourier basis with respect to w. Consider

$$\begin{aligned}\sum_{K\subseteq[k]}(-1)^{|K|}\cdot\sigma_M\left(\overline{w}_K w_{\overline{K}}\right)&=\sum_{K\subseteq[k]}(-1)^{|K|}\cdot\sum_{T\subseteq[mk]}\widehat{\sigma_M}(T)\cdot\chi_T(\overline{w}_K,w_{\overline{K}})\\&=\sum_{\substack{K\subseteq[k]\\T\subseteq[mk]}}(-1)^{|K|}\cdot\widehat{\sigma_M}(T)\cdot\chi_T(w)\cdot(-1)^{\sum_{i\in K}|T_i|}\\&=\sum_{T\subseteq[mk]}\widehat{\sigma_M}(T)\cdot\chi_T(w)\cdot\sum_{K\subseteq[k]}\left[(-1)^{|K|+\sum_{i\in K}|T_i|}\right].\end{aligned}$$

For $i\in[k]$, let $t_i=-1$ if $|T_i|$ is odd and $t_i=1$ if $|T_i|$ is even. Observe that

$$\mathop{\mathbb{E}}_{K\subseteq[k]}\left[(-1)^{|K|+\sum_{i\in K}|T_i|}\right]=\mathop{\mathbb{E}}_{K\subseteq[k]}\left[\chi_K(-t_1,\ldots,-t_k)\right]=\begin{cases}1 & \text{if }\forall i\in[k],t_i=-1,\\0 & \text{otherwise.}\end{cases}$$

Hence, the quantity $\sum_{K\subseteq[k]}\left[(-1)^{|K|+\sum_{i\in K}|T_i|}\right]$ is non-zero if and only if $|T_i|$ is odd for all $i\in[k]$. Furthermore, if it is non-zero, then it equals 2^k. Recall that we defined $\mathcal{T}_{nk}:=\{T\subseteq[mk]:\forall i\in[k],|T_i|\text{ is odd}\}$ in Definition 12. Using this, we have

$$\sum_{K\subseteq[k]}(-1)^{|K|}\cdot\sigma_M\left(\overline{w}_K w_{\overline{K}}\right)=2^k\cdot\sum_{T\in\mathcal{T}_{n,k}}\widehat{\sigma_M}(T)\cdot\chi_T(w). \tag{3}$$

Substituting this in Eq. (2), we have that Δ equals

$$\begin{aligned}&2\sum_{\substack{w\in\{-1,1\}^{mk}\\S\subseteq[nk]}}\mathop{\mathbb{E}}_{M\sim\mathcal{M}^{\otimes k}}\left[\widehat{\rho}(S)\otimes\sum_{T\in\mathcal{T}_{n,k}}\widehat{\sigma_M}(T)\cdot 2^{-mk}\cdot\chi_{M(S)}(w)\cdot\chi_T(w)\cdot\mathbb{1}[M\text{ matches }S]\right]\\&=2\sum_{S\subseteq[nk]}\mathop{\mathbb{E}}_{M\sim\mathcal{M}^{\otimes k}}\left[\widehat{\rho}(S)\otimes\sum_{T\in\mathcal{T}_{n,k}}\widehat{\sigma_M}(T)\cdot\mathop{\mathbb{E}}_{w\sim\{-1,1\}^{mk}}\left[\chi_{M(S)+T}(w)\right]\cdot\mathbb{1}[M\text{ matches }S]\right].\end{aligned}$$

Observe that if M matches S, then $\mathbb{E}_{w\sim\{-1,1\}^{mk}}\left[\chi_{M(S)+T}(w)\right]$ equals 1 if $T=M(S)$ and 0 otherwise. Recall that the sets $S\subseteq[nk]$ such that M matches S and $M(S)\in\mathcal{T}_{n,k}$ are precisely those sets in $\mathcal{S}_{n,k}$ that are matched by M. Hence,

$$\Delta=\sum_{S\in\mathcal{S}_{n,k}}\widehat{\rho}(S)\otimes\mathbb{E}_M\left[\widehat{\sigma_M}(M(S))\cdot\mathbb{1}[M\text{ matches }S]\right].$$

We now upper bound $\|\Delta\|_1$ by triangle inequality as follows.

$$\|\Delta\|_1\leq\sum_{S\in\mathcal{S}_{n,k}}\|\widehat{\rho}(S)\|_1\otimes\mathbb{E}_M\left[\|\widehat{\sigma_M}(M(S))\|_1\cdot\mathbb{1}[M\text{ matches }S]\right].$$

We partition $\mathcal{S}_{n,k}$ and $\mathcal{T}_{n,k}$ into $\sqcup_\ell \mathcal{S}^\ell_{nk}$ and $\sqcup_\ell \mathcal{T}^\ell_{n,k}$ based on the size of the sets as in Definition 12. Observe that every set in $\mathcal{S}_{n,k}$ has size at least $2k$ and every set in $\mathcal{T}_{n,k}$ has size at least k. Thus,

$$\|\Delta\|_1 \leq \sum_{\ell=k}^{mk} \sum_{S \in \mathcal{S}^\ell_{k,n}} \|\widehat{\rho}(S)\|_1 \otimes \mathbb{E}_M \big[\|\widehat{\sigma_M}(M(S))\|_1 \cdot 1[M \text{ matches } S] \big]. \tag{4}$$

We now apply Cauchy-Schwarz to conclude that

$$\begin{aligned}
&\mathop{\mathbb{E}}_{M \sim \mathcal{M}^{\otimes k}} \big[\|\widehat{\sigma_M}(M(S))\|_1 \cdot \mathbb{1}[M \text{ matches } S] \big] \\
&\leq \sqrt{\mathop{\mathbb{E}}_{M \sim \mathcal{M}^{\otimes k}} \big[\|\widehat{\sigma_M}(M(S))\|_1^2 \cdot \mathbb{1}[M \text{ matches } S] \big]} \cdot \sqrt{\Pr_{M \sim \mathcal{M}^{\otimes k}}[M \text{ matches S}]}.
\end{aligned}$$

Fact 14 implies that for any $S \in \mathcal{T}^\ell_{n,k}$, we have $\Pr_{M \sim \mathcal{M}^{\otimes k}}[M \text{ matches } S] \leq O_\ell\left(\frac{\ell^\ell}{(nk)^\ell}\right)$. Substituting this in Eq. (4) implies that

$$\|\Delta\|_1 \leq \sum_{\ell=k}^{mk} \sum_{S \in \mathcal{S}^\ell_{nk}} \|\widehat{\rho}(S)\|_1 \cdot \sqrt{\mathbb{E}_M \left[\|\widehat{\sigma_M}(M(S))\|_1^2 \cdot \mathbb{1}[M \text{ matches } S] \right]} \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right).$$

Again, by Cauchy-Schwarz, we have

$$\|\Delta\|_1 \leq \sum_{\ell=k}^{mk} \sqrt{\sum_{S \in \mathcal{S}^\ell_{n,k}} \|\widehat{\rho}(S)\|_1^2} \cdot \sqrt{\sum_{S \in \mathcal{S}^\ell_{n,k}} \mathbb{E}_M \left[\|\widehat{\sigma_M}(M(S))\|_1^2 \cdot \mathbb{1}[M \text{ matches } S \right]} \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right).$$

By the aforementioned correspondence between sets $S \in \mathcal{S}^\ell_{n,k}$ such that M matches S and sets $T \in \mathcal{T}^\ell_{n,k}$, we have

$$\|\Delta\|_1 \leq \sum_{\ell=k}^{mk} \sqrt{\sum_{S \in \mathcal{S}^\ell_{n,k}} \|\widehat{\rho}(S)\|_1^2} \cdot \sqrt{\sum_{T \in \mathcal{T}^\ell_{n,k}} \mathbb{E}_M \left[\|\widehat{\sigma_M}(T)\|_1^2 \right]} \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right). \tag{5}$$

We now apply the Matrix level-k inequality in Corollary 10 to the functions $\rho : \{-1,1\}^n \to \mathcal{S}(\mathcal{H})$ and $\sigma_M : \{-1,1\}^m \to \mathcal{S}(\mathcal{H})$ where $\mathcal{H}$ is a Hilbert space of dimension 2^c. Corollary 10 implies that

$$\sum_{|S|=2\ell} \|\widehat{\rho}(S)\|_1^2 \leq O_\ell\left((c/\ell)^{2\ell}\right) + O_\ell(1) \quad \text{and} \quad \sum_{|T|=\ell} \|\widehat{\sigma_M}(T)\|_1^2 \leq O_\ell\left((c/\ell)^{\ell}\right) + O_\ell(1).$$

Substituting this in Eq. (5), we get

$$\begin{aligned}
\|\Delta\|_1 &\leq \sum_{\ell=k}^{mk} \sqrt{\sum_{S:|S|=2\ell} \|\widehat{\rho}(S)\|_1^2} \cdot \sqrt{\sum_{T:|T|=\ell} \mathbb{E}_M \left[\|\widehat{\sigma_M}(T)\|_1^2 \right]} \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right) \\
&\leq \sum_{\ell=k}^{mk} \max\left(O_\ell\left(\frac{c^{3\ell/2}}{\ell^{3\ell/2}}\right), O_\ell(1) \right) \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right) \\
&\leq \sum_{\ell=k}^{mk} O_\ell\left(\frac{c^{3\ell/2}}{\ell^\ell (nk)^{\ell/2}}\right) + \sum_{\ell=k}^{mk} O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right).
\end{aligned}$$

Since $\ell \leq mk = \alpha \cdot nk$ for a sufficiently small constant $\alpha > 0$, the function $\ell^{\ell/2}/(nk)^{\ell/2}$ is exponentially decaying for $\ell \in [k, mk]$ and hence the second term is at most $O_k\left(n^{-k/2}\right)$. Our assumption that $(c/k)^{3/2} \leq \tau \cdot n^{1/2}$ for a sufficiently small constant $\tau > 0$ implies that the function $c^{3\ell/2}/(\ell^\ell (nk)^{\ell/2})$ is exponentially decaying for $\ell \in [k, mk]$ and hence, the first term above is at most $O_k\left(\frac{(c/k)^{3k/2}}{n^{k/2}}\right)$. Together, we have

$$\|\Delta\|_1 \leq O_k\left(\frac{(c/k)^{3k/2}}{n^{k/2}}\right) + O_k(n^{-k/2}).$$

This completes the proof of Lemma 3. ◀

4 XOR Lemma for R1 for the Boolean Hidden Matching Problem

In this subsection, we prove Lemma 4 which we restate here for convenience.

▶ **Lemma 4.** *Let C be any R1 protocol of cost c. Then its advantage in computing the $\oplus^k$-Boolean Hidden Matching problem is at most $O_k\left(\frac{(c/k)^2}{n}\right)^{k/2} + O_k(n^{-k/2})$.*

Proof of Lemma 4. The proof of this lemma will be similar to the proof of Lemma 3 and hence we will follow similar notation. Let $z \in \{-1,1\}^c$ be any c-bit message sent by Alice and let $A_z \subseteq \{-1,1\}^{nk}$ be the set of Alice's inputs for which Alice would have sent z to Bob. Let $g(x) = \mathbb{1}[x \in A_z]$. Fix any $M \in \mathrm{supp}(\mathcal{M}^{\otimes k})$. Similar to Lemma 3, let $\mathcal{N}^M(y)$ be the distribution on $y \in \{-1,1\}^{mk}$ induced by sampling $x \sim A_z$ and letting $y = Mx$. Let $\mathcal{Y}^M(y)$ be similarly defined with $y := \overline{Mx}$. So we have that $\mathcal{N}^M(y) = \frac{|\{x \in A_z | Mx = y\}|}{|A_z|}$ and $\mathcal{Y}^M(y) = \frac{|\{x \in A_z | \overline{Mx} = y\}|}{|A_z|}$ for all $y \in \{-1,1\}^{mk}$. Define

$$\mu_1^{(k)} := \frac{1}{2^{k-1}} \sum_{\substack{K \subseteq [k] \\ |K| \text{ is even}}} \mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M \quad \text{and} \quad \mu_{-1}^{(k)} := \frac{1}{2^{k-1}} \sum_{\substack{K \subseteq [k] \\ |K| \text{ is odd}}} \mathcal{Y}_K \mathcal{N}_{\overline{K}}^M. \tag{6}$$

Below we show that for a typical $M \sim \mathcal{M}^{\otimes k}$, these two distributions are close in total variational distance. By arguments similar to [14], this would complete the proof. To this end, let

$$\Delta_{A_z} := \mathop{\mathbb{E}}_{M \sim \mathcal{M}^{\otimes k}} \left[\left\| \mu_1^{(k)} - \mu_{-1}^{(k)} \right\|_1 \right].$$

By Eq. (6), we have $\mu_1^{(k)} - \mu_{-1}^{(k)} = 2^{1-k} \cdot \sum_{K \subseteq [k]} (-1)^{|K|} \mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M$. Hence

$$\Delta_{A_z}^2 \leq 2^{mk} \cdot \mathbb{E}_M \left[\left\| \mu_1^{(k)} - \mu_{-1}^{(k)} \right\|_2^2 \right] = 2^{mk} \cdot \mathbb{E}_M \left[2^{-2k+2} \cdot \left\| \sum_{K \subseteq [k]} \mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M (-1)^{|K|} \right\|_2^2 \right],$$

where the first inequality is by the Cauchy-Schwarz inequality. By Parseval's theorem, we have

$$\Delta_{A_z}^2 \leq 2^{2mk-2k+2} \cdot \mathbb{E}_M \left[\sum_{\substack{T \subseteq [mk] \\ T \neq \emptyset}} \left(\sum_{K \subseteq [k]} \widehat{\mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M}(T) (-1)^{|K|} \right)^2 \right]. \tag{7}$$

Observe that

$$\begin{aligned}
&\widehat{\mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M}(T) \\
&= \frac{1}{2^{mk}} \sum_{y \in \{-1,1\}^{mk}} \left(\mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M\right)(y) \cdot \chi_T(y) \\
&= \frac{1}{2^{mk} \cdot |A_z|} \Big(\left|\left\{x \in A_z | \chi_T\left((Mx)_{\overline{K}} (\overline{Mx})_K\right) = 1\right\}\right| \\
&- \left|\left\{x \in A_z \,\middle|\, \chi_T\left((Mx)_{\overline{K}} (\overline{Mx})_K\right) = -1\right\}\right| \Big) \\
&= \frac{1}{2^{mk} \cdot |A_z|} \Big(\left|\left\{x \in A_z \,\middle|\, \chi_T(Mx) = (-1)^{\sum_{i \in K} |T_i|}\right\}\right| \\
&- \left|\left\{x \in A_z \,\middle|\, \chi_T(Mx) \neq (-1)^{\sum_{i \in K} |T_i|}\right\}\right| \Big) \\
&= \frac{1}{2^{mk}} \sum_{y \in \{-1,1\}^{mk}} \mathcal{N}^{\otimes k}(y) \cdot \chi_T(y) \cdot (-1)^{\sum_{i \in K} |T_i|} \\
&= \widehat{\mathcal{N}^{\otimes k}}(T) \cdot (-1)^{\sum_{i \in K} |T_i|}.
\end{aligned}$$

By an argument analogous to [14, Eq. (3)], we have $\widehat{\mathcal{N}^{\otimes k}}(T) = \frac{2^{nk}}{|A_z| \cdot 2^{mk}} \cdot \widehat{g}(M^\dagger T)$. Hence

$$\sum_{K \subseteq [k]} \widehat{\mathcal{Y}_K^M \mathcal{N}_{\overline{K}}^M}(T) \cdot (-1)^{|K|} = \frac{2^{nk}}{2^{mk} \cdot |A_z|} \cdot \widehat{g}(M^\dagger T) \cdot \sum_{K \subseteq [k]} (-1)^{|K| + \sum_{i \in K} |T_i|}. \tag{8}$$

As we saw in Eq. (3), the term $\sum_{K \subseteq [k]} (-1)^{|K| + \sum_{i \in K} |T_i|}$ is 2^k if $|T_i|$ is odd for all $i \in [k]$ and zero otherwise. Hence, the R.H.S. of Eq. (8) is non-zero only if $T \in \mathcal{T}_{n,k}$ (defined in Definition 12), and in this case equals $\frac{2^{nk}}{2^{mk} \cdot |A_z|} \cdot \widehat{g}(M^\dagger T) \cdot 2^k$. Substituting this in Eq. (7), we have that $\Delta_{A_z}^2$ equals

$$2^{2mk-2k+2} \cdot \mathbb{E}_M \left[\sum_{T \in \mathcal{T}_{n,k}} \frac{2^{2nk+2k}}{2^{2mk} \cdot |A_z|^2} \widehat{g}(M^\dagger T)^2 \right] = 4 \cdot \mathbb{E}_M \left[\sum_{T \in \mathcal{T}_{n,k}} \frac{2^{2n}}{|A_z|^2} \widehat{g}(M^\dagger T)^2 \right].$$

Recall the correspondence between $\mathcal{T}_{n,k}$ and $\mathcal{S}_{n,k}$ as in Definition 12. For every $S \in \mathcal{S}_{n,k}$, there is at most one $T \in \mathcal{T}_{n,k}$ such that $M^\dagger T = S$, furthermore, such a T exists if and only if M matches S. Hence we have that

$$\begin{aligned}
\Delta_{A_z}^2 &\leq 4 \cdot \mathbb{E}_M \left[\sum_{S \in \mathcal{S}_{n,k}} \frac{2^{2n}}{|A_z|^2} \widehat{g}(S)^2 \cdot \mathbb{1}[M \text{ matches } S] \right] \\
&= 4 \cdot \sum_{\ell=k}^{mk} \sum_{S \in \mathcal{S}_{n,k}^\ell} \frac{2^{2n}}{|A_z|^2} \widehat{g}(S)^2 \cdot \Pr_M[M \text{ matches } S] \\
&\leq \sum_{\ell=k}^{mk} \left(\sum_{|S|=2\ell} \frac{2^{2n}}{|A_z|^2} \widehat{g}(S)^2 \right) \cdot O_\ell\left(\frac{\ell^\ell}{(nk)^\ell}\right),
\end{aligned}$$

where we used Fact 14. Let $\mu(A_z) = \frac{|A_z|}{2^n}$. Applying Lemma 8, we have

$$\Delta_{A_z}^2 \leq \sum_{\ell=k}^{mk} \left(2e \cdot \ln\left(\frac{e}{\mu(A_z)^{1/(2\ell)}}\right)\right)^{2\ell} \cdot O_\ell\left(\frac{\ell^\ell}{(nk)^\ell}\right).$$

We now take square root on both sides (and use concavity of the square root function) to get

$$\Delta_{A_z} \leq \sum_{\ell=k}^{mk} \left(2e \cdot \ln\left(\tfrac{e}{\mu(A_z)^{1/(2\ell)}}\right)\right)^{\ell} \cdot O_\ell\left(\tfrac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right).$$

We now multiply both sides by $\mu(A_z)$ and add over all 2^c possibilities for the transcript $z \in \{-1,1\}^c$.

$$\Delta := \sum_{z\in\{-1,1\}^c} \Delta_{A_z} \leq \sum_{z\in\{-1,1\}^c} \mu(A_z) \cdot \sum_{\ell=k}^{mk} \left(2e \cdot \ln\left(\tfrac{e}{\mu(A_z)^{1/(2\ell)}}\right)\right)^{\ell} \cdot O_\ell\left(\tfrac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right).$$

We now use the concavity of the function $h'(\gamma) = \gamma \cdot \ln(e/\gamma^{1/2\ell})^{\ell}$ for $\gamma \in [0,1]$ and all $\ell \in \mathbb{N}$, to conclude that

$$\Delta \leq \sum_{\ell=k}^{mk} \left(\sum_{z\in\{-1,1\}^c} \mu(A_z)\right) \cdot \left(2e \cdot \ln\left(\frac{e \cdot 2^{c/(2\ell)}}{\left(\sum_{z\in\{-1,1\}^c} \mu(A_z)\right)^{1/(2\ell)}}\right)\right)^{\ell} \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right).$$

We use the fact that $\sum_{z\in\{-1,1\}^c} \mu(A_z) = 1$ to conclude that

$$\begin{aligned}\Delta &\leq \sum_{\ell=k}^{mk} \left(2e \cdot \ln\left(e \cdot 2^{c/(2\ell)}\right)\right)^{\ell} \cdot O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right) \\ &\leq \sum_{\ell=k}^{mk} O_\ell\left(\frac{\ell^{\ell/2}}{(nk)^{\ell/2}}\right) + \sum_{\ell=k}^{mk} O_\ell\left(\frac{c^{\ell}}{(\ell nk)^{\ell/2}}\right).\end{aligned}$$

As before, the first term is at most $O(n^{-k/2})$. The assumption that $c \cdot k \leq \tau \cdot n^{1/2}$ for a small enough constant $\tau > 0$ implies that the function $\frac{c^{\ell}}{(\ell nk)^{\ell/2}}$ is exponentially decaying for $\ell \in [k, mk]$. Hence, the second term is at most $O_\ell\left(\frac{(c/k)^k}{n^{k/2}}\right)$. This, along with the techniques of [14] completes the proof of Lemma 4. ◀

References

1. Scott Aaronson. BQP and the polynomial hierarchy. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 141–150, 2010.
2. Scott Aaronson and Andris Ambainis. Forrelation: A problem that optimally separates quantum from classical computing. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 307–316, 2015.
3. Nikhil Bansal and Makrand Sinha. k-forrelation optimally separates quantum and classical query complexity. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1303–1316, 2021.
4. Ziv Bar-Yossef, Thathachar S Jayram, and Iordanis Kerenidis. Exponential separation of quantum and classical one-way communication complexity. *SIAM Journal on Computing*, 38(1):366–384, 2008.
5. Avraham Ben-Aroya, Oded Regev, and Ronald de Wolf. A hypercontractive inequality for matrix-valued functions with applications to quantum computing and ldcs. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 477–486. IEEE, 2008.
6. Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *computational complexity*, 21(2):311–358, 2012.

7 Harry Buhrman, Richard Cleve, John Watrous, and Ronald de Wolf. Quantum fingerprinting. *Physical Review Letters*, 87(16):167902, 2001.

8 Harry Buhrman, Richard Cleve, and Avi Wigderson. Quantum vs. classical communication and computation. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 63–68, 1998.

9 Matthew Coudron and Aram W. Harrow. Universality of EPR pairs in entanglement-assisted communication complexity, and the communication cost of state conversion. In *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA*, volume 137 of *LIPIcs*, pages 20:1–20:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

10 Samuel Fiorini, Serge Massar, Sebastian Pokutta, Hans Raj Tiwary, and Ronald de Wolf. Exponential lower bounds for polytopes in combinatorial optimization. *Journal of the ACM (JACM)*, 62(2):1–23, 2015.

11 Dmitry Gavinsky. On the role of shared entanglement. *Quantum Inf. Comput.*, 8(1):82–95, 2008. doi:10.26421/QIC8.1-2-6.

12 Dmitry Gavinsky. Quantum versus classical simultaneity in communication complexity. *IEEE Transactions on Information Theory*, 65(10):6466–6483, 2019.

13 Dmitry Gavinsky. Bare quantum simultaneity versus classical interactivity in communication complexity. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 401–411, 2020.

14 Dmitry Gavinsky, Julia Kempe, Iordanis Kerenidis, Ran Raz, and Ronald de Wolf. Exponential separations for one-way quantum communication complexity, with applications to cryptography. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 516–525, 2007.

15 Dmitry Gavinsky, Julia Kempe, Oded Regev, and Ronald de Wolf. Bounded-error quantum state identification and exponential separations in communication complexity. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of Computing*, pages 594–603, 2006.

16 Dmytro Gavinsky. Classical interaction cannot replace quantum nonlocality, 2009. doi:10.48550/arXiv.0901.0956.

17 Dmytro Gavinsky, Julia Kempe, and Ronald de Wolf. Strengths and weaknesses of quantum fingerprinting. *CoRR*, 2006. doi:10.48550/arXiv.QUANT-PH/0603173.

18 Uma Girish, Ran Raz, and Avishay Tal. Quantum versus randomized communication complexity, with efficient players. *computational complexity*, 31(2):17, 2022.

19 Uma Girish, Ran Raz, and Wei Zhan. Lower bounds for XOR of forrelations. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM*, volume 207 of *LIPIcs*, pages 52:1–52:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

20 Hamed Hatami, Kaave Hosseini, and Shachar Lovett. Structure of protocols for XOR functions. *SIAM J. Comput.*, 47(1):208–217, 2018.

21 Trinh Huynh and Jakob Nordstrom. On the virtue of succinct proofs: Amplifying communication complexity hardness to time-space trade-offs in proof complexity. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 233–248, 2012.

22 Rahul Jain, Hartmut Klauck, and Ashwin Nayak. Direct product theorems for communication complexity via subdistribution bounds. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing*, pages 599–608, 2007.

23 Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. Prior entanglement, message compression and privacy in quantum communication. In *20th Annual IEEE Conference on Computational Complexity (CCC'05)*, pages 285–296. IEEE, 2005.

24 Michael Kapralov, Sanjeev Khanna, and Madhu Sudan. Streaming lower bounds for approximating max-cut. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1263–1282. SIAM, 2014.

25 Mauricio Karchmer, Ran Raz, and Avi Wigderson. Super-logarithmic depth lower bounds via the direct sum in communication complexity. *Computational Complexity*, 5(3):191–204, 1995.

26 Mauricio Karchmer and Avi Wigderson. Monotone circuits for connectivity require super-logarithmic depth. *SIAM J. Discret. Math.*, 3(2):255–265, 1990. doi:10.1137/0403021.

27 Bo'az Klartag and Oded Regev. Quantum one-way communication can be exponentially stronger than classical communication. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, pages 31–40. Association for Computing Machinery, 2011. doi:10.1145/1993636.1993642.

28 Chin Ho Lee. Fourier bounds and pseudorandom generators for product tests. In *34th Computational Complexity Conference, CCC*, volume 137 of *LIPIcs*, pages 7:1–7:25. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

29 Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. In *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*, pages 103–111, 1995.

30 Ashley Montanaro and Tobias Osborne. On the communication complexity of xor functions, 2010. arXiv:0909.3392.

31 Ilan Newman. Private vs. common random bits in communication complexity. *Information processing letters*, 39(2):67–71, 1991.

32 Ryan O'Donnell. *Analysis of Boolean functions*. Cambridge University Press, 2014.

33 Ran Raz. Fourier analysis for probabilistic communication complexity. *Comput. Complex.*, 5(3/4):205–221, 1995. doi:10.1007/BF01206318.

34 Ran Raz. Exponential separation of quantum and classical communication complexity. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 358–367, 1999.

35 Ran Raz and Avishay Tal. Oracle separation of BQP and PH. *ACM Journal of the ACM (JACM)*, 69(4):1–21, 2022.

36 Alexander A. Sherstov, Andrey A. Storozhenko, and Pei Wu. An optimal separation of randomized and quantum query complexity. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1289–1302. ACM, 2021. doi:10.1145/3406325.3451019.

37 Yaoyun Shi. Tensor norms and the classical communication complexity of nonlocal quantum measurement. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 460–467, 2005.

38 Yaoyun Shi and Zhiqiang Zhang. Communication complexities of xor functions. *arXiv preprint*, 2008. arXiv:0808.1762.

39 Hing Yin Tsang, Chung Hoi Wong, Ning Xie, and Shengyu Zhang. Fourier sparsity, spectral norm, and the log-rank conjecture. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 658–667, 2013. doi:10.1109/FOCS.2013.76.

40 Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *Proceedings of the eleventh annual ACM symposium on Theory of computing*, pages 209–213, 1979.

41 Huacheng Yu. Strong XOR lemma for communication with bounded rounds : (extended abstract). In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS*, pages 1186–1192. IEEE, 2022.

42 Shengyu Zhang. Efficient quantum protocols for xor functions. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1878–1885. SIAM, 2014.

New Sampling Lower Bounds via the Separator

Emanuele Viola
Khoury College of Computer Sciences, Northeastern University, Boston, MA, USA

Abstract

Suppose that a target distribution can be approximately sampled by a low-depth decision tree, or more generally by an efficient cell-probe algorithm. It is shown to be possible to restrict the input to the sampler so that its output distribution is still not too far from the target distribution, and at the same time many output coordinates are almost pairwise independent.

This new tool is then used to obtain several new sampling lower bounds and separations, including a separation between AC0 and low-depth decision trees, and a hierarchy theorem for sampling. It is also used to obtain a new proof of the Patrascu-Viola data-structure lower bound for Rank, thereby unifying sampling and data-structure lower bounds.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases Sampling, data structures, lower bounds, cell probe, decision forest, AC0, rank, predecessor

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.26

Related Version *Full Version*:
`https://eccc.weizmann.ac.il/report/2023/073/revision/3/download`

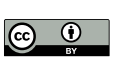
Funding *Emanuele Viola*: Supported by NSF CCF award 2114116.

Acknowledgements We thank the anonymous reviewers for detailed and helpful feedback.

1 Introduction and our results

Obtaining *computational lower bounds* is a fundamental agenda in theoretical computer science, see for example the textbooks [25, 3]. One of the most famous lower bounds is the AC0 lower bound for computing the parity function, which separates small AC0 circuits from models that can compute parity. Another direction that has received much attention is the relationship between AC0 and *low-depth decision trees*. The simple Or function on n bits requires decision trees of depth n to be computed exactly, but the picture is more subtle and useful when we consider *average-case* computation, that is we allow errors on a small fraction of inputs. Indeed, *switching lemmas* [17, 1, 45, 22, 35, 34, 4, 24, 23] (see [44] for an exposition and discussion) can be interpreted as non-trivial simulations of small AC0 circuits by decision trees. On the other hand, functions such as *Tribes* (see, e.g., [30]), computable by a polynomial-size DNF circuits, require large-depth decision trees, even on average.

In this work we study lower bounds and separations in the setting of *sampling*. This is a challenging generalization of average-case complexity, where we seek to bound the resources required to sample approximately a target distribution, given random bits. The study of *sampling lower bounds* [39, 28, 41, 15, 5, 8, 40, 43, 9] has seen significant activity and progress in the last ten years; for a survey talk see [37]. This study has also had impact on other areas. For example, it has had an impact on breakthrough constructions of *two-source extractors*: the papers [10, 26, 12, 11, 7] build on models or results from the study of sampling lower bounds. Also, sampling lower bounds have been used to obtain *data-structure* lower bounds [39]. In fact, jumping ahead, this paper will further develop this connection to data structures.



38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 26; pp. 26:1–26:23

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


Sampling lower bounds for AC0, roughly corresponding to the classical result mentioned above that Parity is not in AC0, have been obtained in [28, 41, 5, 43]. All these lower bounds share common techniques. Interestingly, essentially no technique was known to obtain separations within AC0. The main goal and motivation for this paper is thus to develop new techniques for sampling lower bounds, and apply them to obtain separations within AC0, in particular separating decision-tree from AC0 samplers and obtaining a hierarchy theorem (see Corollary 9). In addition, the new technique is used to "unify" sampling and data-structure lower bounds, that is, to obtain data-structure lower bounds as a consequence of sufficiently strong sampling lower bounds.

The model

The main computational model in this work is a generalization of the decision-tree model known as the *cell-probe* model [46]. Here the input is divided into *words* (a.k.a. *cells)* of w bits, the output is a tuple of *queries*, and each query can be computed by making q probes into the input, adaptively. This model is extensively studied in algorithms, where w corresponds to the *register size* and q to *time*. We note that for $w = 1$ each output query is computed by a *decision tree* of depth q. For larger w each query is also computed by a tree but each internal node probes a word and has 2^w children. Because the output is a tuple of queries, we have several trees of depth q, one for each output query, and so we refer to the algorithm as to a depth-q *forest*. We place no restriction on the number of input words, which we indicate with $\mathbb{N}$. But for concreteness one can replace $\mathbb{N}$ with any large enough integer – as we do later in the proofs. We summarize the model and its key parameters:

▶ **Definition 1.** *We say that $f : W^{\mathbb{N}} \to \Sigma^m$ is a depth-q* forest *with word size w and output alphabet Σ if $W = \{0,1\}^w$ and $f = (f_1, f_2, \ldots, f_m)$ where each $f_i : W^{\mathbb{N}} \to \Sigma$ is a depth-q decision tree where the variables are over W, each internal node has $|W|$ children, and the leaves are labeled with elements from Σ.*

The main goal of this paper is to show that a target distribution S over Σ^m is hard to sample by a low-depth forest. We measure the distance between distributions X and Y over D using *statistical (*a.k.a. *total variation, L_1) distance*

$$\Delta(X, Y) := \max_{T \subseteq D} |\mathbb{P}[X \in T] - \mathbb{P}[Y \in T]|.$$

So the lower-bound goal is to show $\Delta(f(U_{W^{\mathbb{N}}}), S)$ is large for any low-depth forest f, where $U_{W^{\mathbb{N}}}$ is the uniform distribution over $W^{\mathbb{N}}$. In general for a set H we write U_H for the uniform distribution over H, and simply U when H is clear from the context.

Previous sampling lower bounds

Before this work, essentially the only sampling lower bounds in the cell-probe model, or even in the decision-tree model, were those that followed from the sampling lower bounds for AC0 circuits [28, 41, 5, 43] – using the fact that a depth-q tree can be written as a DNF with width qw. This was unsatisfactory for several reasons. First, such results obviously cannot be used to prove separations within AC0. By contrast, this paper obtains such separations. Second, the AC0 lower bounds only hold for sampling *pseudorandom objects*, such as extractors or error-correcting codes. By contrast, the lower bounds in this paper apply to fundamental data-structure problems that do not have pseudorandom properties, and this allows us to unify sampling and data-structure lower bounds.

1.1 Our results

In this work we prove new sampling lower bounds and use them to derive a number of new separations. We emphasize that our results are new even for decision trees, corresponding to word size $w = 1$. However, we obtain stronger results by considering larger word size. Similarly, the lower bounds we prove were not known even for statistical distance $\Omega(1)$. But in fact we prove stronger bounds, where the statistical distance is exponentially close to 1. Via technically simple connections, the first of which was pointed out in [39], this generality enables several applications discussed below. In particular, jumping ahead, it will allow us to unify sampling and data-structure lower bounds. Indeed, our lower bounds hold for fundamental problems in data structures.

First we obtain a sampling lower bound for the distribution $\mathrm{Rank}(U_{\{0,1\}^m})$ where Rank is defined next.

▶ **Definition 2.** *For $x \in \{0,1\}^m$ define Rank(x) as the string $y \in \{0,1,\ldots,m\}^m$ where $y_i := \sum_{j \le i} x_j$ is the rank of i.*

▶ **Example 3.** $\mathrm{Rank}(0,1,0,1) = (0,1,1,2)$.

▶ **Theorem 4.** *Let $f : W^{\mathbb{N}} \to \{0,1,\ldots,m\}^m$ be a depth-q forest with word size $w \ge \log m$. Then $\Delta(f(U_{W^{\mathbb{N}}}), \mathit{Rank}(U_{\{0,1\}^m})) \ge 1 - 2 \cdot 2^{-m/w^{O(q)}}$.*

Throughout this paper, the notation $O(.)$ and $\Omega(.)$ denotes absolute constants.

It follows from [47], which builds on [31], that this bound is tight. In particular, we can sample $\mathrm{Rank}(U)$ with depth $q = O(\log m)/\log w$ and constant statistical distance

This result can also be interpreted as a negative result for sampling *random walks on graphs.* Consider the graph G over $\{0,1,\ldots,m\}$ where the neighbors of Node i are $\{i, i+1\}$. Note that $\mathrm{Rank}(U)$ is the sequence of nodes visited during the walk with edge choices U. Theorem 4 proves a lower bound for sampling this random walk. Note that as the graph is fixed, this result applies even if the algorithm depends on the graph.

Next we consider the *predecessor* problem.

▶ **Definition 5.** *For $x \in \{0,1\}^m$ define Pred(x) as the string $y \in \{0,1,\ldots,m\}^m$ where $y_i := \max\{j : j \le i \text{ and } x_j = 1\}$ is the predecessor of i. (Say $y_i = 0$ if there is no $j \le i$ with $x_j = 1$.)*

Unlike Rank, it turns out that Pred can be sampled efficiently. Specifically, there is a depth-$O(1)$ forest sampling $\mathrm{Pred}(U)$ with statistical distance $1/\mathit{poly}(m)$. This is just because the predecessor of i can be computed by inspecting the bit positions from $i - q \log n$ to i of x, except with error probability $1/n^q$.

Loosely inspired by works in data-structure lower bounds [32], we prove a lower bound for $\mathrm{Pred}(X)$ under a distribution X which is tailored for the applications below. This lower bound is really a lower bound for a "direct-product" version of Pred, where r instances have to be solved simultaneously. In fact, the bound holds even for the *colored* version, where items have colors and we just need to return the color of the predecessor. Next we define this problem and state our results for it.

▶ **Definition 6.** *For an $r \times m$ matrix M with entries in $\{-, \hat{O}, \hat{*}\}$ we define the $r \times m* Colored-Multi-Predecessor *matrix CMPred(M) with entries in $\{\hat{O}, \hat{*}, O, *\}$ as follows. For any i, j we define CMPred$(M)_{i,j}$ to be:*

- *$M_{i,j}$ if $M_{i,j} \ne -$,*
- *$*$ if the predecessor of j on row i is $\hat{*}$ (that is, there is $j' < j$ such that $M_{i,j'} = \hat{*}$ and for every k such that $j' < k < j$ we have $M_{i,k} = -$), and*
- *O otherwise.*

The distribution Π *on* $r \times m$ *matrices is defined as follows, for* m *divisible by* w^r*. Divide row* $i = 1, 2, \ldots, r$ *in consecutive blocks of* w^i *elements. For each block, pick a uniform element, and assign to it a uniform element from* $\{\hat{O}, \hat{*}\}$*. All the other elements are set to* $-$*.*

▶ **Example 7.** $\mathrm{CMPred}\left(\begin{bmatrix} - & \hat{*} & - & \hat{O} \\ - & - & \hat{*} & - \end{bmatrix}\right) = \begin{bmatrix} \mathrm{O} & \hat{*} & * & \hat{O} \\ \mathrm{O} & \mathrm{O} & \hat{*} & * \end{bmatrix}$.

Working with the alphabet $\{\hat{O}, \hat{*}, \mathrm{O}, *\}$ allows us to reconstruct M from $\mathrm{CMPred}(M)$, slightly simplifying the argument.

▶ **Theorem 8.** *There exists a constant* c *such that for* $r = cq$ *the following holds.*
Let $f : W^{\mathbb{N}} \to (\{\hat{O}, \hat{*}, O, *\}^r)^m$ *be a depth-*q *forest with word size* $w \geq \log m$*.*
Let Π *be an* $r \times m$ *random matrix as in Definition 6.*
Then $\Delta(f(U_{W^{\mathbb{N}}}), CMP_{RED}(\Pi)) \geq 1 - 2 \cdot 2^{-m/w^{O(q)}}$*.*

Motivation for studying CMPred(Π): New separations

The problem $\mathrm{CMPred}(\Pi)$ is designed to be easy to sample with a little more resources than we prove lower bounds for. Thus the theorem gives two separations. First, we obtain a *probe-hierarchy* for sampling: for any q there is an explicit problem that can be sampled exactly with $O(q)$ probes, but only very poorly with q. Second, the same problem can be also sampled by an explicit, polynomial-size DNF, thus giving a separation between sampling by cell probes and DNFs. Such results were not known even for word size $w = 1$, statistical distance 0.01 rather than close to 1, and AC0 instead of DNF.

Proving hierarchies and separations among various restricted computational models is a main research agenda of theoretical computer science. We consider them in the context of sampling. For example, it is a classical result that small DNF circuits can compute functions that require decision trees of large depth, even on average. Our results strengthen this separation substantially.

▶ **Corollary 9.** *For every* m, q, w *such that* $w \geq \log m$ *the following holds.*
There exists a distribution $S \subseteq \left(\{0,1\}^{O(q)}\right)^m$ *such that for any depth-*q *forest* f *with word size* w *we have* $\Delta(f(U_{W^{\mathbb{N}}}), S) \geq 1 - 2 \cdot 2^{-m/w^{O(q)}}$*. But* S *can be sampled (with distance* 0*) by both*

(1) *An explicit depth-*$O(q)$ *forest with word size* w*; and*

(2) *An explicit* $\mathrm{poly}(m)$*-size DNF.*

The distribution in this corollary is $\mathrm{CMPred}(\Pi)$ with $r = O(q)$. To sample it, we can identify row i of Π with a string of $\log_2(2 \cdot w^i)^{m/w^i}$ bits, indicating the choice of color and element ($2 \cdot w^i \leq O(m)$ possibilities) for each of the m/w^i blocks. Color i of query j can be computed from these bits probing $O(1)$ words. Repeating this for $i = 1, 2, \ldots, r$ gives (1) in the corollary. (2) is similar.

Previous attempts to establish a separation between forests and AC0 circuits resulted in (i) Theorem 1.4 in [39] which applies to *randomness-efficient* samplers, achieves constant statistical distance, and has $w = 1$, and (ii) Theorem 3 in [43] which applies to *non-adaptive* samplers. It is an open question whether Rank can be sampled by polynomial-size AC0 circuits. Recent results [47] building on [31] imply that it can be sampled with $O(\log m)$ probes and constant statistical distance, which gives *quasi*-polynomial size AC0.

Data structures

A *(static) data-structure problem* is a map $f : \{0,1\}^n \to \Sigma^m$, where m queries over alphabet Σ are to be answered about n bits of data. A *data-structure* with word size w for this problem are two functions $g : \{0,1\}^n \to \{0,1\}^{n+r}, h : \{0,1\}^{n+r} \to \Sigma^m$ where g is arbitrary and h is a depth-q forest with word size w such that $f = h \circ g$. That is, we seek to store the n bits of data into $n + r$ bits so that the queries can be computed fast. Note that the $n + r$ bits are divided in words of w bits. We call r the *redundancy* of the data structure, and we focus on the *succinct* regime $r = o(n)$. Many papers are devoted to proving lower bounds in this regime, including [18, 38, 19, 33, 27]; and it is shown in [42] that improving on the long-standing bounds in [18] would yield new circuit lower bounds.

The paper [39] pointed out a technically simple connection between samplers and data-structures: any data structure can be used to sample the distribution $f(U)$ by a depth-q forest with statistical distance $1 - 2^{-r}$. Simply fill the $n + r$ bits uniformly and run the query algorithms. A data structure is *equivalent* to the special case of samplers *which just use $n + r$ input bits*. But samplers can use *any* number of input bits, and many samplers in the literature do use $(1 + \Omega(1))n$ input bits, for example to sample noise vectors, subsets, or permutations, cf. [39].

Hence, the sampling lower bounds above imply data-structure lower bounds. Theorem 4 gives a new proof of the data-structure lower bound for Rank from [33], which was recently shown to be tight in [47], building on [31]. This new proof shows that the lower bound applies even to samplers. Informally, this suggests that the "reason" why the lower bound for Rank holds is not that the input is "compressed," *but rather that low-depth forests simply cannot generate the type of dependencies in Rank, regardless of their input.*

The program of proving data-structure lower bounds via sampling was suggested a decade ago [39], but the only previous cell-probe lower bound obtained this way is for error-correcting codes and follows from the AC0 lower bounds [28, 6]. This paper shows that this program is feasible for problems such as Rank.

Similarly, we obtain a data-structure lower bound for CMPred. Also, the sampling hierarchy in Corollary 9 translates to a *data-structure hierarchy*. Hierarchies in data structures have been considered since the 90's. [29] gives a *non-explicit* problem where decreasing the *redundancy* by one bit makes the probe time jump from constant to linear. We give explicit problems where increasing the probe time q by a constant factor makes the redundancy shrink from almost linear to zero. Previous bounds such as [33] imply such a result for q about $\log m$. We achieve a broader range including $q = O(1)$. To the best of our knowledge, such a result does not appear in the literature.

▶ **Corollary 10** (Data-structure hierarchy). *For every q and m there exists an explicit function $f : \{0,1\}^m \to (\{0,1\}^{O(q)})^m$ which has a data structure with word size w, redundancy zero, and making $O(q)$ probes, but such that any data-structure with word size w making q probes requires redundancy $r \geq m/w^{O(q)}$.*

The sampling viewpoint is not essential for the data-structure lower bound for CMPred or for Corollary 10: just like Rank, they can be proved without referring to sampling.

Communication protocols

Above we considered one application of proving sampling lower bounds with large error, close to 1, namely data-structure lower bounds. The large statistical distance corresponded to redundancy. In this paper we put forth another application to *communication protocols.*

Here the large statistical distance corresponds to communication. We consider the following communication protocols: we associate to each output query a *party*. In addition to probing input cells as before, the parties also communicate. While sampling protocols have been studied before, see e.g. [2, 20], our setting where each party is charged to access input cells does not seem to have been studied before. Next we define it and then state our result. The result is an easy corollary and our main goal here is to give another interpretation of sampling lower bounds with large statistical distance.

▶ **Definition 11.** *A* sampler protocol *over Σ^m with word size w, q probes, and c total communication is a communication protocol among m parties. The parties share a public random string of cells of w bits each. At each point in time, the protocol specifies which Party i is to go next. Party i can either probe a cell, broadcast communication, or output a value in Σ and stop. The action of Party i at time t depends only on the values of the cells Party i probed in previous times, and on the communication transcript. The output of the protocol is the tuple of elements output by the parties. The number of probes made by each party is at most q, while c is a bound on total communication.*

To get a sense of the parameters, consider for example Rank(U). We can sample it with no error with 1 probe and communication m/w (each party probes a different cell and broadcasts it – then the players sample exactly). And as we remarked earlier, it can also be sampled with $o(\log m)$ probes and no communication, up to constant error. We obtain the following lower bound, which interpolates between these two extremes.

▶ **Corollary 12.** *Let Π be a sampler protocol with word size w, q probes, and communication c whose output has statistical distance δ from Rank(U). Then $c \geq m/w^{O(q)} + \log(1-\delta) - O(1)$.*

2 Techniques

Our results rely on a new proof technique which we call the *cell-probe sampling separator*, or just *separator* for brevity, and which is a main technical contribution of this work. Roughly speaking, this separator result says that if $f : W^{\mathbb{N}} \to \Sigma^m$ is a low-depth forest whose output distribution is close to a target distribution S over Σ^m, then we can restrict the input space to a subset $D \subseteq W^{\mathbb{N}}$ such that when the input to f comes from D, many trees in the output distribution $f(D)$ are *nearly pairwise independent*, and at the same time the output distribution is still not very far from the target S. This latter feature will be formalized by requiring that $f(D)$ is supported on a subset of the support of S, and has entropy almost equal to that of S.

A critical feature of the separator is that the number of trees that are guaranteed to be almost pairwise independent in $f(D)$ is much larger than the entropy gap between $f(D)$ and S. Formally, for a sufficiently spaced-out increasing sequence of integers $t_0, t_1, \ldots$, the separator will guarantee that for some value k there is a set $D_k = D$ and t_k trees that are nearly pairwise independent over $f(D_k)$, while the entropy gap is only about t_{k-1}. (The separator can also guarantee almost ℓ-wise independence for $\ell > 2$, but we only need $\ell = 2$ in our results.)

After some definitions we state the separator.

▶ **Definition 13** (Almost pairwise independence)**.** *Jointly distributed random variables X, Y are ϵ-independent if (X, Y) is ϵ-close in statistical distance to (X, Y') where Y' has the same distribution of Y and is independent from X.*

The *min-entropy* $H_\infty(X)$ of a random variable X is $\min_a \log_2(1/\mathbb{P}[X = a])$.

Notation

To avoid clutter in the more technical exposition of the results, we adopt the convention that for a set S we also denote by S the uniform distribution U_S over S. The meaning will be clear from the context. For example, we shall simply write $\Delta(f(W^{\mathbb{N}}), S)$ for $\Delta(f(U_{W^{\mathbb{N}}}), U_S)$.

▶ **Theorem 14** (Cell-probe sampling separator). *There exists an integer $c \geq 1$ such that the following holds:*

Hypothesis: *Let $f : W^{\mathbb{N}} \to \Sigma^m$ be a depth-q forest with word size w. Let $\alpha \leq 1/c$. Let $t_0, t_1, \ldots$ be a sequence of integers with $t_i \geq t_{i-1} \cdot cqw/\alpha$ for every i. Let $S \subseteq \Sigma^m$ be a set and suppose that $\Delta(f(W^{\mathbb{N}}), S) \leq 1 - 2^{-t_0}$ where $2^{-t_0} \geq \sqrt{8/|S|}$.*

Conclusion: *There exists $k, 1 \leq k \leq O(q/\alpha)$, $D_k \subseteq W^{\mathbb{N}}$, and t_k indices $T \subseteq [m]$ such that:*

- **(0)** $H_\infty(f(D_k)) \geq H_\infty(S) - t_{k-1} \cdot O(qw/\alpha)^2$;
- **(1)** *The support of $f(D_k)$ is contained in S;*
- **(2)** *For every $i, j \in T$ the random variables $(f_i(D_k), f_j(D_k))$ are $O(\alpha)$-independent.*

For example, we can set $t_i = m/w^{a(q/\alpha)-bi}$ which for suitable a, b and $q, w \leq \log m$ satisfies the hypothesis.

Proof sketch of the separator

First we need to understand what it means for $\Delta(f(W^{\mathbb{N}}), S)$ to be at most $1 - \epsilon$. One special case in which this happens is if the distribution $f(W^{\mathbb{N}})$ is equal to the uniform distribution over S with probability ϵ, and otherwise is say a fixed value. Our first Lemma 15 shows that this special case, more or less, is in fact the general case. Specifically, we can condition the input to f on an event of probability about ϵ so that, if D is the resulting set of inputs, $f(D)$ is supported inside of S, and the entropy of $f(D)$ is almost maximum.

At this point we forget S and our goal is to further restrict D so that we have many pairwise independent queries, and at the same time we do not lose too much in entropy.

First we apply the so-called *fixed-set lemma* from [21]. This lemma shows that it is possible to moderately restrict D to a subset $D_1 \subseteq D$ so that no low-depth tree can distinguish D_1 from a *product distribution* R.

At this point, we ask if in $f(D_1)$ there are many (t_1) queries (a.k.a. trees) such that any two of them *intersect probes* with probability $\leq \alpha$. Here we say that two trees intersect probes if there exists i such that both trees probe word i.

If the answer is positive: we argue that we are done. Let us explain why that is the case. First, we can write the probability that two queries probe the same word as a low–depth tree. By the fixed-set lemma, this probability is the same over D_1 and over the product distribution R. However, over a product distribution two queries are independent unless they probe the same word, hence over R two queries are α-independent, and it follows that the same is true over D_1.

We note that our use of the fixed-set lemma is different from [21]. In the latter paper it was used to argue that the input to a tree looks uniform. By contrast, we use it to establish *pairwise independence* among trees, and critically we use it to bound the probability that two trees probe the same word.

If the answer is negative: In this case, by a version of simple "covering arguments" which are widespread since at least the sunflower lemma [16], there is a small set T of trees such that any other tree intersects probes with some tree in the set with probability $\geq \alpha$. Now the idea is to fix the probes of the trees in T to obtain a new input D_2 over which the total expected probe time is reduced. Then again we can apply the fixed-set lemma, and iterate the argument.

This fixing of the probes in T is inspired by a fixing that occurs in the data-structure lower bound for Rank [33]. However, we note that our argument is different. The proof in [33] selects trees in a structured way, with a precise sequence of "gaps." By contrast, our selection comes from the covering argument and is, at this stage, unstructured: we simply count queries. More generally, the proof in [33] proceeds by an *encoding argument,* as is typical in data-structure lower bounds, which is tailored to the problem at hand. The separator avoids that and allows us to establish an intrinsic property of efficient samplers and data structures.

This concludes the informal overview. The formal proof is in Section 3.

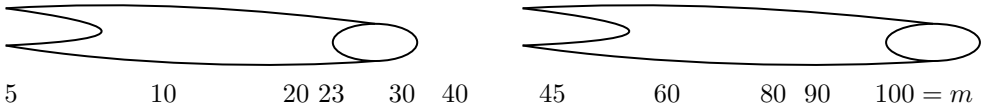
Comparison with switching lemmas

Switching lemmas [17, 1, 45, 22, 35, 34, 24, 23] show that small-width DNF simplify under random restrictions. Since a depth-q decision tree over alphabet $\{0,1\}^w$ can be written as a DNF with width qw, switching lemmas apply to our model too. A main difference between switching lemmas and our separator is that the former restrict the input space *aggressively*, for example fixing all but a constant fraction of the input bits, while our separator restricts the input *moderately*, for example fixing a small, sub-linear number of input words. This distinction is critical, since our problems are easy for DNF.

Comets

Having established the separator, there remains to use it to prove lower bounds. Our approach is based on a combinatorial object that we call *comet*. A d-comet is a triple of integers where the first two, the comet's *tail,* are $\geq d$ times farther apart than the last two, the comet's *head.* We can imagine the sun at position ∞: Blown by solar winds, comet tails point away from the sun.

The following example shows two 4-comets: $(5, 20, 23)$ and $(40, 80, 90)$:



We show in Section 4 that any large set of integers contains many non-overlapping d-comets, for large enough d. In the proofs of the sampling lower bounds (Sections 6 and 7), this result is applied to the t_k trees given by the separator theorem. Because the entropy gap of D_k and S is, as remarked earlier, much less than t_k, it follows that we can find among the trees a comet that is "random," that is, roughly, the query outputs have a lot of entropy. However, we prove that this is impossible, because over $f(D_k)$ the queries are nearly independent, but we show that they are not so in (any restriction of) the target distribution. Here is where we use the geometry of comets: the long tail will impose correlations on the head of the comet. The way this is formalized depends on the problem. For CMPred, we can find blocks in Π which are just a little longer than comets' heads, guaranteeing correlations between the queries in the head. For Rank the argument is a little more complex because a query depends on the entire prefix, so we shall need to guarantee that the bits corresponding to the comet have sufficiently high entropy even conditioned on the prefix.

2.1 Conclusion and open problems

This paper adds new tools to the study of sampling lower bounds, especially the separator theorem. Using them, a number of new lower bounds and separations are obtained. Several natural questions remain open. One is separating adaptive from non-adaptive samplers.

Another is proving cell-probe lower bounds for sampling other distributions, such as *permutations, cf.* [43]. The parameters of the separator do not seem strong enough for the latter goal; in brief, one would need to set α too small.

These new tools can also be used to generalize previous data-structure lower bounds, such as the one for Rank [33], to *sampling* lower bounds. This additional information could be useful in understanding which techniques are suitable for further progress. For example, Membership [29, 36] is a long-standing problem in data structures which asks to store a subset of $[m]$ of size say $m/4$ so that membership queries can be computed fast. It is interesting to note that the corresponding sampling problem is easy: we can sample somewhat well the uniform distribution over these subsets in time $O(1)$ *using 2m input bits.* (Simply taking the And of adjacent pairs of bits will generate exactly the uniform distribution over m iid variables each coming up 1 with probability $1/4$; and this distribution has statistical distance only $1 - \Omega(1/\sqrt{m})$ from the subsets.) Hence, unlike Rank, a strong lower bound for Membership must exploit that the input length is bounded, and this might indicate why this problem is harder than Rank.

3 Proof of the separator Theorem 14

First we need to understand what it means to have slightly non-trivial statistical distance. Let P be a distribution over Σ^m. One way in which P can have statistical distance $\leq 1-\epsilon$ from S is if P is distributed like S with probability ϵ, and it is say fixed with probability $1-\epsilon$. In this case, P has actually very high entropy ($\log_2 |S|$) conditioned on an event of probability ϵ. The next lemma shows that this in fact always happens.

▶ **Lemma 15.** *Let P be a distribution over Σ^m and let $S \subseteq \Sigma^m$. Suppose that $\Delta(P,S) \leq 1-\epsilon$, where $\epsilon \geq \sqrt{8/|S|}$. Then there is a subset $S_0 \subseteq S$ of probability $\mathbb{P}_P[S_0] = \Omega(\epsilon)$ such that the distribution P conditioned on $P \in S_0$ has min-entropy $\geq H_\infty(S) - O(\log 1/\epsilon)$.*

Proof. We also write P for the random variable distributed according to P. Collect all the elements of S in increasing order of mass according to P until right before collecting cumulative mass $\epsilon/2$. Note we don't collect all of S, for else $\mathbb{P}[P \in S] \leq \epsilon/2$ and $\Delta(P,S) \geq 1-\epsilon/2$, contradicting the hypothesis.

Let β be the mass of the next element of S. Let S_0 be the collected elements, S_1 the rest of S, and T the complement of S. By definition, $\mathbb{P}[P \in S_0] < \epsilon/2$, and so $\mathbb{P}[P \in S_1 \bigcup T] \geq 1-\epsilon/2$. Also for every $x \in S_1$ we have $\mathbb{P}[P = x] \geq \beta$ and so $|S_1| \leq \beta^{-1}$. Combining these bounds with the assumption we have

$$1 - \epsilon \geq \Delta(P,S) \geq \mathbb{P}[P \in S_1 \cup T] - \frac{|S_1|}{|S|} \geq 1 - \epsilon/2 - \frac{\beta^{-1}}{|S|}$$

and so $\beta \leq 2/(\epsilon|S|)$.

Because we did not include in S_0 an element of mass β, and we only stop when we reach $\epsilon/2$, the mass of S_0 is $\geq \epsilon/2 - \beta \geq \epsilon/2 - 2/(\epsilon|S|)$. If $\epsilon \geq \sqrt{8/|S|}$ this mass is at least $\epsilon/4$.

For any $x \in S_0$ using the above bound on β we obtain

$$\mathbb{P}[P = x | P \in S_0] = \frac{\mathbb{P}[P=x]}{\mathbb{P}[P \in S_0]} \leq \frac{\beta}{\epsilon/4} \leq \frac{8}{\epsilon^2|S|},$$

as desired. ◀

The above lemma allows us to "forget" about S and focus on f. We need to show that we can restrict the input to a large subset such that many output trees are nearly independent. This is the content of the following theorem. To avoid having to think about infinite sets, in

the remainder of the proof we set the input to the sampler to W^s for an integer s. This is without loss of generality, since obviously any forest of fixed depth can only access a finite number of input words.

We define the (entropy) *loss* of a subset $D' \subseteq D$ to be $\log_2(|D|/|D'|)$. So if D' contains half the elements of D the loss is one.

▶ **Theorem 16.** *There exists an integer $c \geq 1$ such that the following holds:*

Hypothesis: *Let $f : W^s \to \Sigma^m$ be a depth-q forest with word size w. Let $\alpha \leq 1/c$. Let $t_0, t_1, \ldots$ be a sequence of integers with $t_i \geq t_{i-1} \cdot cqw/\alpha$ for every i. Let $D \subseteq W^s$ be a set with loss $\leq t_0$.*

Conclusion: *There exists $k, 1 \leq k \leq O(q/\alpha)$, $D_k \subseteq D$, and t_k indices $T \subseteq [m]$ such that:*

(1) *The loss of $D_k \subseteq D$ is $\leq t_{k-1} \cdot (qw/\alpha)^2$;*

(2) *For every $i, j \in T$ the random variables $f_i(D_k), f_j(D_k)$ are $O(\alpha)$-independent.*

Let us first show how this gives the separator Theorem 14.

Proof of Theorem 14 from Theorem 16. We apply Lemma 15 to $P = f(U)$. Given $S_0 \subseteq S$ from the lemma, we let $D \subseteq W^s$ be the preimage of S_0 according to f. By the lemma, $|D|/|W|^s \geq \Omega(2^{-t_0})$, that is, the loss of $D \subseteq W^s$ is $t_0 + O(1)$. Moreover, $H_\infty(f(D)) \geq \log|S| - O(t_0)$.

We now apply Theorem 16 to this set D and the sequence $t_0 + O(1), t_1, t_2 \ldots$. We can adjust the constant c so that this satisfies the hypothesis. The theorem gives $D_k \subseteq D$ with loss $\leq t_{k-1} \cdot (qw/\alpha)^2$.

Observe that the support of $f(D_k)$ is contained in S, because the support of $f(D)$ is $S_0 \subseteq S$ and $D_k \subseteq D$.

To verify the bound on $H_\infty(f(D_k))$, note that

$$\mathbb{P}[f(D) = x] \geq \mathbb{P}[f(D_k) = x]|D_k|/|D|.$$

Taking inverses and then logs we obtain

$$\log(1/\mathbb{P}[f(D) = x]) \leq \log(1/\mathbb{P}[f(D_k) = x]) + \log(|D|/|D_k|).$$

The left-hand side is at least $H_\infty(f(D)) \geq \log|S| - O(t_0)$. While $\log(|D|/|D_k|) \leq t_{k-1} \cdot (qw/\alpha)^2$. Hence,

$$\log(1/\mathbb{P}[f(D_k) = x]) \geq \log|S| - O(t_0) - t_{k-1} \cdot (qw/\alpha)^2,$$

for any x. The result follows. ◀

3.1 Proof of Theorem 16

The main technical lemma is the following one, which is like Theorem 16 but the requirement of independence is replaced by others easier to work with.

▶ **Lemma 17.** *Theorem 16 holds if we replace (2) with:*

(2') *for every $i, j \in T$: the probability over D_k that $f_i(D_k)$ and $f_j(D_k)$ don't make all distinct probes is $\leq \alpha$, and*

(2") *there exists a product distribution R over words (that is, the words are independent) such that for every depth-$2q$ tree g, $g(D_k)$ and $g(R)$ are α-close.*

▶ **Lemma 18.** *(2') and (2") in Lemma 17 imply (2) in Theorem 16.*

Proof. Let $X = D_k$. Think of $(f_i(X), f_j(X))$ as the output of the tree g obtained by appending f_j to the leaves of f_i. Note that g makes $2q$ probes, possibly repeated. By (2"), there is a product distribution R such that $g(X)$ and $g(R)$ are α-close. Also, the probability that g repeats a probe over X is α-close to the probability that it repeats it over R. Here we use that this probability can be written as the probability that a tree d of depth $2q$ outputs 1, and that the output distributions of d over X and R are α-close. (Tree d is obtained from g by replacing any repeated probe along any path with a leaf 1, and any other leaf with 0.)

By this and (2') the probability that g repeats a probe over R is $\leq 2\alpha$. Because R is product, as long as probes are not repeated the output distribution does not change if we answer the first q probes with R^1 and the next q probes with R^2 where R^1, R^2 are iid copies of R. This shows that $(f_i(X), f_j(X))$ is $O(\alpha)$-close to $(f_i(R^1), f_j(R^2))$. Using again (2"), we can replace each R^i with X^i, where X^1, X^2 are iid copies of X. This gives that $(f_i(X), f_j(X))$ is $O(\alpha)$-close to $(f_i(X^1), f_j(X^2))$. Adjusting constants concludes the proof. ◀

3.2 Proof of Lemma 17

We shall be concerned with inputs in various subsets $X \subseteq D$. If an input word is constant for every $x \in X$ then it needs not be probed but can be "hardwired" in the trees. We shall assume that the trees are always simplified accordingly. We denote by $G(x, X)$ the total number of probes made by all trees on input x, where the trees are simplified with respect to X.

We use the following fixed-set lemma from [21]. We say that distributions X and Y are α-indistinguishable by depth-q trees if for any such tree t, the statistical distance between $t(X)$ and $t(Y)$ is $\leq \alpha$.

▶ **Lemma 19** ([21], Lemma 3.14.). *Let $B \subseteq W^s$ be a subset with loss $\leq b$, where $W = \{0,1\}^w$. There exists $B_1 \subseteq B$ and a product distribution R such that B_1 and R are α-indistinguishable by depth-$2q$ decision trees. Moreover, the loss of $B_1 \subseteq W^s$ is $\leq b \cdot O(wq/\alpha)$.*

For completeness we include the proof in Appendix A.

We begin by applying this lemma to D obtaining $D_1 \subseteq D$ with loss $t_0 \cdot O(wq/\alpha)$. This is the beginning of Iteration 1.

Our goal is to show that at the beginning of Iteration k there exists a subset $D_k \subseteq D$ enjoying the following properties:

(1) [in Theorem 16] the loss of D_k is $\leq t_{k-1} \cdot (qw/\alpha)^2$,

(2") [in Lemma 17] there exists a product distribution R over words such that for every depth-$2q$ tree g, $g(D_k)$ and $g(R)$ are α-close.

(3) $\max_{x \in D_k} G(x, D_k) \leq m(q - \alpha(k-1)/4)$.

Note all these hold at the beginning of Iteration 1.

In an iteration, collect as many trees as possible such that for any two of them, the probability over D_k that they intersect probes is $\leq \alpha$. If you have t_k, then (2') in Lemma 17 holds as well, concluding the proof.

Otherwise, you have a collection of t_k trees such that any other tree will intersect a probe with one of those t with probability $\geq \alpha$. We are going to use this to proceed to the next iteration, i.e., increase the value of k by 1. Because G is non-negative, Property (3) above implies that there can be at most $O(q/\alpha)$ iterations, as desired.

Write Y for the $\leq t_k q$ words probed by the t_k trees in D_k. This is done according to a canonical order, and is a valid definition because the first probe of a tree is fixed, the second is fixed once the answer to the first is, and so on.

Support size

Let $D_{k,y}$ be the inputs in D_k with $Y = y$. We have

$$\mathbb{E}_Y[|D_k|/|D_{k,Y}|] = \sum_y \mathbb{P}[Y = y]\frac{|D_k|}{|D_{k,y}|} = \sum_y 1 \leq |W|^{t_k q}.$$

By Markov's inequality $\mathbb{P}_Y[|D_k|/|D_{k,Y}| \geq M] \leq |W|^{t_k q}/M$. And so with probability $\geq 1 - |W|^{t_k q}/M$ over Y we have $|D_{k,Y}| \geq |D_k|/M$.

Intersection

For a tree f_i let $I_i(x)$ equal 1 if on input x tree f_i intersects probes with at least one of the t_k trees collected, and equal 0 otherwise. Note that for every input x and fixing y we have

$$G(x, D_{k,y}) \leq G(x, D_k) - \sum_{i\in[m]} I_i(x).$$

Because $\mathbb{P}_{x\in D_k}[I_i(x) = 1] \geq \alpha$ for every i, we have

$$\mathbb{E}_{x\in D_k}\left[\sum_{i\in[m]} I_i(x)\right] \geq \alpha m.$$

Because the inner sum is $\leq m$, by Markov's inequality we have that with probability $\geq \alpha/2$ over the choice of Y

$$\mathbb{E}_{x\in D_{k.Y}}\left[\sum_{i\in[m]} I_i(x)\right] \geq \alpha m/2,$$

and

$$\mathbb{E}_{x\in D_{k.Y}} G(x, D_{k,Y}) \leq \mathbb{E}_{x\in D_{k.Y}} G(x, D_k) - \alpha m/2.$$

Combining the arguments

Selecting $M = 2|W|^{t_k q}/\alpha$ above, and by a union bound, there is a value $\bar{y}$ so that

$$\mathbb{E}_{x\in D_{k,\bar{y}}}[G(x, D_{k,\bar{y}})] \leq \mathbb{E}_{x\in D_{k,\bar{y}}} G(x, D_k) - \alpha m/2;$$

and at the same time $|D_{k,\bar{y}}| \geq |D_k| \cdot \alpha|W|^{-t_k q}/2$. That is, we increase the loss by

$$\leq \log(1/\alpha) + wt_k q + 1.$$

Recall that the loss of D_k is $t_{k-1} \cdot (qw/\alpha)^2$.

Note that $D_{k,\bar{y}}$ is still uniform over its support, since it is D_k conditioned on a particular choice for $\leq t_k q$ words. Even though the words are chosen adaptively in D_k, once we condition on a particular value, their locations are fixed.

Reducing G for every input

By Markov's inequality,

$$\mathbb{P}_{x\in D_{k,\bar{y}}}[G(x,D_{k,\bar{y}}) \geq (\mathbb{E}_{x\in D_{k,\bar{y}}}G(x,D_k) - \alpha m/2)(1+\alpha/(4q))] \leq \frac{1}{1+\alpha/(4q)} \leq 1-\alpha/(8q).$$

Hence, for $\geq \alpha/(8q)$ fraction of the inputs x in $D_{k,\bar{y}}$ we have

$$\begin{aligned} G(x,D_{k,\bar{y}}) &\leq (\mathbb{E}_{x\in D_{k,\bar{y}}}G(x,D_k) - \alpha m/2)(1+\alpha/(4q)) \\ &\leq m(q-\alpha(k-1)/4)(1+\alpha/(4q)) - \alpha m/2 \\ &\leq m(q-\alpha k/4), \end{aligned}$$

using (3) in the second inequality. Let $D'_{k,\bar{y}}$ be the set of these inputs. The above gives the desired bound on $\max_{x\in D'_{k,\bar{y}}} G(x,D'_{k,\bar{y}})$, and note that the loss of $D'_{k,\bar{y}} \subseteq D_{k,\bar{y}}$ is $\leq \log(8q/\alpha)$.

Fixed-set lemma

Finally, we apply the fixed-set lemma to $D'_{k,\bar{y}}$ to obtain D_{k+1}; this gives (2"). This application multiplies the loss by $O(wq/\alpha)$, bringing the loss of $D_{k+1} \subseteq D$ to

$$O(wq/\alpha)\cdot O(t_{k-1}\cdot(qw/\alpha)^2 + \log(1/\alpha) + wt_kq + 1).$$

We need this loss to be at most $t_k\cdot(qw/\alpha)^2$. Dividing by wq/α we need to verify that

$$O(t_{k-1}\cdot(qw/\alpha)^2) + O(\log 1/\alpha) + O(wt_kq) + O(1) \leq t_k\cdot qw/\alpha.$$

We claim that each term on the left-hand side is at most one-fourth of the right-hand side. For the first term we use the hypothesis that $t_k \geq t_{k-1}\cdot cqw/\alpha$ for a large enough c, and for the third we use that $\alpha \leq 1/c$ and pick c large enough. This gives (1).

Because $D_{k+1} \subseteq D'_{k,\bar{y}}$, the bound on G still holds for D_{k+1}, and this gives (3).

4 Comets

In this section we define comets and prove a comet-finding lemma which will be used in our sampling lower bound.

▶ **Definition 20.** *A d-comet is a triple of indices (i,j,k) from $[m]$ with $i<j<k$ such that $j-i \geq d(k-j)$. We call (j,k) the head and (i,j) the tail. The head length is $k-j$. A set of comets $\{(i_h,j_h,k_h)\}_h$ is disjoint if the intervals $[i_h,k_h]$ are disjoint.*

▶ **Lemma 21** (Comet-finding). *A subset of $\{1,2,\ldots,m\}$ of size m/ℓ^b contains $\geq m/\ell^{b+c+O(1)}$ disjoint ℓ^c-comets where the head lengths are all in $[\ell^h,\ell^{h+1}]$ for some integer $h \leq b+c+O(1)$, for any $m,b \leq \ell, c \leq \ell$, and $\ell \geq \log m$.*

Proof. Let $d=\ell^c$. First we claim that any subset of size $n := d\log m+2$ contains a d-comet. Let the elements in the set be $a_1,a_2,\ldots$ in increasing order. If (a_1,a_2,a_3) is not a d-comet then $a_3-a_2 > (a_2-a_1)/d$, and so $a_3-a_1 = a_3-a_2+a_2-a_1 \geq (a_2-a_1)(1+1/d)$. Then again if (a_1,a_3,a_4) is not a d-comet we have $a_4-a_3 \geq (a_3-a_1)/d$ and so $a_4-a_1 \geq (a_3-a_1)(1+1/d) \geq (a_2-a_1)(1+1/d)^2$. If we continue this way $n-2$ times, we obtain $a_n \geq (1+1/d)^{n-2} > m$, which is a contradiction.

Now divide the $t := m/\ell^b$ elements of the given set into consecutive blocks of size n. By the previous paragraph, each block contains a comet. Hence we have $\geq t/n - 1$ disjoint d-comets.

At least half of these comets have heads of length $\leq O(mn/t) = \ell^{b+c+O(1)}$, otherwise half the comets have heads longer than that, and we run out of space. Let C_i be the subset of these comets whose head length is in $[\ell^i, \ell^{i+1})$. We only need to consider $i \leq b + c + O(1)$. Hence, there exists $i = h$ and

$$\Omega\left(\frac{t}{n}\right)\frac{1}{b+c+O(1)} \geq \frac{m}{\ell^{b+c+O(1)}}$$

disjoint comets with head lengths in $[\ell^h, \ell^{h+1})$, using that both b and c are $\leq \ell$. ◀

5 A lemma about entropy

In this section we quickly recall a basic result about entropy which will be used in our sampling lower bounds. The *entropy* H of a random variable X is defined as $H(X) := \sum_x \Pr[X = x] \cdot \lg(1/\Pr[X = x])$. The conditional entropy $H(X|Y) := E_{y\in Y} H(X|Y = y)$ (cf. Chapter 2 in [13]).

▶ **Lemma 22.** *Let $Z = (Z_1, \ldots, Z_k)$ where Z_i is supported over a set S_i, and let $\sum_i \log|S_i| = M$. Suppose $H(Z) \geq M - a$. There is a set $G \subseteq [k]$ of size $|G| \geq k - a/\epsilon$ such that for any $i \in G$ we have*

$$H(Z_i|Z_1Z_2\ldots Z_{i-1}) \geq \log|S_i| - \epsilon.$$

In particular, Z_i is $4\sqrt{\epsilon}$ close to uniform over S_i.

Proof. By the chain rule for entropy ([13], Equation 2.21)

$$\sum_{i\leq k}(\log|S_i| - H(Z_i|Z_1Z_2\ldots Z_{i-1})) \leq a.$$

Applying Markov inequality to the non-negative random variable $\log|S_i| - H(Z_i|Z_1Z_2\ldots Z_{i-1})$ (for random $i \in [k]$), we have

$$\mathbb{P}_{i\in[k]}[\log|S_i| - H(Z_i|Z_1Z_2\ldots Z_{i-1}) \geq \epsilon] \leq a/(k\cdot\epsilon),$$

yielding the desired G.

The "in particular" part holds because conditioning reduces entropy: $H(Z_i) \geq H(Z_i|Z_1Z_2\ldots Z_{i-1})$ *(*[13], Equations 2.60 and 2.92) and then applying Pinsker's inequality ([14], Chapter 3; Exercise 17). ◀

6 Proof of Theorem 8

We can assume that $q \leq w$, for else the statistical bound is trivial and the theorem is true. We apply Theorem 14 with $\alpha = 1/10$ and the sequence

$$t_i := m/w^{c_0(q/\alpha)-c_1 i},$$

for constants c_0, c_1 to be set later. For large enough c_1 this satisfies the hypothesis of the theorem that $t_i \geq t_{i-1}\cdot cqw/\alpha$. We also need to show that $2^{-t_0} \geq \sqrt{8/|S|}$, where $|S|$ is the number of matrices Π in the definition of CMPred. This is true since $|S| \geq 2^{\Omega(m/w)}$.

Let k, D_k, and t_k be as provided by the theorem. Recall that

$$H_\infty(f(D_k)) \geq H(\Pi) - t_{k-1}\cdot O(qw/\alpha)^2.$$

Finding comets among trees

The theorem provides $t := t_k = m/w^{c_0(q/\alpha)-c_1 k}$ trees. Applying the Comet-Finding Lemma 21 with $c = 3$ and $\ell = w \geq \log m$ gives a set of

$$t' := m/w^{c_0(q/\alpha)-c_1 k+O(1)}$$

disjoint w^3-comets, where the head lengths are in $[w^h, w^{h+1})$ for some $h \leq c_0(q/\alpha) + O(1)$. Note that to apply the lemma we need that $c_0(q/\alpha) \leq w$. This is guaranteed since $w \geq \log m$ and $q = O(\log m)/\log\log m$ for else the conclusion of the theorem holds trivially.

We shall get a contradiction looking at the row of the matrix corresponding to blocks of length w^{h+2}; the other rows can be ignored.

A random comet

To each of the above t' comets we associate three *relevant*, consecutive blocks. Of these, the middle block is the first block that intersects the head of the comet. Note that:

- the relevant blocks cover the head of the comet, since the blocks have length w^{h+2} while the head has length $\leq w^{h+1}$.
- the relevant blocks of different comets are disjoint, since the tails of each comet have length $\geq w^h \cdot w^3$, while the blocks relevant to a comet are contained in an interval of length $3w^{h+2}$ intersecting the head.

Note that from CMPred(Π) we can reconstruct Π, and moreover $f(D_k)$ is in the range of CMPred. Hence we can define

$$X := \text{CMPred}^{-1}(f(D_k))$$

and we have $H(X) = H(f(D_k))$. Let B_i be the portion of X in the three blocks relevant to comet i, in our current set of t' comets. Recall that in row $h+2$ of the CMPred distribution Π, each block is given by a variable uniform over a support of size $2 \cdot w^{h+2}$. Hence B_i is a random variable uniform over its support $Supp(B_i)$ of size $(2 \cdot w^{h+2})^3$.

We want to argue that one such variable is close to uniform in our distribution $f(D_k)$. Indeed, recall from the beginning of the proof that

$$H(X) \geq H_\infty(f(D_k)) \geq H(\Pi) - t_{k-1} \cdot O(qw/\alpha)^2.$$

Since $H(X, Y) \leq H(X) + H(Y)$ for any random variables X, Y, we have that,

$$H(B_1, B_2, \ldots, B_{t'}) \geq t' \log |Supp(B_i)| - t_{k-1} \cdot O(qw/\alpha)^2.$$

By Lemma 22, each B_i is α-close to uniform, except for those in a "forbidden" set of size $t_{k-1} \cdot O(q^2w^2/\alpha^4)$.

Now for the critical point, t' is larger than the size of this forbidden set. This is true because we only lost $w^{O(1)}$ factors, so it suffices to make the constant c_1 large enough in the definition of the sequence t_i. Formally,

$$t_{k-1} \cdot O(q^2w^2/\alpha^4) = (m/w^{c_0(q/\alpha)-c_1(k-1)}) \cdot O(q^2w^2/\alpha^4)$$

which is smaller than $t' = m/w^{c_0(q/\alpha)-c_1 k+O(1)}$ for c_1 large enough. Here we are using that $q \leq O(\log m)/\log\log m, w \geq \log m, \alpha = \Theta(1)$.

Breaking correlation in the random comet

At this point we have a w^3-comet (p, i, j) where the head length $(j - i)$ is in $[w^h, w^{h+1}]$ and

(1) The answers to queries i and j are α-independent, and
(2) the relevant blocks are α-close to uniform.

In the query answers consider just the color corresponding to row $h + 2$ for query i and j. Let them be $C(i)$ and $C(j)$.

Because the relevant blocks are α-close to uniform, for any color c we have both $\mathbb{P}[C(i) = c] \leq 1/2 + \alpha$ and $\mathbb{P}[C(j) = c] \leq 1/2 + \alpha$. Also, because $C(i)$ and $C(j)$ are α-independent, we have $\mathbb{P}[C(i) = C(j)] \leq 1/2 + 2\alpha$.

However, $C(i)$ and $C(j)$ are in fact highly correlated. The only event in which $\mathbb{P}[C(i) \neq C(j)]$ is if the head of the comet contains an element. The head has length $\leq w^{h+1}$. The blocks have length w^{h+2}. If the variables in the blocks were uniform, the chance that the head contains an element is $\leq 1/w$. The block is only α-close to uniform, so this probability is $\leq 1/w + \alpha$. Hence, $\mathbb{P}[C(i) = C(j)] \geq 1 - 1/w - \alpha$. For $\alpha = 1/10$, this is larger than the above value of $1/2 - 2\alpha$, concluding the proof.

Reducing CMPred to Pred

We quickly recall this reduction to justify the claim made in the introduction that we obtain a lower bound for Pred under a suitable distribution. Given $x, y \in \{0,1\}^m$ we create $z \in \{0,1\}^{m^3}$ such that $(\text{Pred}(x)_i, \text{Pred}(y)_i)$ depends only on (and therefore can be reduced to computing) $\text{Pred}(z)_j$. Let $x \otimes y$ be the $m \times m$ matrix where the i, j coordinate is $x_i \cdot y_j$. We can also think of this as a vector z in $\{0,1\}^{m^2}$ listing the elements in the matrix in row order. Note that $(\text{Pred}(x)_m, \text{Pred}(y)_m)$ is the same as $\text{Pred}(z)_{m^2}$ written in base m. However to compute $(\text{Pred}(x)_i, \text{Pred}(y)_i)$ for $i < m$ this doesn't quite work. One simple fix is to *zero-out* part of the matrix. Define $x \otimes_i y$ to be the same as $x \otimes y$ except that only the top-left $i \times i$ sub-matrix may be non-zero; Then $(\text{Pred}(x)_i, \text{Pred}(y)_i)$ can be obtained from $\text{Pred}(x \otimes_i y)_{i \cdot m^2}$. Hence we can reduce two instances x and y of Pred to the instance $(x \otimes_1 y, x \otimes_2 y, \ldots)$. Repeat ℓ times for 2^ℓ instances.

7 Proof of Theorem 4

We can assume that $q \leq \log m$, for else the statistical statistical bound is trivial and the theorem is true. We apply the separator Theorem 14 with $\alpha = 1/1000$ and the sequence

$$t_i := m/w^{c_0(q/\alpha) - c_1 i},$$

for constants c_0, c_1 to be set later. For large enough c_1 this satisfies the hypothesis of the theorem that $t_i \geq t_{i-1} \cdot cqw/\alpha$. The hypothesis that $1 - 2^{-t_0} \geq \sqrt{8/|S|} = \sqrt{8/2^m}$ holds as well since $|S| = 2^m$.

Let k and D_k be as given by the theorem. Let

$$X := \textsc{Rank}^{-1} f(D_k).$$

Note that this is a valid definition because $f(D_k)$ is in the range of Rank, and the latter is 1-1. The separator theorem guarantees that $H_\infty(f(D_k)) \geq m - t'_{k-1}$, where

$$t'_{k-1} := t_{k-1} \cdot O(q^2 w^2/\alpha^4).$$

Hence also $H(X) \geq m - t'_{k-1}$.

Comets

We now apply the comet-finding Lemma 21 to the $t_k = m/w^{c_0(q/\alpha)-c_1 k}$ trees given by the separator. For $c = 1$, the lemma gives a set of

$$t'_k := m/w^{c_0(q/\alpha)-c_1 k+O(1)}$$

disjoint w-comets. We shall only use that they are 100-comets, and their head lengths will not be relevant now. We want to find a comet whose outputs are "sufficiently random."

Define $a := t'_{k-1}$ and $b := t'_k$.

Partition X into b consecutive blocks, where each block contains exactly one comet and intersects no others. Let $Z_1, Z_2, \ldots, Z_b$ be the blocks, and let $|Z_i| = s_i$ with $\sum_i s_i = m$. Applying Lemma 22 we find $\geq b - a/\epsilon$ blocks i such that $H(Z_i|Z_1 Z_2 \ldots Z_{i-1}) \geq s_i - \epsilon$. We set $\epsilon = 1/w$ (a sufficiently small constant would be enough), and we verify that $b - a/\epsilon \geq 1$, yielding at least one block i^* such that

$$H(Z_{i^*}|Z_1 Z_2 \ldots Z_{i^*-1}) \geq s_{i^*} - \epsilon. \tag{1}$$

The inequality $b - a/\epsilon \geq 1$ is true because we only lost $w^{O(1)}$ factors, so it suffices to make the constant c_1 large enough in the definition of the sequence t_i. Formally,

$$\frac{b}{a} = \frac{t_k}{t_{k-1}} \cdot \frac{1}{O(q^2 w^2/\alpha^4) \cdot w^{O(1)}} \geq \frac{w^{c_1}}{w^{O(1)}} > w^{100}.$$

The inequalities holds for c_1 large enough and using $q \leq \log m, w \geq \log m, \alpha = \Theta(1)$.

Breaking correlation in the random comet

Hence we now have a comet (p, i, j) that is contained in an interval Z_{i^*} such that:

(1) Equation 1 holds, and
(2) $f_i(D_k), f_j(D_k)$ are α-independent.

The next lemma directly contradicts this and concludes the proof.

▶ **Lemma 23.** *Let $X_1, X_2, \ldots, X_m$ be $0-1$ random variables, and (p, i, j) a c-comet for a sufficiently large c. Let $\ell := i - p$ and $d := j - i$.*

Suppose that

$$H(X_{p+1}, X_{p+2}, \ldots, X_j | X_1, X_2, \ldots, X_p) \geq \ell + d - 1/c.$$

Then there exists an integer t such that

$$\mathbb{P}_X\left[\textsc{Rank}(X)_j \geq t + \ell/2 + d/2 + c^{1/3}\sqrt{d}\right] \geq 1/10, \text{ and}$$
$$\mathbb{P}_X\left[\textsc{Rank}(X)_i < t + \ell/2\right] \geq 1/10, \text{ but}$$
$$\mathbb{P}_X\left[\textsc{Rank}(X)_j \geq t + \ell/2 + d/2 + c^{1/3}\sqrt{d} \bigwedge \textsc{Rank}(X)_i < t + \ell/2\right] \leq 1/1000 (\ll 1/10 \cdot 1/10).$$

Proof. Let us start with the last inequality, because we can prove it without getting our hands on t. The probability is at most

$$\mathbb{P}_X\left[\sum_{k=i+1}^{j} X_k \geq d/2 + c^{1/3}\sqrt{d}\right].$$

By Pinsker's inequality ([14], Chapter 3; Exercise 17) the distribution of $X_{i+1}, X_{i+2}, \ldots, X_j$ is $4/\sqrt{c}$ close to the uniform $U_1 U_2 \ldots U_d$. Hence the above probability is

$$\leq \Pr_U \left[\sum_{k=1}^{d} U_k \geq d/2 + c^{1/3}\sqrt{d} \right] + 4/\sqrt{c} \leq 1/2000 + 4/\sqrt{c} \leq 1/1000.$$

where the second inequality follows from Chebyshev's inequality for sufficiently large c.

We now verify the first two inequalities in the conclusion of the lemma. Let $Y := X_1, X_2, \ldots, X_p$ stand for the prefix, and $Z := X_{p+1}, X_{p+2}, \ldots, X_j$ for the $\ell + d$ high-entropy variables. Let

$$A := \{y \in \{0,1\}^p : H(Z|Y = y) \geq \ell + d - 2/c\}$$

be the set of prefix values conditioned on which Z has high entropy. We claim that $\mathbb{P}[Y \in A] \geq 1/2$. This is because, applying Markov Inequality to the non-negative random variable $\ell + d - H(Z|Y = y)$ (for y chosen according to Y),

$$\begin{aligned}\mathbb{P}[Y \notin A] &= \mathbb{P}_{y \in Y}[\ell + d - H(Z|Y = y) > 2/c] \\ &\leq \mathbb{E}_{y \in Y}[\ell + d - H(Z|Y = y)]/(2/c) \\ &= (\ell + d - H(Z|Y))/(2/c) \leq (1/c)/(2/c) = 1/2.\end{aligned}$$

Note that for every $y \in A$ we have, by definition, that the $(\ell + d)$-bit random variable $(Z|Y = y)$ has entropy at least $\ell + d - 2/c$, and so by Pinsker's inequality ([14], Chapter 3; Exercise 17) the random variable $(Z|Y = y)$ is $(\epsilon := 4\sqrt{2/c})$-close to uniform over $\{0,1\}^{\ell+d}$. Therefore, for any subset $S \subseteq A$, the random variable

$$(Z|Y \in S) \text{ is } \epsilon\text{-close to uniform over } \{0,1\}^{\ell+d}. \tag{2}$$

Now define t to be the largest integer such that

$$\mathbb{P}\left[Y \in A \wedge \textsc{Rank}(X)_p \geq t\right] \geq 1/4. \tag{3}$$

Since by definition of t we have $\mathbb{P}[Y \in A \wedge \textsc{Rank}(X)_p \geq t + 1] < 1/4$, we also have

$$\mathbb{P}\left[Y \in A \wedge \textsc{Rank}(X)_p \leq t\right] \geq 1/2 - 1/4 = 1/4. \tag{4}$$

We obtain the desired conclusions as follows, denoting by $U_1, U_2, \ldots,$ uniform and independent $0-1$ random variables. The first probability in the conclusion of the lemma is at least

$$\mathbb{P}\left[\textsc{Rank}(X)_j \geq t + (\ell + d)/2 + \sqrt{\ell}/c^{1/6}\right]$$

because $\ell \geq c \cdot d$.

Writing $\text{Rank}(X)_j$ as the sum of the first p bits and the rest, the above probability is at least

$$\mathbb{P}[\sum_{k \leq \ell+d} Z_k \geq (\ell + d)/2 + \sqrt{\ell}/c^{1/6} | Y \in A \wedge \textsc{Rank}(X)_p \geq t] \cdot \mathbb{P}[Y \in A \wedge \textsc{Rank}(X)_p \geq t].$$

The second factor is $\geq 1/4$ by (3). Also by (2) in the first factor we can replace the Z_k with uniform bits changing the probability by at most ϵ. Hence the first factor is at least

$$\mathbb{P}[\sum_{k \leq \ell+d} U_k \geq (\ell + d)/2 + \sqrt{\ell}/c^{1/6}] - \epsilon.$$

In turn the probability is

$$\geq 1/2 - \sqrt{\ell}/c^{1/6} \cdot \Theta(1/\sqrt{\ell}) \geq 1/2 - \Theta(1/c^{1/6})$$

using an estimate of the central binomial coefficient provided e.g. in [13], Lemma 17.5.1. Overall, the first probability in the conclusion of the lemma is

$$\left(1/2 - \Theta(1/c^{1/6}) - \epsilon\right)(1/4) \geq 1/10$$

for large enough c.

We now turn to the second probability in the conclusion of the lemma. Proceeding in a similar way, this probability is at least

$$\mathbb{P}\left[\sum_{k\leq \ell} Z_k < \ell/2 \middle| Y \in A \wedge \sum_k Y_k \leq t\right] \cdot \mathbb{P}\left[Y \in A \wedge \sum_k Y_k \leq t\right]$$
$$\geq \left(\mathbb{P}\left[\sum_{k\leq \ell} U_k < \ell/2\right] - \epsilon\right) \cdot (1/4) \geq (1/2 - \epsilon) \cdot (1/4) \geq 1/10$$

for all sufficiently large c. Here the second inequality uses (2) and (4). ◀

8 Proof of Corollary 12

We claim that there is a depth-q $f : W^s \to \Sigma^m$ such that the statistical distance between $f(U)$ and S is at most $1 - \Omega(1-\delta)/2^c$. Then the result follows from the sampling lower bounds.

To prove the claim, consider fixing the communication transcript of the protocol to i. Because the communication is fixed, each party can be implemented as a depth-q forest. If the protocol dictates Party j to send a message that does not match i, Party j outputs any value and stops. Let $C = 2^c$ and consider the C forests f_i where each f_i corresponds to the protocols run with fixed communication transcript i. The result now follows from the next lemma, letting P_i be the output distribution of f_i, and $Q_i(x)$ the probability that f outputs x using transcript i.

▶ **Lemma 24.** *Let P be a distribution and S a set. Suppose $P(x) = \sum_{i=1}^{C} Q_i(x)$ where each $Q_i(x) \in [0,1]$ but $\sum_x Q_i(x) = 1$ is not required. Suppose $\Delta(P,S) = \delta$. Define P_i to be the probability distribution with $P_i(x) = Q_i(x)$ and the remainder $1 - \sum_x Q_i(x)$ mass is put arbitrarily.*

Then there exists i such that $\Delta(P_i, S) \leq 1 - \epsilon$ where $\epsilon := 0.1 \cdot (1-\delta)/C$.

Proof. We use that

$$\Delta(A,B) = \sum_{x : A(x) \geq B(x)} A(x) - B(x).$$

Suppose there exists i such that

$$\sum_{x : Q_i(x) \leq S(x)} Q_i(x) \geq \epsilon.$$

Then $\Delta(P_i, S) = \sum_{x : P_i(x) \leq S(x)} S(x) - P_i(x) \leq 1 - \epsilon$ and we are done.

Let

$T_i := \{x \in S : Q_i(x) \geq 1/|S|\} \subseteq S.$

By above each Q_i puts mass at most ϵ outside of T_i. Now we show that the mass in T_i is concentrated on few points. Suppose $|T_i| \geq \epsilon|S|$ for some i. Then $\Delta(P_i, S) = \sum_{x:S(x)\leq P_i(x)} P_i(x) - S(x) \leq 1 - \sum_{x\in T_i} 1/|S| \leq 1 - \epsilon$ and we are done.

Now we can contradict the hypothesis. Let

$T := \{x \in S : P(x) \geq 1/|S|\} \subseteq S.$

Note that $T_i \subseteq T$ for every i. Hence each Q_i contributes $\leq \epsilon|S|$ elements to T via T_i, and further contributes ϵ mass to distribute for others. With mass α we obtain $\leq \alpha|S|$ elements such that $P(x) \geq S(x)$. Hence $|T| \leq C\epsilon|S| + C\epsilon|S| = 2C\epsilon|S|$.

Let α, β, γ be respectively the masses that P puts outside of S, in T, and in $S \setminus T$. Note that $\gamma \leq C\epsilon$, since each Q_i puts $\leq \epsilon$ mass on $S \setminus T_i$. We have

$$\Delta(P, S) = \sum_{x:P(x)\geq S(x)} P(x) - S(x) = \alpha + \beta - \sum_{x\in T} S(x) \geq \alpha + \beta - 2C\epsilon.$$

We have $\alpha + \beta = 1 - \gamma \geq 1 - C\epsilon$.

Hence we get

$\delta \geq \Delta(P, S) \geq 1 - 3C\epsilon,$

as desired. ◀

References

1 Miklós Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24(1):1–48, 1983.

2 Andris Ambainis, Leonard J. Schulman, Amnon Ta-Shma, Umesh V. Vazirani, and Avi Wigderson. The quantum communication complexity of sampling. *SIAM J. on Computing*, 32(6):1570–1585, 2003.

3 Sanjeev Arora and Boaz Barak. *Computational Complexity*. Cambridge University Press, 2009. A modern approach.

4 Paul Beame, Russell Impagliazzo, and Srikanth Srinivasan. Approximating AC^0 by small height decision trees and a deterministic algorithm for #AC^0SAT. In *Proceedings of the 27th Conference on Computational Complexity, CCC 2012, Porto, Portugal, June 26-29, 2012*, pages 117–125. IEEE Computer Society, 2012. doi:10.1109/CCC.2012.40.

5 Chris Beck, Russell Impagliazzo, and Shachar Lovett. Large deviation bounds for decision trees and sampling lower bounds for AC0-circuits. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:42, 2012.

6 Chris Beck, Russell Impagliazzo, and Shachar Lovett. Large deviation bounds for decision trees and sampling lower bounds for AC0-circuits. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 101–110, 2012.

7 Avraham Ben-Aroya, Dean Doron, and Amnon Ta-Shma. Explicit two-source extractors for near-logarithmic min-entropy. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:88, 2016. URL: http://eccc.hpi-web.de/report/2016/088, arXiv:TR16-088.

8 Itai Benjamini, Gil Cohen, and Igor Shinkar. Bi-lipschitz bijection between the boolean cube and the hamming ball. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2014.

9 Eshan Chattopadhyay, Jesse Goodman, and David Zuckerman. The space complexity of sampling. *Electron. Colloquium Comput. Complex.*, page 106, 2021. URL: https://eccc.weizmann.ac.il/report/2021/106, arXiv:TR21-106.

10 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *ACM Symp. on the Theory of Computing (STOC)*, pages 670–683, 2016.

11 Gil Cohen. Making the most of advice: New correlation breakers and their applications. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 188–196, 2016. doi:10.1109/FOCS.2016.28.

12 Gil Cohen and Leonard J. Schulman. Extractors for near logarithmic min-entropy. *Electronic Colloquium on Computational Complexity (ECCC)*, 23:14, 2016.

13 Thomas Cover and Joy Thomas. *Elements of Information Theory (Wiley Series in Telecommunications and Signal Processing)*. Wiley-Interscience, 2006.

14 Imre Csiszar and Janos Korner. *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Academic Press, Inc., 1982.

15 Anindya De and Thomas Watson. Extractors and lower bounds for locally samplable sources. In *Workshop on Randomization and Computation (RANDOM)*, 2011.

16 P. Erdős and R. Rado. Intersection theorems for systems of sets. *J. London Math. Soc.*, 35:85–90, 1960. doi:10.1112/jlms/s1-35.1.85.

17 Merrick L. Furst, James B. Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, 1984.

18 Anna Gál and Peter Bro Miltersen. The cell probe complexity of succinct data structures. *Theoretical Computer Science*, 379(3):405–417, 2007.

19 Alexander Golynski. Cell probe lower bounds for succinct data structures. In *20th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 625–634, 2009.

20 Mika Göös and Thomas Watson. A lower bound for sampling disjoint sets. *ACM Trans. Comput. Theory*, 12(3):20:1–20:13, 2020. doi:10.1145/3404858.

21 Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by adaptive procedures with advice, and lower bounds on hardness amplification proofs. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2018. Available at https://www.ccs.neu.edu/home/viola/papers/adaptivemajority.pdf.

22 Johan Håstad. *Computational limitations of small-depth circuits*. MIT Press, 1987.

23 Johan Håstad. On the correlation of parity and small-depth circuits. *SIAM J. on Computing*, 43(5):1699–1708, 2014.

24 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 961–972, 2012.

25 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Springer, 2012.

26 Xin Li. Improved two-source extractors, and affine extractors for polylogarithmic entropy. In *IEEE Symp. on Foundations of Computer Science (FOCS)*, 2016.

27 Mingmou Liu and Huacheng Yu. Lower bound for succinct range minimum query. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *ACM Symp. on the Theory of Computing (STOC)*, pages 1402–1415. ACM, 2020. doi:10.1145/3357713.3384260.

28 Shachar Lovett and Emanuele Viola. Bounded-depth circuits cannot sample good codes. *Computational Complexity*, 21(2):245–266, 2012.

29 Peter Bro Miltersen. Cell probe complexity – A survey, 1999. Invited talk/paper at Advances in Data Structures (Pre-conference workshop of FSTTCS'99).

30 Ryan O'Donnell. *Analysis of Boolean Functions*. Cambridge University Press, 2014.

31 Mihai Pătraşcu. Succincter. In *49th IEEE Symp. on Foundations of Computer Science (FOCS)*. IEEE, 2008.

32 Mihai Patrascu and Mikkel Thorup. Time-space trade-offs for predecessor search. In Jon M. Kleinberg, editor, *ACM Symp. on the Theory of Computing (STOC)*, pages 232–240. ACM, 2006. doi:10.1145/1132516.1132551.

33 Mihai Pătraşcu and Emanuele Viola. Cell-probe lower bounds for succinct partial sums. In *21th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 117–122, 2010.

34 Alexander A. Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Ann. of Math.*, 181(2):415–472, 2015. doi:10.4007/annals.2015.181.2.1.

35 Nathan Segerlind, Sam Buss, and Russell Impagliazzo. A switching lemma for small restrictions and lower bounds for k-DNF resolution. *SIAM J. on Computing*, 33(5):1171–1200, 2004.

36 Mikkel Thorup. Mihai patrascu: Obituary and open problems. *Bulletin of the EATCS*, 109:7–13, 2013. URL: http://albcom.lsi.upc.edu/ojs/index.php/beatcs/article/view/163/176.

37 Emanuele Viola. The Complexity of Distributions, Fall 2018 talk at the Simons Institute. https://www.youtube.com/watch?v=078b085HE3w.

38 Emanuele Viola. Bit-probe lower bounds for succinct data structures. *SIAM J. on Computing*, 41(6):1593–1604, 2012.

39 Emanuele Viola. The complexity of distributions. *SIAM J. on Computing*, 41(1):191–218, 2012.

40 Emanuele Viola. Extractors for turing-machine sources. In *Workshop on Randomization and Computation (RANDOM)*, 2012.

41 Emanuele Viola. Extractors for circuit sources. *SIAM J. on Computing*, 43(2):355–972, 2014.

42 Emanuele Viola. Lower bounds for data structures with space close to maximum imply circuit lower bounds. *Theory of Computing*, 15:1–9, 2019. URL: https://theoryofcomputing.org/articles/v015a018/v015a018.pdf.

43 Emanuele Viola. Sampling lower bounds: boolean average-case and permutations. *SIAM J. on Computing*, 49(1), 2020. Available at https://www.ccs.neu.edu/home/viola/papers/sampling-lower-bounds.pdf.

44 Emanuele Viola, 2022. https://emanueleviola.wordpress.com/2022/09/14/myth-creation-the-switching-lemma/.

45 Andrew Yao. Separating the polynomial-time hierarchy by oracles. In *26th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 1–10, 1985.

46 Andrew Chi-Chih Yao. Should tables be sorted? *J. of the ACM*, 28(3):615–628, 1981. doi:10.1145/322261.322274.

47 Huacheng Yu. Optimal succinct rank data structure via approximate nonnegative tensor decomposition. In Moses Charikar and Edith Cohen, editors, *ACM Symp. on the Theory of Computing (STOC)*, pages 955–966. ACM, 2019. doi:10.1145/3313276.3316352.

A Proof of the fixed-set Lemma 19

Begin with R equal to the uniform distribution over W^s. If there are q words and q values such that the probability of getting those values in B is larger than $(1+\alpha)/W^q$ then we fix them to those values, in both B and R. Now we have subsets of W^{s-q}, the loss has decreased by an additive $\log_2 1/(1+\alpha) = \Omega(\alpha)$, and we repeat the process.

Because the initial loss was b, this process stops after $O(b/\alpha)$ iterations. In the end, the loss inside the final universe is at most b, since we never increase loss. With respect to the original universe, because we fixed $O(qb/\alpha)$ words, the loss is at most $O(wqb/\alpha) + b \leq b \cdot O(wq/\alpha)$.

Let B_1 and R be the distributions when the process stops. Consider any tree $g : W^s \to \{0,1\}$ of depth q. Let P be the collection of paths in g leading to the output 1. Note that each path $p \in P$ corresponds to q input words and q values for them. Write $\mathbb{P}_X[p]$ for the probability of following path p under distribution X. By above we have $\mathbb{P}_{B_1}[p] \leq (1+\alpha)/W^q = (1+\alpha)\mathbb{P}_R[p]$.

Hence

$$\mathbb{P}[g(B_1) = 1] = \sum_{p \in P} \mathbb{P}_{B_1}[p] \leq \sum_{p \in P} (1+\alpha)\mathbb{P}_R[p] = (1+\alpha)\mathbb{P}[g(R) = 1].$$

And so in particular $\mathbb{P}[g(B_1) = 1] \leq \mathbb{P}[g(R) = 1] + \alpha$.

Repeating the argument with 0 and 1 swapped yields the lemma for trees with boolean alphabet. To prove the lemma for a tree g' with arbitrary alphabet, reduce to the case of boolean alphabet in the following standard way. Suppose that the statistical distance between $g'(R)$ and $g'(B_1)$ is $> \alpha$. This means that there exists a set T such that

$$|\mathbb{P}[g'(R) \in T] - \mathbb{P}[g'(B_1) \in T]| > \alpha.$$

Define tree g with boolean output as $g(x) := 1$ iff $g'(x) \in T$; note this just amounts to changing the labels of the leaves of g'. Now the left-hand side of the inequality above can be written as

$$|\mathbb{P}[g(R) = 1] - \mathbb{P}[g(B_1) = 1]|$$

and this contradicts the result for trees with boolean outputs and concludes the proof of the lemma.

A Ihara-Bass Formula for Non-Boolean Matrices and Strong Refutations of Random CSPs

Tommaso d'Orsi
Department of Computer Science, ETH Zürich, Switzerland

Luca Trevisan
Department of Computing Sciences, Bocconi University, Milano, Italy

Abstract

We define a novel notion of "non-backtracking" matrix associated to any symmetric matrix, and we prove a "Ihara-Bass" type formula for it.

We use this theory to prove new results on polynomial-time strong refutations of random constraint satisfaction problems with k variables per constraints (k-CSPs). For a random k-CSP instance constructed out of a constraint that is satisfied by a p fraction of assignments, if the instance contains n variables and $n^{k/2}/\epsilon^2$ constraints, we can efficiently compute a certificate that the optimum satisfies at most a $p + O_k(\epsilon)$ fraction of constraints.

Previously, this was known for even k, but for odd k one needed $n^{k/2}(\log n)^{O(1)}/\epsilon^2$ random constraints to achieve the same conclusion.

Although the improvement is only polylogarithmic, it overcomes a significant barrier to these types of results. Strong refutation results based on current approaches construct a certificate that a certain matrix associated to the k-CSP instance is quasirandom. Such certificate can come from a Feige-Ofek type argument, from an application of Grothendieck's inequality, or from a spectral bound obtained with a trace argument. The first two approaches require a union bound that cannot work when the number of constraints is $o(n^{\lceil k/2 \rceil})$ and the third one cannot work when the number of constraints is $o(n^{k/2}\sqrt{\log n})$.

We further apply our techniques to obtain a new PTAS finding assignments for k-CSP instances with $n^{k/2}/\epsilon^2$ constraints in the semi-random settings where the constraints are random, but the sign patterns are adversarial.

2012 ACM Subject Classification Theory of computation; Mathematics of computing → Probability and statistics

Keywords and phrases CSP, k-XOR, strong refutation, sum-of-squares, tensor, graph, hypergraph, non-backtracking walk

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.27

Related Version *Full Version*: arXiv:2204.10881

Funding This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreements No 815464 and 834861).

Acknowledgements Thanks to Pravesh Kothari for useful discussions about semi-random CSPs.

1 Introduction

If we take a random instance of 3SAT with n variables and $m \geq cn$ clauses where c is a sufficiently large constant, then almost surely the instance is not satisfiable. Indeed, an instance of random 3SAT with n variables and n/ϵ^2 clauses is almost surely such that at most a $7/8 + O(\epsilon)$ fraction of clauses can be simultanously satisfied by the best assignment. Finding a *certificate* that a specific random formula exhibits such behaviour is, however, believed to be quite hard.

In 2002, Feige [8] formulated the hypothesis that it is computationally intractable to find *strong refutations* of random 3-SAT formulas when the number of clauses is slightly superlinear in the number of variables. A *strong refutation* of a 3-SAT formula is a certificate, verifiable in polynomial time, that every assignment fails to satisfy a constant fraction of the clauses. Feige proved that his hypothesis has several consequences for the hardness of approximation of various problems.

Because of its centrality to the theories of proof complexity and of average-case complexity, and its connection to other questions in cryptography, computational complexity, and statistical physics, the complexity of strong refutations for random 3SAT and other random constraint satisfaction problems has been extensively studied since the 1980s.

Among several important algorithmic milestones, we mention the idea of using spectral techniques to find refutations and strong refutations (introduced in [13, 12] and then refined in subsequent work) and a reduction from the problem of finding strong refutations for random 3SAT to the problem of finding strong refutations for random 3XOR (introduced in [8] and then refined in subsequent work).

The state of the art concerning polynomial-time computable strong refutations of random constraint satisfaction problems is a 2015 paper by Allen, O'Donnell and Witmer [2]. We refer the reader to the introduction of [2] for an extended survey of algorithmic ideas and results related to refutations of random constraint satisfaction problems. Allen, O'Donnell and Witmer [2] show how to obtain strong refutations for random k-XOR constraint satisfaction problems on n variables and $n^{k/2}(\log n)^{O(1)}$ constraints. When k is even, $O(n^{k/2})$ constraints suffice. Thanks to a reduction from arbitrary constraint satisfaction to k-XOR (of which we provide a self-contained simpler proof in the full version of the paper), similar bounds hold for any constraint satisfaction problem over k variables.

To illustrate the difference between odd k and even k, we briefly discuss how a strong refutation for random 4-XOR and random 3-XOR instances is constructed.

In general, if we have an instance of k-XOR with m constraints and n variables, a strong refutation is a certificate that

$$\max_{x\in\{-1,1\}^n} \sum_{i_1,\ldots,i_k} T_{i_1,\ldots,i_k} x_{i_1}\cdots x_{i_k} \leq \epsilon m$$

where T is a symmetric tensor of order k such that $T_{i_1,\ldots,i_k} = 0$ if there is no constraint on the k-tuple of variables $x_{i_1},\ldots,x_{i_k}$, and otherwise $T_{i_1,\ldots,i_k} = \pm 1$ depending on the right-hand-side of the constraint.

When $k = 4$, we can flatten the tensor to an $n^2 \times n^2$ symmetric matrix M (where $M_{(a,b),(c,d)} = T_{a,b,c,d}$) and we have

$$\max_{x\in\{-1,1\}^n} \sum_{i_1,\ldots,i_4} T_{i_1,\ldots,i_4} x_{i_1}\cdots x_{i_4} = \max_{x\in\{-1,1\}^n} (x^{\otimes 2})^{\mathsf{T}} M x^{\otimes 2}$$

Now we can relax the right-hand side to a maximization over arbitrary n^2-dimensional Boolean vectors and further relax to the ∞-to-1 norm:

$$\max_{x\in\{-1,1\}^n} (x^{\otimes 2})^{\mathsf{T}} M x^{\otimes 2} \leq \max_{y\in\{-1,1\}^{n^2}} y^{\mathsf{T}} M y \leq \max_{y,z\in\{-1,1\}^{n^2}} y^{\mathsf{T}} M z = ||M||_{\infty\to 1}$$

Finally, the last expression above can be upper bounded by ϵm, by using Chernoff bounds and a union bound over all the 2^{2n^2} possible choices for y and z, which is possible if m is a sufficiently large constant times n^2/ϵ^2. Finally, we can use Grothendieck's inequality to get us a certified upper bound of the $\infty \to 1$ norm in polynomial time up to a constant factor.

For 3-XOR, the idea is to apply a Cauchy-Schwarz step to reduce the problem of bounding a degree-4 problem, and then to flatten the resulting 4-tensor to an $n^2 \times n^2$ matrix M such that

$$\max_{x\in\{-1,1\}^n} \sum_{i_1,i_2,i_4} T_{i_1,i_2,i_3} x_{i_1} x_{i_2} x_{i_3} \leq \sqrt{n} \cdot \sqrt{\max_{x\in\{\pm 1\}^n} (x^{\otimes 2})^\intercal M x^{\otimes 2}} \leq \sqrt{n} \cdot \sqrt{\max_{y,z\in\{\pm 1\}^{n^2}} y^\intercal M z}$$

Unfortunately, now it is not possible any more to bound the maximum on the above right-hand via a union bound over 2^{2n^2} cases. Indeed, for this to be possible, we would need our distribution to have at least order of n^2 bits of entropy, and so we would need to have order of n^2 constraints.

The alternative is to obtain a bound in terms of the spectral norm of M, using the fact that

$$\max_{y,z\in\{\pm 1\}^{n^2}} y^\intercal M z \leq n^2 \cdot ||M|| \ .$$

But for a sparse matrix to have a non-trivial bound on its spectral norm, we have to have at least $\operatorname{poly}\log n$ non-zero entries per row on average[1], and for this to happen the number of constraints has to be at least of the order of $n^{1.5}\operatorname{poly}\log n$. In the regime of $n^{1.5}\operatorname{poly}\log n$ random 3-XOR constraints, a spectral norm bound on M can be established via trace methods, and this is how the results of [2] are proved in the case of odd k.

Semi-random CSPs

The complementary question to that of certifying strong refutations, concerns the design of algorithms that satisfy as-many-as-possible clauses in the given CSP instance. As for refutations, complexity theory paints a grim picture for (approximately) solving worst case instances [18, 6, 11]. But, in the average case, polynomial time approximation schemes are known [3, 1] when the number of clauses is of the order $n^{k/2}(\log n)^{O(1)}$.

The algorithmic techniques behind these PTAS are closely related to those used for refutations and, in particular, again boils down to studying the spectrum of the flattened tensor representing the instance.

Remarkably, groundbreaking work [15], showed that a similar picture holds in the significantly more general settings of *smoothed* CSPs: where both the literal negation patterns and clauses are chosen arbitrarily, but then signs are randomly flipped with a small, yet constant, probability.[2].

1.1 Our Results

Strong refutations

Our first result breaks the $n^{k/2}\operatorname{poly}\log n$ barrier for strong refutations of random k-XOR instances, with odd k.

[1] This is similar to the phenomenon that the quasirandomness of a $G_{n,p}$ random graphs can be certified in terms of the non-trivial eigenvalues of the adjacency matrix only if the average degree is at least logarithmic. We will return to the graph analogy shortly.

[2] Smoothed CSPs were first introduced in [9]

▶ **Theorem 1** (Strong refutations of random k-XOR). *There exists an efficient algorithm that, given an instance $\boldsymbol{\mathcal{I}}$ of random k-XOR with $n^{k/2}/\epsilon^2$ constraints, with probability at least* 0.99*, finds strong refutation of $\boldsymbol{\mathcal{I}}$, that is, a certificate that*

$$\mathrm{Opt}_{\boldsymbol{\mathcal{I}}} \leq \frac{1}{2} + O(\epsilon)\,.$$

Using the known reduction of general k-CSP to k-XOR, of which we provide a simple self-contained proof, we have the following consequence.

▶ **Theorem 2** (Strong refutations of random CSPs). *Let $P : \{-1,+1\}^k \to \{0,1\}$ be a Boolean k-ary predicate, and call $\mathbb{E}\,P$ the probability that P is satisfied by a random assignment. There exists a polynomial time algorithm that given a random instance $CSP(P)$ instances $\boldsymbol{\mathcal{I}}$, over n variables, with at least $n^{k/2}/\epsilon^2$ constraints, with probability at least* 0.99*, finds a strong refutation of $\boldsymbol{\mathcal{I}}$, that is, a certificate that*

$$\mathrm{Opt}_{\boldsymbol{\mathcal{I}}} \leq \mathbb{E}\,P + O(\epsilon)\,.$$

Robust approximation algorithms against adversarial sign patterns

Our techniques can be further applied to design efficient algorithms finding an assignment with value $\mathrm{Opt} - O(\epsilon)$ beyond the $n^{k/2}\,\mathrm{polylog}\,n$ barrier. Our sharp results not only works for random instances, but also in the semi-random settings where: *first*, clauses are sampled randomly, and *second*, given the instance, the sign pattern of *each* clause is adversarially perturbed. Such perturbations are not captured by the smooth models of [9, 15] and hence require different algorithmic challenges. In the special case of even k, [17] provided a PTAS whenever $p \geq n^{k/2}\,\mathrm{polylog}\,n$.

▶ **Theorem 3** (Algorithm for k-XOR with adversarial patterns). *Let $n\,, k$ be positive integers, $\epsilon > 0\,, n$ and $n^{-k/2}/\epsilon^2 < 1$. Let $\mathcal{I}$ be a k-XOR instance constructed through the following process:*

- *Sample a random k-XOR instance $\boldsymbol{\mathcal{I}}'$ with at least $n^{k/2}/\epsilon^2$ constraints.*
- *Given $\boldsymbol{\mathcal{I}}'$, arbitrarily (possibly adversarially) replace the sign of each clause in $\boldsymbol{\mathcal{I}}'$.*

There exists a randomized algorithm, running in time $n^{O(k/\epsilon^2)}$, that returns an assignment $\hat{\mathbf{x}}$ with value

$$\mathrm{Val}_{\mathcal{I}}(\hat{\mathbf{x}}) \geq \mathrm{Opt}_{\mathcal{I}} - O(\epsilon)\,,$$

with probability at least 0.99*.*

As in the case of strong refutations, Theorem 3 can be extended to k-CSPs.

▶ **Theorem 4** (Algorithm for semi-random k-CSPs). *Let $n\,, k$ be positive integers, $\epsilon > 0\,, n$ and $n^{-k/2}/\epsilon^2 < 1$. Let $P : \{-1,+1\}^k \to \{0,1\}$ be a Boolean k-ary predicate. Let $\mathcal{I}$ be a $CSP(P)$ instance constructed through the following process:*

- *Sample a random $CSP(P)$ instance $\boldsymbol{\mathcal{I}}'$ with at least $n^{k/2}/\epsilon^2$ constraints.*
- *Given $\boldsymbol{\mathcal{I}}'$, for each clause in $\boldsymbol{\mathcal{I}}'$, replace the sign pattern with an arbitrary (possibly adversarial) sign pattern.*

There exists a randomized algorithm, running in time $n^{O(k/\epsilon^2)}$, that returns an assignment $\hat{\mathbf{x}}$ with value

$$\mathrm{Val}_{\mathcal{I}}(\hat{\mathbf{x}}) \geq \mathrm{Opt}_{\mathcal{I}} - O(\epsilon)\,,$$

with probability at least 0.99*.*

1.2 Our Techniques

We develop new techniques to bound[3]

$$\max_{x\in\{\pm 1\}^N} x^{\mathsf{T}}\mathbf{M}x \tag{1}$$

when $\mathbf{M}$ is a random $N\times N$ matrix with only a constant expected number of non-zero entries per row and per column, and in which such entries are not independent.

A toy problem

Before we explain our ideas, consider the following question, which models some of the difficulties that we encounter: suppose that we are given a random graph on N vertices, and such that every edge exists with probability d/N, where d is a constant, but the edges are only known to be $poly\log N$-wise independent, and not fully independent. Can we certify that the graph has interesting quasirandom properties, for example can we certify that the Max Cut optimum is at most a $1/2+O(1/\sqrt{d})$ fraction of edges?

One approach could be to bound $||\mathbf{A}-\mathbb{E}\,\mathbf{A}||_{\infty\to 1}$ where $\mathbf{A}$ is the adjacency matrix of the graph. If the graph has mutually independent random edges, that is, if it is sampled from an Erdős-Reniy distribution $G_{N,\frac{d}{N}}$, then we can use a union bound over 2^{2N} cases to argue that with high probability

$$||\mathbf{A}-\mathbb{E}\,\mathbf{A}||_{\infty\to 1}\leq O(\sqrt{d}N)$$

which is certifiable in polynomial time, up to a constant factor loss, using Grothendieck's inequality and which certifies that the Max Cut optimum is at most $1/2+O(1/\sqrt{d})$. Unfortunately, if the edges are only polylog N-wise independent, then it is not possible to take such union bound.

Another option in the fully independent case is to use the results of Feige and Ofek [10], which show that, after removing nodes of degree larger than, say, $10d$, the adjacency matrix of the residual graph has second eigenvalue at most $O(\sqrt{d})$ with high probability. Unfortunately the proof of Feige and Ofek also relies on a union bound over $2^{O(N)}$ cases, and so it cannot work in the polylog N-wise independent case.

A trace argument can be used to prove that, with high probability, we have

$$||\mathbf{A}-\mathbb{E}\,\mathbf{A}||\leq O(\sqrt{d\log N})$$

which provides a polynomial time certificate that the Max Cut optimum is at most $1/2+O(\sqrt{\log N}/\sqrt{d})$, and the trace calculation only requires $O(\log N)$-wise independence. It does, however, introduce an extra logarithmic factor, which is unavoidable because the spectral norm of $||\mathbf{A}-\mathbb{E}\,\mathbf{A}||$ is $\tilde{\Omega}(\sqrt{\log N})$ when d is constant.

It is conceivable that one could prove the result of Feige and Ofek (that the adjacency matrix has second largest eigenvalue $O(\sqrt{d})$ after the removal of high-degree vertices) through a trace bound on the adjacency matrix of the truncated graph, although it seems very difficult to deal with the conditional distribution of edges given that the edges survive the truncation.

[3] We use boldface to denote random variables.

A solution to the toy problem

Although all standard techniques fail, there is a way to combine certain recent results to solve our toy problem. The starting point is the fact that, given an undirected graph $G = (V, E)$, we can define the "non-backtracking" $2|E| \times 2|E|$ matrix B of G, and that this matrix satisfies the Ihara-Bass equation

$$\det(\mathrm{Id} - xB) = (1 - x^2)^{|E|-|V|} \cdot \det(\mathrm{Id} - xA + x^2(D - \mathrm{Id}))$$

where A is the adjacency matrix of the graph, D is the diagonal matrix of degrees, and the above equation holds as an identity of polynomials of degree $2|E|$ in x. See the survey of Horton [16] for an exposition of these definitions and results.

Fan and Montanari [7] show that bounds on the spectral radius of B imply useful PSD inequalities on A. In particular, if $\lambda_{\min}$ is the smallest real eigenvalue of B, then we have

$$A \succeq -|\lambda_{\min}| \cdot \mathrm{Id} - \frac{1}{|\lambda_{\min}|} \cdot (D - \mathrm{Id})$$

In the context of their work on the Stochastic Block Model, Bordenave, Lalarge and Massoulié [5] use a trace argument to prove a result that implies that $\lambda_{\min} \geq -(1 + o(1)) \cdot \sqrt{d}$ in $G_{N,\frac{d}{N}}$ random graphs, and so all these results together imply that the Max Cut of a $G_{N,\frac{d}{N}}$ random graph is with high probability at most $1/2 + (1 + o(1))/\sqrt{d}$, and that this upper bound is efficiently certifiable, for example by the dual of the Goemans-Williamson relaxation.

The key point is that there was never a union bound over $2^{O(N)}$ cases in the above argument and that, in fact, everything works assuming polylog N-wise independence of the edges.[4]

From unweighted graphs to general symmetric matrices

Our goal is to develop an analog of this argument where we work with the $n^2 \times n^2$ matrix M that comes up in the analysis of 3-XOR (or, in general, with the $n^{\lceil k/2 \rceil} \times n^{\lceil k/2 \rceil}$ matrix that comes up in the analysis of k-XOR when k is odd) instead of the adjacency matrix A of the pseudorandom graph analysed above.

The first challenge in carrying out this program is that the original notion of non-backtracking matrix is defined only with respect to 0/1 Boolean symmetric matrices, while we want to study matrices with positive and negative entries that can be arbitrary integers.

A certain generalization of non-backtracking matrices was already introduced in [20, 7], however for technical reasons *we* were not able to use it to carry out our program. We thus introduce a novel theory of "non-backtracking" matrices associated to any given symmetric matrix. In Section 3, given a symmetric $N \times N$ matrix M with Nz non-zero entries, we define an $Nz \times Nz$ "non-backtracking" matrix B_M associated to M, and we prove (see Theorem 7) an Ihara-Bass-type identity

$$\det(\mathrm{Id} - xB_M + x(L_M - J_M)) = (1 - x^2)^{Nz/2-N} \cdot \det(\mathrm{Id} - xM + x^2(D_M - \mathrm{Id}))$$

[4] Incidentally, this combination of Fan-Montanari ideas and Bordenave-Lalarge-Massouli's bounds, also implies that if A' is the adjacency matrix of a graph G sampled from a distribution in which edges have probability d/N and are polylog N wise independent, and then truncated by removing all vertices of degree more than, say, $10d$, then we have with high probability $A' \succeq -O(\sqrt{d}) \cdot I$, proving a one-sided version of the result of Feige and Ofek.

where D_M, L_M and J_M are certain matrices that are associated to M. When M is Boolean, $L_M = J_M$ and D_M is the diagonal matrix such that $(D_M)_{i,i} = \sum_j M_{i,j}$, so our equation becomes the standard Ihara-Bass equation in the case of Boolean M. Conveniently, closed non-backtracking walks W arising from the definition of B_M take value in $\{\pm \prod_{(i,j)\in W} M_{ij}\}$, allowing one to easily mimic arguments used for standard non-backtracking matrices.

Now, given a bound on the spectral radius of $B_M - L_M + J_M$, it is possible, with an argument in the style of Fan and Montanari, to deduce a certifiable bound on the ∞-to-1 norm of M.

Bounding the spectral radius via weighted hyper-walks

Studying the spectral radius of $B_{\mathbf{M}} - L_{\mathbf{M}} + J_{\mathbf{M}}$ –matrices associated to the matrix $\mathbf{M}$ coming from random k-XOR instances– is the main technical challenge of this work.

Our bound relies on a trace argument of $B_{\mathbf{M}}$. However, compared to Bordenave, Lalarge and Massoulié [5] our setup presents a number of new technical challenges.

One challenge comes from the extra terms that we have in the non-Boolean case. In particular, our non-backtracking matrix $B_{\mathbf{M}}$ has entries that are the absolute values of certain entries of $\mathbf{M}$. To compute an expectation of the trace of the symmetrization of a power of $B_{\mathbf{M}}$, we replace absolute values with squares, and bound the error that we incur because of this.

Perhaps the most important challenge comes from the fact that the trace bound ultimately boils down to a weighted count of certain closed "hypergraph walks" performed on the hypergraph corresponding to constraints of the k-XOR instance. These objects arise from our notion of non-backtracking walks on the symmetric matrix $\mathbf{M}$ obtained from the instance. This count is performed by showing that such walks can be encoded with a small number of bits. It is enough to count walks in which every hyperedge is repeated at least twice, and the crux of the argument is that the second time we see a hyperedge we can encode that hyperedge in a compact way. A naive way of doing that would point back to the previous step in the walk in which that hyperedge appeared, and this costs $\log \ell$ bits where ℓ is the length of the walk. To obtain the right result, however, repeated hyperedges have to be represented with an amortized constant number of bits per occurrence. The argument of Bordenave, Lalarge and Massoulié [5] relies on the assumption, which is true with high probability, that the graph is "tangle-free," meaning that small subgraphs have at most one cycle. We have to work with a looser notion of tangle-free hypergraph in order to prove that it holds with high probability, but we are still able to obtain the desired bound.

From spectral bounds to algorithms

It is clear that an algorithm certifying tight bounds on Equation (1) for the matrix M obtained from k-XOR instances can be used for strong refutations. Instead, to obtain Theorem 3 additional ideas are needed.

Our starting point is the local-to-global rounding paradigm of [3]. As it is often the case, the odd settings are significantly more challenging than the regimes with k even. Hence consider first a 2-XOR random instance $\boldsymbol{\mathcal{I}}$. Up to the signs of the clauses, this may be represented as a graph $\mathbf{G}$ over n vertices. Now, for a distribution ν over assignments, one may define the local and global correlations as

$$\mathrm{LC}_{\mathbf{G}}(\nu) = \mathop{\mathbb{E}}_{(\mathbf{a},\mathbf{b})\sim E(\mathbf{G})} \left|\mathrm{Cov}_{\nu}\left(\mathbf{x}^{\mathbf{a}}, \mathbf{x}^{\mathbf{b}}\right)\right|$$
$$\mathrm{GC}(\nu) = \mathop{\mathbb{E}}_{(\mathbf{a},\mathbf{b})\sim [n]\times[n]} \left|\mathrm{Cov}_{\nu}\left(\mathbf{x}^{\mathbf{a}}, \mathbf{x}^{\mathbf{b}}\right)\right| .$$

If the local correlation is bounded by ϵ, it is possible to obtain an assignment with value $\mathrm{Opt}_{\boldsymbol{I}} - O(\epsilon)$ simply looking at the *first moment* of ν. Moreover, one can always transform ν into a distribution with small global correlation in polynomial time.

With these observations, the argument of [3] comes down to: *(i)* bounding the difference between local and global correlation in terms of the spectral radius $\rho_{\mathbf{G}}$ of the centered adjacency matrix of the graph $\mathbf{G}$, *(ii)* showing that one can always find, in time $n^{O(1/\epsilon^2)}$, a degree $O(1)$ pseudo-distribution over the hypercube with global correlation at most ϵ. As we only required low-degree moments to obtain the desired assignment, the argument goes through in this case as well.

To combine this approach with the bounds previously illustrated and extend the argument to random k-XOR instances with $m \geq \Omega(n^{k/2}/\epsilon^2)$ clauses, we need to introduce two novel ingredients. *First*, we need new notions of local and global correlations which difference can be bounded studying the matrix $\mathbf{M}$ arising from the instance. *Second*, we need to bound this difference not in term of the eigenvalues of $\mathbf{M}$ but rather in terms of Equation (1).

A careful Cauchy-Schwarz application allows us to formulate notions of local and global correlations in terms of $\mathbf{M}$. Its squaring step, further allows us to get rid of absolute values, thus providing an avenue to bound the difference between local and global correlation in terms of $\max_{x\in\{\pm 1\}^n} x^{\mathsf{T}}\mathbf{M}x$.

Finally, since the adversarial perturbations in Theorem 3 cannot alter the "hypergraph walks" required to prove our bound, we are able to generalize our result to these settings.

1.3 Perspective

Several results on the average-case complexity of Sum-of-Square relaxations rely on proving that sparse matrices with non-independent entries are "quasirandom" in an appropriate sense. We have introduced a new approach to prove results of this form, which applies to very sparse matrices that have only a constant expected number of non-zero entries per row and per column. We hope that our ideas will find further application, for example to the context of semi-random instances of constraint satisfaction problems [14] or of higher-degree Sum-of-Square relaxations of random constraint satisfaction problems [19, 21].

Our theory could also be useful to study problems on random weighted graphs.

Our certificates prove certain PSD inequalities, and can be seen as Semidefinite Duals of certain Sum-of-Squares relaxations, but the computation of the certificate only requires an eigenvalue computation of a certain matrix, and does not require the solution of an SDP. There might be other ways to apply our theory so that one uses SDP relaxations only in the analysis, but the algorithm itself is purely spectral.

1.4 Organization

In the rest of the paper we first introduce preliminary notions, including those of CSPs and strong refutation, then present a proof of our generalized Ihara-Bass formula. We show the proofs of our main Theorems in the full version of the paper.

2 Preliminaries

We introduce some notation, useful facts and needed preliminary notions. We denote random variables in **bold**. We use lower case letters $a, b, c, d, \ldots$ to denote indices or scalars (the use willl be clear from context). We use the greek letters α, β, η to denote multi-indices. The cardinality of a multi-index α is $|\alpha|$. The i-th index in α is $\alpha(i)$. We may thus write a monomial (with coefficient c) in indeterminates $x_1, \ldots, x_n$ as $c \cdot x^\alpha$. For two multi indices $\alpha, \beta \in [n]^k$ we denote by (α, β) the multi-index obtained concatenating α and β. Multi-indices $\alpha, \beta \in [n]^k$ satisfy $\alpha = \beta$ if at each position the corresponding indices are identical. We use $S(\alpha)$ to denote the unordered multi-set of indices in α. We use n to denote our ambient dimension. For functions $f, g : \mathbb{R} \to \mathbb{R}$ we write $f = o(g)$ and $g = \omega(f)$ if $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

Matrices

For a matrix $M \in \mathbb{R}^{n\times n}$, we denote by $\lambda_1(M) \geq \ldots \geq \lambda_n(M)$ its eigenvalues. Then $\rho(M) := \max_i |\lambda_i(M)|$ is the spectral radius of M. W When the context is clear we simply write $\lambda_1, \ldots, \lambda_n$. The spectral radius of a matrix satisfies the following inequality.

▶ **Lemma 5** (Gelfand's Formula). *Let $M \in \mathbb{R}^{n\times n}$ and let $\|\cdot\|_*$ be a norm. Then for any positive integer z*

$$\rho(M) \leq \|M^z\|_*^{1/z} .$$

We write $\|M\|$ for the spectral norm of a matrix M, $\|M\|_{\mathrm{F}}$ for its Frobenius norm $\|M\|_{\infty\to 1} := \max_{x,y\in\{\pm 1\}^n} \langle M, xy^{\mathsf{T}}\rangle$ and $\|M\|_{\max} := \max_{ij} \left|M_{ij}\right|$. Furthermore, we let

$$\|M\|_{\mathrm{Gr}} = \max\{\langle M, X\rangle \mid X \succeq 0, X_{ii} \leq 1, \forall i \in [n]\} .$$

We denote by $|M|$ the matrix with entries $(|M|)_{ij} := \left|M_{ij}\right|$. We write Id_t for the t-by-t identity matrix, $\mathbf{0}$ for the zero matrix and J for the all-ones matrix.

Graphs

For a graph G, $V(G)$ and $E(G)$ denotes respectively its set of vertices and edges. $\vec{E}(G) := \{(u, v) : u \neq v \in V(G), uv \in E(G)\}$ is the set of all its ordered pairs such that $\{u, v\} \in E(G)$. For $e \in \vec{E}(G)$, $s(e)$ and $t(e)$ are respectively the source and target of the oriented edge. We write e^{-1} for its inverse. We also write K_n for the complete graph over n vertices. For a graph G with n vertices, we write $A(G) \in \mathbb{R}^{n\times n}$ for its adjacency matrix. For a vertex $v \in V(G)$, we denote by $\deg_G(v)$ its degree. We denote by $N_{G,t}(v)$ the set of vertices in G at distance t from v. We and drop the subscript G when the context is clear. If the graph G is weighted with weights given by $w : V(G) \times V(G) \to \mathbb{R}$, then $A_{uv} = w(\{uv\})$. If $e \neq E(G)$, then we assume $w(e) = 0$. A walk W in a graph G is a sequence of vertices $(v_1, \ldots, v_{z+1})$. A walk $v_1, \ldots, v_{z+1}$ is said to be non-backtracking if for any $i \leq z - 1$, $v_i \neq v_{i+2}$.

2.1 CSPs, k-XOR and strong refutations

k-XOR

A random k-XOR instance $\boldsymbol{\mathcal{I}}$ with n variables and $p\binom{n}{k}(1 \pm o(1))$ clauses can be generated by picking a random symmetric tensor $\mathbf{T}$, with independent entries, such that $\mathbf{T}_\alpha = 0$ if the indices in the multi-index $\alpha \in [n]^k$ are not distinct and otherwise:

$$\mathbf{T}_\alpha = \begin{cases} 0 & \text{with probability } 1-p\,, \\ +1 & \text{with probability } p/2\,, \\ -1 & \text{with probability } p/2\,. \end{cases}$$

We denote by m the exact number of clauses in the instance. Then $\boldsymbol{\mathcal{I}}$ consists of the m k-XOR predicates $k\text{-XOR}(\alpha) = \frac{1-x^\alpha(-\mathbf{T})_\alpha}{2}$ where $\mathbf{T}_\alpha$ is non-zero. We use $\mathcal{F}_{k\text{-XOR}(n,p)}$ to denote such distribution and $\boldsymbol{\mathcal{I}} \sim \mathcal{F}_{k\text{-XOR}(n,p)}$ to denote a random instance. We let $\mathrm{Val}_{\boldsymbol{\mathcal{I}}}(x)$ be the fraction of constrained satisfied by the assignment $x \in \{\pm 1\}^n$ and $\mathrm{Opt}_{\boldsymbol{\mathcal{I}}} := \max_{x\in\{\pm1\}^n} \mathrm{Val}_{\boldsymbol{\mathcal{I}}}(x)$. For any assignment $x \in \{\pm 1\}^n$ we have

$$\mathrm{Val}_{\boldsymbol{\mathcal{I}}}(x) = \frac{1}{2} + \frac{1}{m(\boldsymbol{\mathcal{I}})} \sum_{\alpha\in[n]^k} \frac{x^\alpha \mathbf{T}_\alpha}{2}\,.$$

Notice that since m will be $(1 \pm o(1))p\binom{n}{k}$ with overwhelming probability, we blur the distinction between these parameters. Then the max k-XOR problem is that of finding an assignment with value

$$\max_{x\in\{\pm1\}^n} \sum_{\alpha\in[n]^k} \mathbf{T}_\alpha x^\alpha\,. \tag{2}$$

This is captured by the following proposition.

▶ **Proposition 6.** *Let $\boldsymbol{\mathcal{I}} \sim \mathcal{F}_{k\text{-}XOR(n,p)}$ and let $\mathbf{T}$ be the associated k-th order tensor. Then with overwhelming probability*

$$\mathrm{Opt}_{\boldsymbol{\mathcal{I}}} \le \frac{1}{2} + (1+o(1))\left(\binom{n}{k}\cdot p\right)^{-1} \cdot \sum_{\alpha\in[n]^k} \mathbf{T}_\alpha x^\alpha\,.$$

Throughout the paper we assume k to be an *odd* integer as for the even case sharp refutation algorithms are already known (e.g see [2]).

A random k-XOR instance $\boldsymbol{\mathcal{I}}$ with n variables and exactly m clauses can be generated by picking m times a clause at random out of the $\binom{n}{k}$ possible k-XOR-clause. It is possible to show that a refutation algorithm for $\boldsymbol{\mathcal{I}} \sim \mathcal{F}_{k\text{-XOR}(n,p)}$ can also be used for refutation of k-XOR instances sampled through this second process. For this reason, we blur the distinction between these two processes. We direct the reader interested in a formal reduction to [2] (Appendix D).

CSPs

Given a predicate $P : \{-1,1\}^k \to \{0,1\}$, an instance $\mathcal{I}$ of the CSP(P) problem over variables $x_1,\dots,x_n$ is a multi-set of pairs (c,α) representing constraints of the form $P(c\circ x^\alpha) := P(c_1 x^{\alpha(1)},\dots,c_k x^{\alpha(k)}) = 1$ where $\alpha \in [n]^k$ is the scope and $c \in \{\pm1\}^k$ is the negation pattern. We can represent the predicate P as a multi-linear polynomial of degree k in indeterminates $c_1 x^{\alpha(1)},\dots,c_k x^{\alpha(k)}$,

$$P(c\circ x^\alpha) = \sum_{d\le k} P_d(c\circ x^\alpha)\,,$$

where P_d denotes the degree d part of the predicate. In particular $P_0 := P_0(c\circ x^\alpha)$ denotes the constant part of the polynomial, which does not depend on the assignment.

The fraction of all possible assignments that satisfy P is given by $\mathbb{E}_{\mathbf{z} \overset{u.a.r}{\sim} \{\pm 1\}^k}[P(\mathbf{z})]$. For any assignment $x \in \{\pm 1\}^n$ and an instance $\mathcal{I}$ over m constraints we have

$$\mathrm{Val}_{\mathcal{I}}(x) = \frac{1}{m} \sum_{(c,\alpha)\in\mathcal{I}} P(c \circ x^{\alpha})$$

$$\text{and} \quad \mathrm{Opt}_{\mathcal{I}} = \max_{x\in\{\pm 1\}^n} \mathrm{Val}_{\mathcal{I}}(x) \ .$$

A *random* CSP(P) instance $\boldsymbol{\mathcal{I}}$ with $(1 + o(1))m = p \cdot 2^k \cdot n^k$ constraints can be generated as follows:

- **(i)** Pick independently with probability p each pair $(\mathbf{c}, \boldsymbol{\alpha})$ where $\mathbf{c}$ is a random negation pattern from $\{-1, +1\}^k$ and α is a multi-index from $[n]^k$,
- **(ii)** For each such pair $(\mathbf{c}, \boldsymbol{\alpha})$ add the constraint $P(\mathbf{c} \circ x^{\boldsymbol{\alpha}}) = 1$ to $\boldsymbol{\mathcal{I}}$.

Notice that we do not rule out predicates with same multi-index but different negation pattern as multi-indices in which an index appears multple time. We also do not assume P to be symmetric. We denote such distribution by $\mathcal{F}_{\mathrm{CSP(P)}}(n, p)$.

As in the case of k-XOR a random CSP(P) instance $\boldsymbol{\mathcal{I}}$ with n variables and exactly m clauses can be generated by picking m times a clause and a negation pattern at random. Again it is possible to show that a refutation algorithm for $\boldsymbol{\mathcal{I}} \sim \mathcal{F}_{\mathrm{CSP(P)}}(n, p)$ can also be used for refutation of instances sampled through this second process (see Appendix D in [2]).

Refutation and certification

We say that $\mathcal{A}$ is a δ-*refutation algorithm* for random CSP(P) if $\mathcal{A}$ has the following properties:

- **(i)** on all instances $\mathcal{I}$ the output of $\mathcal{A}$ si either $\mathrm{Opt}_{\mathcal{I}} \leq 1 - \delta$ or "fail",
- **(ii)** if $\mathrm{Opt}_{\mathcal{I}} > 1 - \delta$ then $\mathcal{A}$ *never* outputs $\mathrm{Opt}_{\mathcal{I}} \leq 1 - \delta$.

More generally, for an set of possible inputs $\mathcal{S}$ and a property p over instances in $\mathcal{S}$, we say that an algorithm $\mathcal{A}$ *certifies* p if:

- **(i)** on all inputs $\mathcal{I} \in \mathcal{S}$ the output of $\mathcal{A}$ is either "$\mathcal{I}$ satisfies p" or "fail",
- **(ii)** if $\mathcal{I} \in \mathcal{S}$ does not satisfy p then $\mathcal{A}$ *never* outputs "$\mathcal{I}$ satisfies p".

In the context of random CSP(P) (and hence k-XOR), a ***strong refutation*** is a δ-refutation for $1 - \delta \leq \mathbb{E}_{\mathbf{x} \overset{u.a.r}{\sim} \{\pm 1\}^k}[P(\mathbf{x})] + o(1)$.

3 A generalized Ihara-Bass formula

In this section we present an extension of the Ihara-Bass theorem (see [16] and references therein) to arbitrary real symmetric matrices. We remark that our extension differs from the one in [7].

Throughout the section we assume to be given a *symmetric* matrix $A \in \mathbb{R}^{n\times n}$ with $2m$ non-zero entries and zeroed diagonal. We use the following notation. We will use letters u, v to denote indices in $[n]$ and e, f for indices in $[2m]$. We conveniently think of A as the adjacency matrix of a weighted undirected graph G with n vertices and $2m$ oriented edges. Then $uv \in E(G)$ if $A_{uv} \neq 0$, moreover then the inverse edge vu is also in $E(G)$ since $A_{uv} = A_{vu}$ by definition. Recall for an edge $e \in E(G)$ we write e^{-1} for its inverse and for a vertex $v \in V(G)$ we write $N^+(v)$ (respectively $N^-(v)$) for its set of outgoing (resp. incoming) oriented edges in G. We write $\sigma_{uv} = \mathrm{sign}(A_{uv})$. To reason about the spectrum of A, we introduce several matrices: the diagonal matrices

$$D(A) \in \mathbb{R}^{n\times n}, \quad \text{with } D_{uv}(A) = \begin{cases} \sum_w |A_{uw}| & u = v \\ 0 & \text{otherwise.} \end{cases}$$

$$Q(A) \in \mathbb{R}^{m\times m}, \quad \text{with } Q_{ef}(A) = \begin{cases} |A_e| & e = f \\ 0 & \text{otherwise.} \end{cases}$$

the block matrices

$$J(A) = \begin{pmatrix} 0 & \mathrm{Id}_m \\ \mathrm{Id}_m & 0 \end{pmatrix} \in \mathbb{R}^{2m\times 2m}$$

$$L(A) = \begin{pmatrix} 0 & Q(A) \\ Q(A) & 0 \end{pmatrix} \in \mathbb{R}^{2m\times 2m}$$

and the source, target and non-backtracking matrices

$$S(A) \in \mathbb{R}^{n\times 2m}, \quad \text{with } S_{ue}(A) = \begin{cases} \sigma_{uv}\sqrt{|A_{uv}|} & \text{if } u \text{ is the source of } e = uv \text{ and } u < v \\ \sqrt{|A_{uv}|} & \text{if } u \text{ is the source of } e = uv \text{ and } u > v \\ 0 & \text{otherwise.} \end{cases}$$

$$T(A) \in \mathbb{R}^{n\times 2m}, \quad \text{with } T_{ue}(A) = \begin{cases} \sigma_{uv}\sqrt{|A_{uv}|} & \text{if } u \text{ is the target of } e = vu \text{ and } u < v \\ \sqrt{|A_{uv}|} & \text{if } u \text{ is the target of } e = vu \text{ and } u > v \\ 0 & \text{otherwise.} \end{cases}$$

$$B(A) \in \mathbb{R}^{2m\times 2m}, \quad \text{with } B_{ef}(A) = \begin{cases} \sigma_e\sigma_f\sqrt{|A_eA_f|} & \text{if } ef \text{ is a non-backtracking walk} \\ & e = uv, \text{ f}{=}vw \text{ and } v < u, w \\ \sigma_e\sqrt{|A_eA_f|} & \text{if } ef \text{ is a non-backtracking walk} \\ & e = uv, \text{ f}{=}vw \text{ and } w < v < u \\ \sigma_f\sqrt{|A_eA_f|} & \text{if } ef \text{ is a non-backtracking walk} \\ & e = uv, \text{ f}{=}vw \text{ and } u < v < w \\ \sqrt{|A_eA_f|} & \\ & \text{if } ef \text{ is a non-backtracking walk} \\ & e = uv, \text{ f}{=}vw \text{ and } v > u, w \\ 0 & \text{otherwise.} \end{cases}$$

When the context is clear we simply write B for $B(A)$ (analogously for the other matrices). To gain intuition on these linear maps, it is instructive to consider the case when A is the adjacency matrix of an unweighted graph G. Then D is the degree diagonal matrix with $D_{uu} = deg_G(u)$, $L = J$ and B corresponds to the non-backtracking matrix of G.

Throughout the other sections of the paper, for a given non-backtracking matrix $B \in \mathbb{R}^{2m\times 2m}$, we will consider the related extension matrix $B^* \in \mathbb{R}^{2n^2\times 2n^2}$ with entries

$$B^*_{ef} = \begin{cases} B_{ef} & \text{if } e, f \in E(G) \\ 0 & \text{otherwise.} \end{cases}$$

For simplicity of the notation, we will often denote B^* simply by B. The context will always be clarified by the ambient dimension. We can now state the main result of the section.

▶ **Theorem 7** (Generalized Ihara-Bass Theorem). *Let n, m be integers and let $A \in \mathbb{R}^{n\times n}$ be a symmetric matrix with m non-zero entries, all off-diagonal. Let B, L, J, D defined as above. Then, for any $u \in \mathbb{R}$,*

$$\det\left(\mathrm{Id}_{2m} - u(B + L - J)\right) = (1-u^2)^{m-n}\left(\mathrm{Id}_n - uA + u^2 D - u^2\mathrm{Id}_n\right) .$$

Our proof of Theorem 7 closely resembles the proof of Bass [4]. We first observe that the matrices above satisfy several useful identities, than tackle the theorem.

▶ **Lemma 8.** *Using the definitions above:*

i) $SJ = T$ *and* $TJ = S$,
ii) $A = ST^{\mathsf{T}}$,
iii) $D = SS^{\mathsf{T}} = TT^{\mathsf{T}}$,
iv) $B + L = T^{\mathsf{T}}S$.

Proof. For i), notice that $SJ \in \mathbb{R}^{n\times 2m}$ and $SJ_{ue} = \langle S_{u,-}, J_{-,e}\rangle = S_{ue^{-1}} = T_{ue}$, where in the third step we used symmetry of A. A similar argument can be made to show $TJ = S$. For ii) observe that

$$A_{uv} = \langle S_{u,-}, T_{v,-}\rangle = \sum_e S_{ue}T_{ve}$$

which is nonzero only when $e = uv$. In that case, by definition $A_{uv} = \sigma_{uv}\,|A_{uv}| = S_{ue}T_{ve}$ since either $u < v$ or $u > v$. Consider now SS^{T}, the matrix is diagonal since each edge has at most one source vertex, then

$$(SS^{\mathsf{T}})_{uu} = \sum_e S_{ue}^2 = \sum_{v\in N^+(u)} |A_{uv}| = D_{uu} .$$

A symmetric derivation shows $D_{uu} = (TT^{\mathsf{T}})_{uu}$. It remains to prove iv). It is trivial to check that

$$(T^{\mathsf{T}}S)_{ee} = \langle T_{-,e}, S_{-,e}\rangle = \sum_u T_{ue}S_{ue} = 0 ,$$

since there are no self-loops in the graph. For distinct $e, f \in [2m]$

$$(T^{\mathsf{T}}S)_{ef} = \sum_u T_{ue}S_{uf} .$$

There is at most one non-zero element in the sum, corresponding to the case when u is the target vertex of e and the source of f, which means ef is a walk of length 2 in G. If ef is a non-backtracking walk (that is, $e \neq f^{-1}$) then $B_{ef} = (T^{\mathsf{T}}S)_{ef}$ and $L_{ef} = 0$. Conversely, if $e = f^{-1}$ then $B_{ef} = 0$ and $L_{ef} = (T^{\mathsf{T}}S)_{ef}$. Finally, signs can be checked case by case. ◀

We are now ready to prove Theorem 7.

Proof of Theorem 7. In the following identities all matrices are $(n + 2m)\times(n + 2m)$ block matrices where the first block has size $n \times n$. Let $u \in \mathbb{R}$,

$$\begin{aligned}
&\begin{pmatrix} \mathrm{Id}_n & 0 \\ T^{\mathsf{T}} & \mathrm{Id}_{2m}\end{pmatrix}\begin{pmatrix} \mathrm{Id}_n(1-u^2) & Su \\ 0 & \mathrm{Id}_{2m} - (B+L-J)u\end{pmatrix} \qquad (3)\\
&= \begin{pmatrix} \mathrm{Id}(1-u^2) & Su \\ T^{\mathsf{T}}(1-u^2) & T^{\mathsf{T}}Su + \mathrm{Id}_{2m} - (B+L-J)u\end{pmatrix}\\
&= \begin{pmatrix} \mathrm{Id}(1-u^2) & Su \\ T^{\mathsf{T}}(1-u^2) & \mathrm{Id}_{2m} + Ju\end{pmatrix} .
\end{aligned}$$

On the other hand

$$\begin{aligned}
&\begin{pmatrix} \mathrm{Id}_n(1-u^2) - Au + Du^2 & Su \\ 0 & \mathrm{Id}_{2m} + Ju \end{pmatrix} \begin{pmatrix} \mathrm{Id}_n & 0 \\ T^\mathsf{T} - S^\mathsf{T} u & \mathrm{Id}_{2m} \end{pmatrix} \\
&= \begin{pmatrix} \mathrm{Id}_n(1-u^2) - Au + Du^2 + ST^\mathsf{T} u - SS^\mathsf{T} u^2 & Su \\ T^\mathsf{T} - S^\mathsf{T} u + JT^\mathsf{T} u - JS^\mathsf{T} u^2 & \mathrm{Id}_{2m} + Ju \end{pmatrix} \\
&= \begin{pmatrix} \mathrm{Id}_n(1-u^2) & Su \\ T^\mathsf{T}(1-u^2) & \mathrm{Id}_{2m} + Ju \end{pmatrix}.
\end{aligned} \tag{4}$$

Putting Equation (3) and Equation (4) together and taking determinants we get

$$(1-u^2)^n \det\left(\mathrm{Id}_{2m} - (B + L - J)u\right) = \det\left(\mathrm{Id}_n(1-u^2) - Au + Du^2\right) \det\left(\mathrm{Id}_{2m} + Ju\right).$$

Now notice that

$$\mathrm{Id}_{2m} + Ju = \begin{pmatrix} \mathrm{Id}_m & \mathrm{Id}_m u \\ \mathrm{Id}_m u & \mathrm{Id}_m \end{pmatrix}$$

and thus $\det\left(\mathrm{Id}_{2m} + Ju\right) = (1-u^2)^m$. Rearranging, the result follows. ◀

3.1 Norm bounds via the Ihara-Bass formula

In this section we show how Theorem 7 can be used to study the spectrum of a real symmetric matrix A via the spectrum of related matrices. The central tool is the theorem below.

▶ **Theorem 9.** *Let $A \in \mathbb{R}^{n \times n}$ a symmetric matrix with zero diagonal. Let B, L, J, D be as defined in Section 3. Let $\lambda_{\min}$ be the smallest eigenvalue of the matrix $B + L - J \in \mathbb{R}^{2m \times 2m}$. Then for any $\lambda \leq \lambda_{\min}$*

$$A \succeq -|\lambda| \,\mathrm{Id}_n - |\lambda|^{-1} (D - \mathrm{Id}_n).$$

Proof. Let $\lambda_{\min}$ be the smallest real eigenvalue of $B + L - J$. By Theorem 7 we know -1 is a real eigenvalue of $B + L - J$ and thus $\lambda_{\min} \leq -1$. Moreover, for every $\lambda < \lambda_{\min}$ we have $\det\left(\mathrm{Id}_{2m} - \lambda^{-1}B + \lambda^{-1}L - \lambda^{-1}J\right) \neq 0$ otherwise λ would be an eigenvalue smaller than $\lambda_{\min}$. Define the matrix

$$M_\lambda := \mathrm{Id}_n - \lambda^{-1}A + \lambda^{-2}(D - \mathrm{Id}_n).$$

By the same reasoning as in Theorem 7, $\det(M_\lambda) \neq 0$ as long as $\lambda < \lambda_{\min}$. We make the stronger claim

$$\forall \lambda,\, \lambda_{\min} : M_\lambda \succ \mathbf{0}.$$

To prove the above claim, suppose toward a contradiction that $\lambda' < \lambda_{\min}$ is such that $M_{\lambda'}$ has a negative eigenvalue. Since M_λ tends to Id_n when $\lambda \to -\infty$, there is a value $\lambda_{PD} < \lambda'$ such that $M_{\lambda_{PD}}$ is strictly positive definite. Consider now the smallest eigenvalue of M_λ for values of λ in the range (λ_{PD}, λ'). The smallest eigenvalue of M_λ varies continuously with λ, it is positive for $\lambda = \lambda_{PD}$ and it is negative for $\lambda = \lambda'$, so it must be equal to zero for some $\lambda^* \leq \lambda' < \lambda_{\min}$. But this means that $\det(M_{\lambda^*}) = 0$ and so λ^* is an eigenvalue of $B + L - J$, which contradicts the definition of $\lambda_{\min}$. We have thus established our claim. Rearranging the result follows. ◀

A crucial consequence of Theorem 9 is that, exploiting the diagonal structure of the matrices D, Id_n one can bound the norm $\|A\|_{\infty\to 1}$ as a function of the smallest eigenvalue of the associated non-backtracking matrix.

▶ **Corollary 10.** *Let $A \in \mathbb{R}^{n\times n}$ a symmetric matrix with zero diagonal. Let $\lambda_{\min}$ and $\lambda'_{\min}$ be respectively the smallest eigenvalue of the matrix $B(A) + L(A) - J(A) \in \mathbb{R}^{2m\times 2m}$ and $B(-A) + L(A) - J(A) \in \mathbb{R}^{2m\times 2m}$, for B, L, J, D as defined in Section 3. Then, for any $\lambda \geq \max\left\{|\lambda_{\min}|, \left|\lambda'_{\min}\right|\right\}$,*

$$\|A\|_{\infty\to 1} \leq 2\,\mathrm{Tr}\left|\left(\lambda \mathrm{Id}_n + \lambda^{-1}(D(A) - \mathrm{Id}_n)\right)\right|$$

Proof. Define

$$R := \left|\lambda \mathrm{Id}_n + \lambda^{-1}(D(A) - \mathrm{Id}_n)\right| .$$

By Theorem 9 for any $x \in \{\pm 1\}^n$ we have $\left|x^\intercal A x\right| \leq \left|x^\intercal R x\right|$. For any $y \in \{\pm 1\}^n$ we can write

$$\begin{aligned} 2\left|x^\intercal A y\right| &\leq \left|(x+y)^\intercal A(x+y) - x^\intercal A x - y^\intercal A y\right| \\ &\leq \left|(x+y)^\intercal A(x+y)\right| + \left|x^\intercal A x\right| + \left|y^\intercal A y\right| . \end{aligned}$$

Now $x + y \in \{-2, 0, +2\}^n$ and thus

$$\left|(x+y)^\intercal A(x+y)\right| \leq 4 \max_{z\in\{\pm 1\}^n} z^\intercal R z ,$$

the result follows by definition of R. ◀

References

1. Vedat Levi Alev, Fernando Granha Jeronimo, and Madhur Tulsiani. Approximating constraint satisfaction problems on high-dimensional expanders. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 180–201. IEEE, 2019.
2. Sarah R. Allen, Ryan O'Donnell, and David Witmer. How to refute a random CSP. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 689–708, 2015. doi:10.1109/FOCS.2015.48.
3. Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *2011 ieee 52nd annual symposium on foundations of computer science*, pages 472–481. IEEE, 2011.
4. Hyman Bass. The Ihara-Selberg zeta function of a tree lattice. *International Journal of Mathematics*, 3(06):717–797, 1992.
5. Charles Bordenave, Marc Lelarge, and Laurent Massoulié. Non-backtracking spectrum of random graphs: Community detection and non-regular ramanujan graphs. In *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 1347–1357, 2015. doi:10.1109/FOCS.2015.86.
6. Siu On Chan. Approximation resistance from pairwise-independent subgroups. *Journal of the ACM (JACM)*, 63(3):1–32, 2016.
7. Zhou Fan and Andrea Montanari. How well do local algorithms solve semidefinite programs? In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 604–614. ACM, 2017.
8. Uriel Feige. Relations between average case complexity and approximation complexity. In *STOC 2002*, pages 534–543, 2002.
9. Uriel Feige. Refuting smoothed 3cnf formulas. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 407–417. IEEE, 2007.

10 Uriel Feige and Eran Ofek. Spectral techniques applied to sparse random graphs. *Random Struct. Algorithms*, 27(2):251–275, 2005.

11 Dimitris Fotakis, Michael Lampis, and Vangelis Th Paschos. Sub-exponential approximation schemes for csps: From dense to almost sparse. *arXiv preprint*, 2015. `arXiv:1507.04391`.

12 Joel Friedman and Andreas Goerdt. Recognizing more unsatisfiable random 3-SAT instances efficiently. In *Automata, Languages and Programming, 28th International Colloquium, ICALP 2001, Crete, Greece, July 8-12, 2001, Proceedings*, volume 2076 of *Lecture Notes in Computer Science*, pages 310–321. Springer, 2001.

13 Andreas Goerdt and Michael Krivelevich. Efficient recognition of random unsatisfiable k-SAT instances by spectral methods. In *STACS 2001, 18th Annual Symposium on Theoretical Aspects of Computer Science, Dresden, Germany, February 15-17, 2001, Proceedings*, volume 2010 of *Lecture Notes in Computer Science*, pages 294–304. Springer, 2001.

14 Venkatesan Guruswami, Pravesh K. Kothari, and Peter Manohar. Algorithms and certificates for boolean CSP refutation: "smoothed is no harder than random". *arXiv*, 2109.04415, 2021. `arXiv:2109.04415`.

15 Venkatesan Guruswami, Pravesh K Kothari, and Peter Manohar. Algorithms and certificates for boolean csp refutation: smoothed is no harder than random. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 678–689, 2022.

16 Matthew D Horton, HM Stark, and Audrey A Terras. What are zeta functions of graphs and what are they good for? *Contemporary Mathematics*, 415:173–190, 2006.

17 Pravesh K. Kothari. Personal communication, 2022.

18 Dana Moshkovitz and Ran Raz. Two-query pcp with subconstant error. *Journal of the ACM (JACM)*, 57(5):1–29, 2008.

19 Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random CSPs below the spectral threshold. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 121–131. ACM, 2017.

20 Yusuke Watanabe and Kenji Fukumizu. Graph zeta function in the bethe free energy and loopy belief propagation. *Advances in Neural Information Processing Systems*, 22, 2009.

21 Alexander S. Wein, Ahmed El Alaoui, and Cristopher Moore. The Kikuchi hierarchy and tensor PCA. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 1446–1468, 2019.

Towards Optimal Depth-Reductions for Algebraic Formulas

Hervé Fournier
Université Paris Cité, IMJ-PRG, France

Nutan Limaye
IT University Copenhagen, Denmark

Guillaume Malod
Université Paris Cité, IMJ-PRG, France

Srikanth Srinivasan
Aarhus University, Denmark

Sébastien Tavenas
Université Savoie Mont Blanc, CNRS, France

Abstract

Classical results of Brent, Kuck and Maruyama (IEEE Trans. Computers 1973) and Brent (JACM 1974) show that any algebraic formula of size s can be converted to one of depth $O(\log s)$ with only a polynomial blow-up in size. In this paper, we consider a fine-grained version of this result depending on the degree of the polynomial computed by the algebraic formula.

Given a homogeneous algebraic formula of size s computing a polynomial P of degree d, we show that P can also be computed by an (unbounded fan-in) algebraic formula of depth $O(\log d)$ and size $\text{poly}(s)$. Our proof shows that this result also holds in the highly restricted setting of monotone, non-commutative algebraic formulas.

This improves on previous results in the regime when d is small (i.e., $d = s^{o(1)}$). In particular, for the setting of $d = O(\log s)$, along with a result of Raz (STOC 2010, JACM 2013), our result implies the same depth reduction even for *inhomogeneous* formulas. This is particularly interesting in light of recent algebraic formula lower bounds, which work precisely in this "low-degree" and "low-depth" setting.

We also show that these results cannot be improved in the monotone setting, even for commutative formulas.

2012 ACM Subject Classification Theory of computation → Algebraic complexity theory

Keywords and phrases Algebraic formulas, depth-reduction

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.28

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2023/009/`

Funding *Nutan Limaye*: A part of this work was conducted during a research visit funded by Université Paris Cité in summer 2022.
Srikanth Srinivasan: Supported by start-up package from Aarhus University, and benefited greatly from a research visit sponsored by the Guest researchers faculty program at Université Paris Cité in summer 2022.
Sébastien Tavenas: This work is supported by ANR-22-CE48-0007.

1 Introduction

In this paper, we study a basic question regarding computational tradeoffs between two resources for the model of *algebraic formulas*.

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 28; pp. 28:1–28:19

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

An algebraic formula F for a multivariate polynomial $P(x_1, \ldots, x_n)$ is simply an algebraic expression for P made up of nested additions and multiplications. Equivalently, it can be defined as a rooted directed tree where the leaves are labelled by variables and internal nodes (or gates) compute either linear combinations or products of their children (a formal definition can be found in Section 2 below). Unless otherwise stated, we do not bound the number of children of a gate (in other words, we consider formulas of *unbounded fan-in*).

The two basic computational resources that describe the complexity of an algebraic formula F are its *size*, which is the number of leaves in the underlying tree, and its *depth*, which naturally is the depth of the tree. Polynomials[1] that have efficient (i.e., poly(n)-sized) algebraic formulas form the algebraic complexity class VF. Like its Boolean counterpart NC^1, this is a natural and important complexity class.

Tradeoffs between size and depth in the setting of formulas and related models of computation have been the focus of many previous works, starting from the early 1970s [28, 3, 4, 31, 25, 17, 21, 23, 1, 12, 29, 9, 11, 7, 15]. We describe a few such results here.

- In the Boolean setting, Spira [28] and independently Khrapchenko (see [32]) showed that any Boolean formula of size s can be converted to a Boolean formula of depth $O(\log s)$ while keeping the size bounded by $s^{O(1)}$. These results were replicated in the algebraic setting in results of Brent, Kuck and Marayuma [3] and Brent [4]. The constants involved in the bounds for the depth and the size were improved in many follow-up works [19, 20, 13, 5, 2].
- This question has also been studied for the more general model of *algebraic circuits,* where the underlying tree is replaced by a directed acyclic graph (DAG). A well-known result of Valiant, Skyum, Berkowitz and Rackoff [31] showed that an algebraic circuit of size s computing a polynomial of degree poly(s) can be converted to a circuit of depth[2] $O(\log s)$ and size poly(s). These results were also shown to hold for *multilinear*[3] circuits by Raz and Yehudayoff [23].
- The above results are known to be tight in various settings. In the *monotone* case[4], Shamir and Snir [25] showed the existence of an explicit polynomial P with a poly(n)-sized circuit such that any circuit of depth $o(\log n)$ for P is of superpolynomial size. Similar results were obtained in the multilinear case by Raz [21] and Chillara, Limaye and Srinivasan [8] (for multilinear circuits and formulas, respectively).
- Beginning with the work of Agrawal and Vinay [1], a recent line of work [12, 29, 9, 11, 7, 14] has shown that algebraic circuits and formulas can be converted to formulas of *constant depth* with a *sub-exponential* blow-up in size. In contrast, our focus in this paper is primarily on reducing depth as much as possible while keeping the size bounded by poly(s), as in the results listed previously.

The question

In this paper, we ask the question of whether stronger *depth-reduction* results can be proved given a bound d on the degree of the polynomial $P(x_1, \ldots, x_n)$ computed by the algebraic formula. In general, an algebraic formula of size s can compute a polynomial of degree at most s. When $d = s$ (or $d = s^{\Omega(1)}$), the above results imply that an algebraic formula F for P can be converted to another formula F' of depth $O(\log d)$ without significant blow-up in size. Does such a result hold for any d (or more specifically, when $d = s^{o(1)}$)?

---

[1] Strictly speaking, we should refer here to infinite sequences of polynomials, but we ignore this distinction.
[2] In the bounded fan-in case, this would be depth $O(\log^2 s)$ instead.
[3] A circuit or formula is multilinear if each of its gates computes a multilinear polynomial.
[4] where the underlying field is $\mathbb{R}$ and all constants are non-negative, so cancellations do not occur.

Note that this question only makes sense for algebraic formulas of unbounded fan-in. If the fan-in of each gate is bounded by a constant, then any formula of size s must have depth at least $\Omega(\log s)$ (and so, Brent, Kuck, and Marayuma's result [3] is optimal). However, in many settings (see e.g. the third motivation below), we want a finer analysis of the formula depth that can be achieved by formulas of unbounded fan-in.

Motivation

While the question is fairly natural in our opinion, there are also many concrete reasons that lead to this line of inquiry.

- It is easy to see from the proof of Valiant, Skyum, Berkowitz, and Rackoff [31] that any algebraic *circuit* can be depth-reduced to depth $O(\log d)$ with only a $\text{poly}(s, d)$ blow-up in size. So the natural generalization for small degrees is indeed true in the setting of algebraic circuits.
- It also follows from previous results that the depth bound of $O(\log s)$ can be improved when the degree is $d = o(\log s)$. This is due to a result of Raz [22]: any formula F for a homogeneous polynomial P of degree d can be converted to a *homogeneous* formula[5] F' efficiently. Further, it is easy to see that any homogeneous formula computing a polynomial of degree d has depth $O(d) = o(\log s)$. So, in this regime for the degree, standard depth-reduction results *can* be strengthened.
- Finally, very recent results in algebraic complexity [16] have suggested a way of proving lower bounds against *low-depth* algebraic formulas for computing *low-degree* polynomials, which naturally raises the question of obtaining the best possible depth-reduction results in this setting.
  More specifically, Limaye, Srinivasan, and Tavenas [16] showed how to prove lower bounds against algebraic formulas (and even circuits) of small depth. Their proof proceeds by converting an algebraic formula of size s and depth Δ to a homogeneous algebraic formula of size $\text{poly}(s)$ and depth $O(\Delta)$, and then proving lower bounds against homogeneous algebraic formulas of depth $O(\Delta)$. An important point regarding the first step is that it only works in the "low-degree setting" of $d = O(\log s/\log\log s)$. The second step proves lower bounds against homogeneous formulas of depth up to $O(\log\log d)$.
  To make this proof idea work for general (unbounded-depth) algebraic formulas, we would like to be able to homogenize and depth-reduce algebraic formulas as much as possible. The aforementioned result of Raz [22] already shows that we can homogenize algebraic formulas efficiently in the low-degree setting. So it is natural to investigate the best possible depth-reduction for homogeneous algebraic formulas in the low-degree setting.

Results

Our main result is a depth-reduction result for *homogeneous* formulas that efficiently reduces the depth to $O(\log d)$, matching what was already known for algebraic circuits by the result of [31].

▶ **Theorem 1** (Main Result). *Let F be a homogeneous algebraic formula of size s computing a polynomial P of degree $d \geqslant 2$. Then P is also computed by a homogeneous formula F' of size $\text{poly}(s)$ and depth $O(\log d)$. Moreover, if F is monotone and/or non-commutative, then so is F'.*

[5] A homogeneous formula is one where each gate computes a homogeneous polynomial. This means that the formula does not compute intermediate polynomials of degree larger than d.

Here, a *monotone* algebraic formula is one that does not exploit cancellations in any way, and a *non-commutative* formula describes a polynomial expression in a domain where the input variables do not commute when multiplied with each other (formal definitions are given in Section 2). These are both settings in which formula upper bounds are harder to prove, and hence the depth-reduction result in this setting implies the result in the standard setting. It can also be checked that the depth-reduction procedure above preserves other interesting properties of the formula, such as *multilinearity* and *set-multilinearity*.

Using the aforementioned result of Raz that allows us to homogenize algebraic formulas in the low-degree setting, we get the following depth-reduction even for *inhomogeneous* formulas.

▶ **Corollary 2.** *Let $d = O(\log n)$. Then a homogeneous polynomial P defined on n variables with degree $d \geqslant 2$ has an algebraic formula of size* $\mathrm{poly}(n)$ *if and only if it has an algebraic formula of depth $O(\log d)$ and size* $\mathrm{poly}(n)$.

In particular, this means that to prove superpolynomial lower bounds against general algebraic formulas in the low-degree setting, it suffices to prove such lower bounds against homogeneous algebraic formulas of depth $O(\log d)$. As far as we know, nothing below the trivial $O(d)$ bound was known before for such an implication. This brings us much closer to the regime of depths for which we have lower bounds [16].

The statements are even starker in the non-commutative setting, where it is a long-standing problem to prove separations between *Algebraic Branching Programs* (ABPs) and formulas. In recent work [30], it was shown how to prove such a result for depths that are $o(\sqrt{\log d})$. The results of this paper show that it suffices to prove such a result for depth $O(\log d)$.[6]

Finally, we also show that our results cannot be improved asymptotically in terms of depth, unless we use techniques that exploit cancellations in some way.

▶ **Theorem 3** (Lower Bound). *Let n and $d = d(n)$ be growing parameters such that $d(n) \leqslant \sqrt{n}$. Then there is a monotone algebraic formula F of size at most n and depth $O(\log d)$ computing a polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ of degree at most d such that any monotone formula F' of depth $o(\log d)$ computing P must have size $n^{\omega(1)}$.*

It should be noted that a well-known result of Gupta, Kamath, Kayal and Saptharishi [9] shows how to exploit cancellations to obtain better depth-reduction results. However, in general this does not reduce the depth of a given formula by more than a constant factor without incurring a significant blow-up in size.[7] As a result, we believe that the above result is a strong indication that our depth reduction result is tight up to a constant factor in the depth.

2 Preliminaries

Basic notation

Throughout, unless otherwise specified, we work with polynomials over a field $\mathbb{F}$. We will work with the multivariate ring of polynomials $\mathbb{F}[x_1, \ldots, x_n]$ or its *non-commutative* analog $\mathbb{F}\langle x_1, \ldots, x_n \rangle$.

[6] We note that Raz's result, though only stated for the commutative setting, works just as well in the non-commutative case.

[7] More precisely, the result of [9] shows how to convert a low-degree *homogeneous* depth-4 formula to an *inhomogeneous* depth-3 formula efficiently. In general, this can be used to reduce the depth of a small-depth formula by a multiplicative factor of 2.

2.1 Algebraic formulas

We start with some brief definitions and results related to algebraic formulas. For much more about this model, see the standard references [27, 24].

The model

An algebraic formula over the multivariate polynomial ring $\mathbb{F}[x_1, \ldots, x_n]$ is a rooted, directed tree with edges directed towards the root. Leaves are labelled by variables $x_1, \ldots, x_n$ or by the constant 1 and edges by non-zero field constants. Internal nodes (i.e., gates) by $+$ and $\times$ and compute linear combinations (based on the edge weights) or products of their children. We will assume, with loss of generality, that if a node α has for child a leaf labelled by 1, then α is a $+$-gate and that if a $+$-gate α has only children labelled by 1, then α is the output of the formula.[8] A *non-commutative* algebraic formula over the multivariate polynomial ring $\mathbb{F}\langle x_1, \ldots, x_n\rangle$ is defined similarly, with the additional assumption that the children of any $\times$-gate are linearly ordered, and the corresponding product is computed in this order.

Unless explicitly stated, the algebraic formulas we consider have *unbounded* fan-in (i.e., a gate can have any number of inputs). The *size* of F will denote the number of leaves,[9] the *depth* of F the longest leaf-to-root path. The *product-depth* and the *sum-depth* of F are defined to be the maximum number of product gates and sum gates encountered on a leaf-to-root path, respectively.

A *parse tree* of a formula F is a subformula of F which corresponds to the way a monomial is built in the evaluation of F. Parse trees of F can be defined inductively as follows:

- If F has a top $+$-gate, a parse tree of F is obtained by taking a parse tree of one of its children together with the corresponding edge to the root of F;
- If F has a top $\times$-gate, a parse tree of F is obtained by a taking a parse tree of each of its children, together with the incoming edges of the root of F;
- The only parse tree of a leaf is itself.

The polynomial computed by a parse tree is a single monomial, which is equal to the product of the variables labelling its leaves, multiplied by the product of the scalars labelling its edges. The polynomial computed by a formula F is easily seen to be the sum of the monomials computed by all its parse trees.

A formula F is called *monotone* if any monomial computed by a parse tree of F has a non-zero coefficient in the polynomial computed by F.

We now recall some well-known results from the literature regarding algebraic formulas. It should be noted that these results (specifically Theorems 4, 5 and 10 later on) are usually proved in the general setting of commutative formulas. However, it is easy to see that the proofs of these results carry over to the monotone, non-commutative setting without significant change.

Depth-reduction

Classical results [3, 4] show that any algebraic formula of small size can be simulated by one of small depth and not much larger size. Formally,

[8] This ensures that a formula can compute polynomials with a constant term but forbids using many arithmetic operations just to compute constants.

[9] This is within a constant factor of the number of gates, as long as each gate has fan-in at least 2 each (which is without loss of generality).

▶ **Theorem 4.** *Let F be a (non-commutative or commutative) algebraic formula of size s computing a polynomial P. Then there is an algebraic formula F' of size at most* $\mathrm{poly}(s)$ *and depth $\Delta = O(\log s)$ computing P. We may also assume that each gate in F' has fan-in 2. Furthermore, F is homogeneous and/or monotone, then so is F'.*

Homogeneity

Each gate in an algebraic formula has a *syntactic degree* defined in a natural way. Leaves labelled by the constant 1 have syntactic degree 0, leaves labelled with a variable have syntactic degree 1, $\times$-gates have a syntactic degree that is the sum of the syntactic degrees of their children, and $+$-gates have a syntactic degree that is equal to the largest of the syntactic degrees of their children. The syntactic degree of a formula is defined as the syntactic degree of its output. Notice that in a formula the syntactic degree of any gate is bounded by the syntactic degree of the formula.

We will further assume that no gate computes the zero polynomial.

A formula is *homogeneous* if each gate in the formula computes a homogeneous polynomial. Equivalently, in terms of syntactic degrees, this means that all the children of a sum gate have the same syntactic degree.

Raz [22] showed how to convert a possibly inhomogeneous formula F to a homogeneous formula with a relatively small blow-up in size.

▶ **Theorem 5.** *Let F be a (non-commutative or commutative) algebraic formula of size s and product-depth Δ computing a polynomial P such that all gates in F have fan-in 2. Then there is a* homogeneous *algebraic formula F' of size at most $O\left(s \cdot \binom{\Delta+d+1}{d}\right)$ and product-depth Δ computing P. In particular, if $\Delta = O(\log s)$ and $d = O(\log s)$, then the formula F' has size* $\mathrm{poly}(s)$.

3 Main result

Proof Overview

While the proof of the main result is fairly short and (in our opinion) clean, we add some remarks here to clarify why previous depth-reduction proofs are not applicable in our setting.

The first attempt in proving Theorem 1 would be to try to use the proof strategy behind Theorem 4. Here, we start with a formula F of size s and find a subformula G of size roughly $s/2$ rooted at some gate α of F. It is not hard to show that the polynomial computed by F can be written as

$$F = G \times H_1 + H_2$$

where H_1 and H_2 are also computed by formulas of size $s/2$.[10] We then apply induction to these three subformulas to get the result. Unfortunately, this strategy does not use the degree of the formula at all, and therefore only yields a formula of depth $O(\log s)$.

In the homogeneous setting, it is sometimes more natural to do induction on the degree of the underlying formula, in which case G and H_1 (in the decomposition above) would be subformulas of degree roughly $d/2$. We get the following recursion on the worst-case size of the depth-reduced version of F, which we denote by $T(s, d)$

$$T(s, d) \leqslant T(s_1, d/2) + T(s - s_1, d/2) + T(s - s_1, d)$$

[10] Here, for simplicity, we are assuming that F is a commutative formula. In the non-commutative setting, we would instead get $F = H_1' G H_1'' + H_2$.

where s_1 denotes the size of G. Unfortunately, in this case, the formulas H_1 and H_2 *could* have size nearly s, resulting in considerable size blow-up. Indeed, when s_1 is much smaller than s (say $s_1 = s^{o(1)}$), the above recursion only yields $T(s,d) = s^{O(\log d)}$, which is a superpolynomial size blow-up.

It may be possible to interleave recursions with respect to size and depth, but we were unable to make this work.

Another possible strategy could be to follow the work of [31] which produces *circuits* of the required depth. Unfortunately, the proof of [31] is a memoization procedure, which seems to yield circuits even when applied to a formula F. Turning the resulting circuit of depth $O(\log d)$ into a formula seems to increase the size to $s^{\Omega(\log d)}$.

The approach we take is somewhat more global than the recursive strategies outlined above. Our first motivating example is seemingly the worst-case example for depth-reduction: a *comb* of depth greater than d with alternating sums and products. More formally, a comb computes the following polynomial (up to identifying variables).

$$C(x) = x_1 + (x_2 \times (x_3 + (x_4 \times \cdots)))$$

Note that the above yields a formula of depth greater than d where d is the degree of the underlying polynomial. However, we observe that such a comb actually computes a polynomial with only a few monomials, and hence can be written trivially as a depth-2 $\sum\prod$-formula without much of a size blow-up.

Building on this observation, the overall strategy is to decompose the formula into a top part G, which is a (generalized) comb, whose leaves are subformulas of F to which we will apply the same procedure recursively. We then write G as a $\sum\prod$-formula, and replace its leaves with the depth-reduced versions of the subformulas. This gives the depth-reduced version of F.

The correct definition of G is crucial, and somewhat subtle (at least to us), but with the proper definitions in place, the proof goes through without much trouble.

3.1 Proof of Theorem 1

In this section we prove our main result (Theorem 1). We start by showing a simple depth-reduction result for the case of *skew formulas*. A formula is said to be skew if every multiplication gate in it has at most 1 non-trivial child (i.e., a non-leaf node).

We will say that a leaf in a skew-formula is a $+$-leaf if the parent of that leaf is a $+$ gate and a $\times$-leaf otherwise. We show that any skew formula can be converted efficiently into a depth-2 formula, i.e., a $\sum\prod$-formula.

▶ **Lemma 6.** *Let G be a skew formula with sum-depth δ, wherein all the gates have fan-in 2 and the leaves are labelled by distinct variables. Then the polynomial computed by G is a multilinear polynomial with at most 2^δ monomials. Moreover any variable labelling in G*

- *a $+$-leaf,*
- *or a $\times$-leaf whose sibling is a leaf*

appears in exactly one monomial. We will call them the non-duplicable *variables.*

Proof. We prove this by induction on the depth of the formula. The base case is when the depth is 0 or 1. In both cases, the statement trivially holds.

For the induction case, assume that the depth is at least 2. Suppose $G = G_1 + G_2$. The sum-depths of G_1 and G_2 are at most $\delta - 1$. Let f_1, f_2 be the multilinear polynomials computed by G_1, G_2, respectively. We know that the leaves of G are labelled with distinct

variables. Hence, G_1, G_2 have the same property and the variable sets labelling the leaves of G_1 and G_2 are disjoint. The depths of G_1 and G_2 are strictly smaller than the depth of G. By the induction hypothesis we have that f_1 and f_2 have at most $2^{\delta-1}$ monomials. Moreover, any non-duplicable variable in G_1 appears in at most one monomial in f_1. Similarly, any non-duplicable variable in G_2 appears in at most one monomial in f_2. Hence, the polynomial computed by G, i.e., $f_1 + f_2$, has at most 2^δ monomials and each non-duplicable variable appears in at most one monomial in it.

Suppose the top gate of G is a $\times$ gate. As G is a skew formula it is either $x \times G_1$ or $G_1 \times x$, where x is a variable. In particular since the depth of G is at least two, the variable x is duplicable. Let f_1 be the multilinear polynomial computed by G_1. By our assumption, the variable x does not appear in f_1. The depth of G_1 is strictly smaller than G. By the induction hypothesis we have that f_1 has at most 2^δ monomials and any non-duplicable variable in G_1 appears in at most one monomial. As x can distribute over the monomials of f_1, we have that the polynomial computed by G is multilinear with at most 2^δ monomials and any non-duplicable variable appears in at most one monomial.

(Note that this can also be seen using parse trees, since the polynomial computed is the sum of the monomials computed by the parse trees. The multilinearity is obvious. Note that parse trees do not really "branch" at multiplication gates because of the skewness and are therefore combs. To build a parse tree starting from the root we will have two choices for each addition gate we encounter on the path, and there are at most δ, so we get at most 2^δ parse trees. The only parse tree containing a given non-duplicable variable is defined by the path from the root to this leaf.) ◀

If a formula is homogeneous, it implies that for any gate α, the degree of the polynomial computed by α coincides with the syntactic degree d_α of α. Based on this remark, below we prove a stronger statement than Theorem 1. Specifically, Theorem 1 is stated for homogeneous formulas. But here, we show a depth-reduction for formulas for which the syntactic degree is small.

▶ **Theorem 7** (Refinement of Theorem 1). *Let F be an algebraic formula of size s and of syntactic degree $d_F \geqslant 2$. Then P is also computed by a formula F' of size* $\mathrm{poly}(s)$ *and depth $O(\log d_F)$. Moreover, if F is homogeneous, monotone and/or non-commutative, then so is F'.*

Proof. Let us start with a formula obtained from F after applying Theorem 4. That is, we will assume that we have a formula of size $\mathrm{poly}(s)$ such each gate in it has fan-in 2, sum-depth and product-depth are bounded by $O(\log s)$. For notational simplicity, from now on, F will refer to this new formula.

Let δ be a positive integer. For a formula G of syntactic degree $d_G \geqslant 1$ and sum-depth $\Delta(G)$ we define a potential function $\phi_\delta(G)$ as follows.

$$\begin{cases} \phi_{\delta,1}(G) = \lceil \log(d_G) \rceil \\ \phi_{\delta,2}(G) = \lceil \Delta(G)/\delta \rceil \end{cases}$$

and let

$$\phi_\delta(G) = \phi_{\delta,1}(G) + \phi_{\delta,2}(G).$$

We will show that the potential function bounds the depth of the depth-reduced formula that we will construct. We will also use it to bound the size of the resulting formula. Specifically, we prove the following lemma.

▶ **Lemma 8.** *Let δ be a positive integer. Any formula F of fan-in 2, syntactic degree $d \geqslant 1$, sum-depth Δ, and size s can be parallelized into a formula F' (of arbitrary fan-in) of product-depth at most $\phi_\delta(F)$ and size at most $s \cdot 2^{\delta \log(d)}$. Further, if F is homogeneous, monotone and/or non-commutative, so is F'.*

Since we ensured that the sum-depth is bounded by $O(\log s)$, taking $\delta = \left\lceil \frac{\log s}{\log d} \right\rceil$ and applying Lemma 8, we get that the final formula F' has size at most $\mathrm{poly}(s)$ and product-depth at most

$$\phi_\delta(F) = O\left(\lceil \log d \rceil + \left\lceil \log(s) \frac{\log d}{\log s} \right\rceil \right) = O(\log d).$$

By collapsing sum gates that feed into other sum gates, we see that the depth of F' can be assumed to be at most twice its product-depth, which is $O(\log d)$. This thus finishes the proof of the theorem. ◀

We now prove Lemma 8.

Proof of Lemma 8. For any gate α in F, let F_α denote the subformula rooted at α and d_α be its syntactic degree. We do the proof by induction on $\phi_\delta(F)$.

The base case $\phi_\delta(F) = 0$ trivially holds. Consider the following set of gates of F:

$$\mathcal{A} = \left\{ \alpha \mid \alpha \text{ is a not a leaf labelled 1 and } \phi_\delta(F_\alpha) < \phi_\delta(F) = \phi_\delta(F_{\mathrm{parent}(\alpha)}) \right\}.$$

For any gate α in $\mathcal{A}$, the induction hypothesis tells us that we can construct a formula F'_α of product-depth at most $\phi_\delta(F) - 1$ and size at most $s_\alpha \cdot 2^{\delta \log(d_\alpha)}$ computing the same polynomial as F_α.

Let us consider the formula G obtained by replacing these gates from $\mathcal{A}$ in F by leaves (labelled with distinct variables). Notice that for a product-gate β in F, at most one of its children has syntactic degree larger than $d_\beta/2$, where d_β is the syntactic degree of β. Consequently, G is a skew formula. The other child of β is a $\times$-leaf in G. Moreover, G has sum-depth at most δ (since $\phi_{\delta,2}$ strictly decreases for gates below).

Hence, we can use Lemma 6 to simplify G: we get that the polynomial computed by G is a multilinear polynomial in its leaves and has 2^δ monomials. We can then write G as a $\sum\prod$-formula G' such that each duplicable gate appears in at most 2^δ monomials and each non-duplicable gate in at most 1.

The new formula F' for F is obtained from G' by replacing each variable leaf, which corresponds to some gate α in F, by its depth-reduced version F'_α constructed using the induction hypothesis above.

The product-depth of F' is bounded by the product-depth of the gates $\alpha \in \mathcal{A}$ plus the product-depth of G, which is equal to 1 after rewriting it as a $\sum\prod$-formula. That is, the product-depth is at most $(\phi_\delta(F) - 1) + 1 = \phi_\delta(F)$. By construction if G does not contain leaves labelled by 1, then it is also the case for G', otherwise, G' still has at most one such leaf. The size of the resulting formula is bounded by

$$\sum_{\alpha \text{ non-duplicable}} \left(s_\alpha \cdot 2^{\delta \log(d_\alpha)} \right) + \left(2^\delta \cdot \sum_{\alpha \text{ duplicable}} \left(s_\alpha \cdot 2^{\delta \log(d_\alpha)} \right) \right) + \mathbb{1}_{\text{G' has a constant leaf}}.$$

Notice that if α is duplicable, it must be a $\times$-leaf with its sibling β not a leaf. This means that $d_\alpha \leqslant d_\beta \leqslant d$ since the syntactic degree is maximal at the root. Hence $d_\alpha \leqslant d/2$ so the contribution of duplicable gates is bounded by

$$\sum_{\alpha \text{ duplicable}} 2^\delta \left(s_\alpha \cdot 2^{\delta(\log(d)-1)} \right) \leqslant \sum_{\alpha \text{ duplicable}} \left(s_\alpha \cdot 2^{\delta \log(d)} \right).$$

The contribution of non-duplicable gates is bounded by

$$\sum_{\alpha \text{ non-duplicable}} s_\alpha \cdot 2^{\delta \log d}$$

since any gate in F has syntactic degree at most d. By the choice of $\mathcal{A}$, the subformulas F_α are disjoint so $\sum_{\alpha \in \mathcal{A}} s_\alpha \leqslant s$, with strict inequality if G' has a constant leaf. Hence the size of F' is bounded by $s \cdot 2^{\delta \log(d)}$.

Finally, it is straightforward to verify that the construction preserves homogeneity, monotonicity and/or non-commutativity. ◀

▶ Remark 9. It is easy to note that our depth reduction procedure does not increase the syntactic degree of the formula.

We also observe that putting Theorem 1 together with Theorems 4 and 5 immediately implies Corollary 2.

Proof of Corollary 2. Given a (possibly inhomogeneous) formula F of size $s = \text{poly}(n)$ computing a polynomial of degree $d = O(\log n)$, we first apply standard depth-reduction (Theorem 4) to get an equivalent formula F_1 of size $s_1 = \text{poly}(n)$ and depth $\Delta_1 = O(\log n)$ where each gate has fan-in 2. Applying Raz's homogenization theorem (Theorem 5) to F_1 yields an equivalent *homogeneous* formula F_2 of size $s_2 = \text{poly}(n)$ and depth $\Delta_2 = O(\log n)$. We can now apply Theorem 1 to F_2 to get an equivalent formula F' of size $s' = \text{poly}(n)$ and depth $\Delta' = O(\log d)$. ◀

3.2 Reducing the size blow-up

We note that the above strategy can be easily adapted to yield a small depth formula of size s' that is nearly linear in the size s of the original formula, at the expense of increasing the depth by a large constant.

The proof is nearly identical to the proof of Theorem 1 above. The new ingredient is a near-linear depth-reduction in the setting where there is no bound assumed on the degree of the above formula. More precisely, Bshouty, Cleve and Eberly [5] (see also the work of Bonet and Buss [2]) showed the following (we sketch Bonet and Buss' proof in Appendix A for completeness).

▶ **Theorem 10** (Depth-reduction with near-linear size)**.** *The following holds for any $\varepsilon > 0$. Let F be a (non-commutative or commutative) algebraic formula of size s computing a polynomial P. Then there is an algebraic formula F' of size at most $s^{1+\varepsilon}$ and depth $\Delta = 2^{O(1/\varepsilon)} \cdot \log s$ computing P. We may also assume that each gate in F' has fan-in 2. Furthermore, if F is homogeneous and/or monotone, then so is F'.*

Using the above result, we can prove the following improved version of our depth-reduction.

▶ **Theorem 11.** *Assume that F is a (commutative or non-commutative) formula of size s and syntactic degree $d \geqslant 1$ computing a polynomial P. Then P is also computed by a formula F'' of size at most $s^{1+\varepsilon}$ and depth $\Delta = 2^{O(1/\varepsilon)} \cdot \log d$. Furthermore, if F is a homogeneous and/or monotone formula, then so is F''.*

Proof. If $d^{4/\varepsilon} \geqslant s$, then the depth-reduction of Theorem 10 already gives a satisfactory solution. Indeed the size is bounded by $s^{1+\varepsilon}$ and the depth is bounded by

$$2^{O(1/\varepsilon)} \log s \leqslant \frac{4}{\varepsilon} 2^{O(1/\varepsilon)} \log d \leqslant 2^{O(1/\varepsilon)} \log d.$$

So we assume that $s > d^{4/\varepsilon}$. By first applying Theorem 10 (with $\varepsilon/2$ instead of ε), we obtain a formula F' of size at most $s^{1+\varepsilon/2}$, depth $\Delta' = 2^{O(1/\varepsilon)} \cdot \log s$, and fan-in 2 computing P.

Now, we apply Lemma 8, while setting $\delta = \lfloor (\varepsilon \log s)/(2 \log d) \rfloor$. Notice that since $\varepsilon \log s > 4 \log d$, it ensures that $\delta > (\varepsilon \log s)/(4 \log d) > 1$.

Then the strategy produces a formula F'' of product-depth at most

$$\phi_\delta(F') \leqslant \phi_{\delta,1}(F') + \phi_{\delta,2}(F') < \lceil \log d \rceil + 2^{O(1/\varepsilon)} \log s \frac{4 \log d}{\varepsilon \log s} = 2^{O(1/\varepsilon)} \log d$$

and size at most

$$s^{1+\varepsilon/2} \cdot 2^{\delta \log d} \leqslant s^{1+\varepsilon/2} s^{\varepsilon/2}.$$

This proves the theorem. ◀

3.3 Reducing the product fan-ins to 2

It is natural to ask if Theorem 1 can be proved while ensuring that the fan-in of each gate is bounded by 2, as in Theorem 1 and Theorem 10. This is not possible, as a formula of fan-in 2 and depth $O(\log d)$ can only compute polynomials on at most $\text{poly}(d)$ variables, while formulas of size s may have up to s variables. However, this does not rule out reducing the fan-in of the product gates to 2. Indeed, the *circuit* depth-reduction of [31] does exactly this. We show now that this can also be done for algebraic formulas with bounded syntactic degree.

▶ **Theorem 12.** *Let F be a (commutative or non-commutative) algebraic formula F of size s, depth Δ, and syntactic degree $d \geqslant 1$ computing a polynomial P. Then P can also be computed by a formula F' of size s and depth $\Delta' = O(\Delta + \log d)$ where each product gate of F' has fan-in 2. Furthermore, if F is a homogeneous and/or monotone formula, then so is F'.*

Plugging this in Theorem 7 gives a depth-reduction to formulas of depth $O(\log d)$ and size $\text{poly}(s)$ such that all product gates have fan-in at most 2. A similar result can be obtained with a smaller blow-up in size by combining this statement with Theorem 11.

Proof. It suffices to prove a weaker version of the above theorem where each product gate has fan-in at most 3. We can then replace each of the products of fan-in 3 by a tree of product gates of fan-in 2 and size 3. This has the effect of increasing the depth at most by a factor of 2, which does not affect the overall result.

So we will prove this slightly weaker version. In this setting, we will aim for a depth $\Delta' = \Delta + \log d$.

This is done by induction on the depth Δ of the formula. The case of $\Delta = 0$ is trivial. Let F be a formula of depth $\Delta > 0$ and syntactic degree d.

Assume that the output gate of F is a sum gate, and $F_1, \ldots, F_t$ are the subformulas of F of depth $\Delta - 1$. By definition, each F_i has syntactic degree at most d. Applying the induction hypothesis to each of the F_i yields a formula F'_i with product gates of fan-in at most 3. The formula F' can then be defined as the sum of these formulas.

Now we come to the main case, which is when the output gate of F is a product gate. Assume that $F_1, \ldots, F_t$ are the subformulas of F of depth $\Delta - 1$ in the order[11] that they appear in F. Let d_i denote the syntactic degree of F_i. Define

$$m = \min\{j \mid \sum_{i=1}^{j} d_i \geqslant d/2\}.$$

Let F_ℓ be the formula obtained from F by keeping only the subformulas $F_1, \ldots, F_{m-1}$, and F_r be the formula obtained by keeping only $F_{m+1}, \ldots, F_t$. We use the induction hypothesis on F_ℓ, F_m, and F_r to get formulas F'_ℓ, F'_m and F'_r. Finally, we set

$$F' = F'_\ell \times F'_m \times F'_r.$$

The size[12] of F' is the sum of the sizes of F'_ℓ, F'_m and F'_r, which is at most s by the induction hypothesis. Let Δ'_ℓ, Δ'_m and Δ'_r denote the depths of F'_ℓ, F'_m and F'_r respectively. The depth of F' is

$$1 + \max\{\Delta'_\ell, \Delta'_m, \Delta'_r\} \leqslant 1 + \max\{\Delta + \log(d/2), \Delta - 1 + \log d, \Delta + \log(d/2)\} = \Delta + \log d$$

where the second inequality uses the induction hypothesis, and the fact that F_ℓ and F_r have syntactic degree at most $d/2$ and F_m has depth at most $\Delta - 1$. ◀

4 Tightness

Given integers $k \geqslant 1$ and $r \geqslant 2$, we will define a polynomial $H^{(k,r)}$. Intuitively, we want to define this polynomial as a standard universal polynomial for formulas. It is composed of k-nested inner products, each one of size r. In the following we will drop the superscript in $H^{(k,r)}$ and write simply H instead.

The polynomial H will be defined over the set of $(2r)^k$ variables

$$\{x_{\sigma,\tau} \mid \sigma \in [2]^k, \tau \in [r]^k\}.$$

Let us define recursively polynomials $H_{u,v}$ for all $(u, v) \in [2]^{\leqslant k} \times [r]^{\leqslant k}$ such that $|u| = |v|$:

$$H_{u,v} = x_{u,v} \qquad \text{when } |u| = |v| = k$$

$$H_{u,v} = \sum_{a=1}^{r} H_{u1,va} H_{u2,va} \qquad \text{otherwise.}$$

The polynomial H is defined as the polynomial $H_{\varepsilon,\varepsilon}$. Note that H is a polynomial of degree $d = 2^k$ and has r^{d-1} monomials.

From its definition, H is computed by a monotone formula M of size $(2r)^k$ and depth $2k$, with a $+$-gate at the top, alternating layers of $+$-gates and $\times$-gates, with $+$-gates of fan-in r and $\times$-gates of fan-in 2, and leaves labelled with distinct variables.

For words u and v over the same alphabet, we write $u \sqsupset v$ if u is a prefix of v. There is a natural one-to-one correspondance between prefixes of words of $([r] \times [2])^k$ and nodes of M, which is the following. Let $\sigma = \sigma_1 \ldots \sigma_k \in [2]^k$, and $\tau = \tau_1 \ldots \tau_k \in [r]^k$. The word $\tau_1\sigma_1 \ldots \tau_k\sigma_k$ corresponds to a path from the root of M to the leaf labelled $x_{\sigma,\tau}$, while

[11] The order is important in the non-commutative setting.
[12] Recall that the size of a formula is the number of its leaves.

proper prefixes of $\tau_1\sigma_1 \dots \tau_k\sigma_k$ correspond to internal gates in M along this path. For $\ell < k$, $u = u_1 \dots u_\ell \in [2]^\ell$ and $v = v_1 \dots v_\ell \in [r]^\ell$, the node which corresponds to the word $v_1u_1 \dots v_\ell u_\ell$ is the $+$-gate of M computing $H_{u,v}$.

The polynomial H is easily seen to be set-multilinear with respect to the sets of variables $\{X_\sigma \mid \sigma \in [2]^k\}$ where $X_\sigma = \{x_{\sigma,\tau} \mid \tau \in [r]^k\}$. This means that each monomial has exactly one variable from each set X_σ.

▶ Remark 13. For $|u| = |v| = \ell \leqslant k$, $u \in [2]^\ell$ and $v \in [r]^\ell$, the polynomial $H^{(k,r)}_{u,v}$ is defined over the set of variables

$$X_{u,v} = \{x_{\sigma,\tau} \mid \sigma \in [2]^k,\ \tau \in [r]^k, u \sqsupset \sigma,\ v \sqsupset \tau\}$$

and is the polynomial $H^{(k-\ell,r)}$ (upto renaming of the variables). In particular, its degree is $d' = 2^{k-\ell}$, it has $r^{d'-1}$ monomials and it is set-multilinear with respect to $\{X_\sigma \mid \sigma \in [2]^k,\ u \sqsupset \sigma\}$.

Before proving hardness of H for small-depth monotone formulas, we need to show that the gates of a monotone formula computing H cannot compute too many monomials. This is proved in Lemma 15 below.

▶ **Proposition 14.** *Consider two variables $x_{\sigma,\tau}$ and $x_{\sigma',\tau'}$. If $x_{\sigma,\tau}x_{\sigma',\tau'}$ appears in a monomial of H, and if σ and σ' have a common prefix of length $\ell < k$, then τ and τ' have a common prefix of length $\ell + 1$.*

Proof. Observe that $x_{\sigma_1,\tau_1} \dots x_{\sigma_p,\tau_p}$ is a monomial of H if and only if these variables form the leaves of a parse tree of M. As observed above, the root-to-leaf path leading to the variable x_{σ_i,τ_i} in M is obtained by taking in turn the first letter of τ_i, the first letter of σ_i, the second letter of τ_i, etc.

If the product $x_{\sigma,\tau}x_{\sigma',\tau'}$ appears in a monomial of H, it must be possible to complete the union of the two paths, from the root to $x_{\sigma,\tau}$ and from the root to $x_{\sigma',\tau'}$, into a parse tree of the formula M.

If the longest common prefix of τ and τ' were of length at most ℓ, then the lowest common ancestor of $x_{\sigma,\tau}$ and $x_{\sigma',\tau'}$ in M would be a $+$-gate, which is not possible in a parse tree where each $+$-gate has a single child. ◀

▶ **Lemma 15.** *If F is a monotone formula which computes the polynomial H and if α is a gate of F of degree d_α, then the number of monomials of the polynomial computed at gate α is at most $r^{d_\alpha - 1}$.*

Proof. Since H has r^{d-1} monomials, the result is true when $d_\alpha = d$ by monotonicity. Assume now that $d_\alpha < d$.

Let

$$I = \{\sigma \in [2]^k \mid \text{some variable of } X_\sigma \text{ appears in } \alpha\}.$$

For $u \in [2]^{\leqslant k}$, let $I_u = \{\sigma \in [2]^k \mid u \sqsupset \sigma\}$. Let $\{u_1, \dots, u_p\}$ be the set of words w of minimal length such that $I_w \subseteq I$. Then I is the disjoint union $\bigcup_{\ell \in [p]} I_{u_\ell}$. Since $d_\alpha < d$, $I \neq [2]^k$ so no u_ℓ is the empty word.

Consider some $\ell \in [p]$. Let $\bar{u}_\ell$ be obtained by switching the last letter of u_ℓ. By minimality of the length of u_ℓ, we must have $I_{\bar{u}_\ell} \not\subseteq I$. Let σ be a word in $I_{\bar{u}_\ell} \setminus I$. Since F is monotone and computes a set-multilinear polynomial, it is a set-multilinear formula and therefore the polynomial computed at α must be multiplied by some variable $x_{\sigma,\tau}$. Let v_ℓ be the prefix of length $|u_\ell|$ of τ. Consider any variable $x_{\sigma',\tau'}$ appearing in α such that $u_\ell \sqsupset \sigma'$. Since the product $x_{\sigma,\tau}x_{\sigma',\tau'}$ must appear in a monomial of H by monotonicity, it must be that $v_\ell \sqsupset \tau'$ by Proposition 14.

Any monomial m in the polynomial computed in gate α can be written in a unique way $m = m_1 \cdots m_p$ with m_ℓ set-multilinear with respect to $\{X_\sigma \mid u_\ell \sqsupset \sigma\}$. By the above, m_ℓ is a monomial over the variables X_{u_ℓ,v_ℓ} of degree $|I_{u_\ell}|$. By monotonicity, it should be possible to complete leaves of M labelled with variables from m_ℓ into a parse tree of M appearing in H, which proves that m_ℓ is a monomial of H_{u_ℓ,v_ℓ}. There are at most $r^{|I_{u_\ell}|-1}$ such submonomials m_ℓ by Remark 13. It follows that the number of monomials of the polynomial computed in node α is at most

$$\prod_{\ell=1}^{p} r^{|I_{u_\ell}|-1} \leqslant r^{|I|-1} = r^{d_\alpha - 1}. \qquad \blacktriangleleft$$

We are ready to prove hardness of the polynomial H for monotone computation. We shall make use of the following "product lemma", which comes in different forms in e.g. [27, 10, 24].

▶ **Lemma 16.** *A degree-d homogeneous formula F of size s and product-depth Δ can be written as a sum of $O(s)$ polynomials, each of which is a product of $\Omega(\Delta d^{1/\Delta})$ many polynomials of positive degree. Moreover, each of these polynomials is computed by some gate in F.*

▶ **Proposition 17.** *If F is a monotone formula of product-depth $\Delta \leqslant \log d$ which computes H, then its size is at least $r^{\Omega(\Delta d^{1/\Delta})}$.*

Proof. Let $t = \Delta d^{1/\Delta}$. Since F is monotone, it is homogeneous and by Lemma 16 can be written as

$$F = \sum_{i=1}^{s'} \prod_{j=1}^{t_i} F_{i,j}$$

with $s' = O(s)$ and $t_i = \Omega(t)$ for all i, and $F_{i,j}$ is of degree at least 1 and computed by some gate in F.

Let $d_{i,j}$ be the degree of $F_{i,j}$. By Lemma 15, each $F_{i,j}$ computes at most $r^{d_{i,j}-1}$ monomials. The number of monomials of $\prod_{i=1}^{t_j} F_{i,j}$ is therefore bounded by

$$\prod_{j=1}^{t_i} r^{d_{i,j}-1} \leqslant r^{\sum_{j=1}^{t_i} d_{i,j} - t_i} \leqslant r^{d-t}$$

since $\sum_{j=1}^{t_i} d_{i,j} = d$. It follows that the number of monomials computed by F is at most $s \cdot r^{d-t}$. Since F computes H which has r^{d-1} monomials, we get $s = r^{\Omega(t)}$. $\blacktriangleleft$

We can now get Theorem 3 from the introduction, which is restated here for convenience.

▶ **Theorem 18.** *Let n and $d = d(n)$ be growing parameters such that $d(n) \leqslant \sqrt{n}$. Then there is a monotone algebraic formula F of size at most n and depth $O(\log d)$ computing a polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ of degree at most d such that any monotone formula F' of depth $o(\log d)$ computing P must have size $n^{\omega(1)}$.*

Proof of Theorem 3. Choose parameters $k(n)$ and $r(n)$ such that $P(n) := H^{(k,r)}$ has $\Theta(n)$ variables and degree $\Theta(d)$: let $k = \log d$ and $r = \frac{1}{2} n^{1/\log d}$. Condition $d(n) \leqslant \sqrt{n}$ ensures that $r \geqslant 2$. The polynomial P has a monotone formula of size $O(n)$ and depth $O(\log d)$. By Proposition 17, any monotone formula of product-depth $\Delta \leqslant \log d$ computing P has size

$$r^{\Omega(\Delta d^{1/\Delta})} = \left(\frac{1}{2} n^{1/\log d}\right)^{\Omega(\Delta d^{1/\Delta})} = (n/d)^{\left(\frac{\Delta d^{1/\Delta}}{\log d}\right)}$$

which is $n^{\Omega\left(\frac{\Delta d^{1/\Delta}}{\log d}\right)}$ using the hypothesis $d(n) \leqslant \sqrt{n}$. Since $\frac{\Delta d^{1/\Delta}}{\log d} \to +\infty$ when $\Delta = o(\log d)$ this bound is $n^{\omega(1)}$. $\blacktriangleleft$

5 Conclusion and Open questions

In this paper we investigated the possibility of reducing the depth of a formula of size s computing a polynomial of degree d to $O(\log d)$ while keeping the size $s^{O(1)}$.

We showed (Theorem 1) that we can do such a transformation when F is *homogeneous*. More generally, Theorem 7 states that we can achieve it as soon as the syntactic degree of F is polynomially bounded in d.

Structure inside VF

Let us consider a sequence of polynomials (f_n) whose number of variables and degree are bounded polynomially in n (such a family is usually called a p-family, see for example [6]). We can then consider three classes of such families:

- $\text{homF}[s(n)] = \{(f_n) \mid f_n \text{ is computed by a homogeneous formula of size } \text{poly}(s(n))\}$,
- $\text{lowSynDegF}[s(n)] = \{(f_n) \mid f_n \text{ is computed by a formula of size } \text{poly}(s(n)) \text{ and of syntactic degree } \text{poly}(\deg(f_n))\}$
- $\text{lowDepthF}[s(n)] = \{(f_n) \mid f_n \text{ is computed by a formula of size } \text{poly}(s(n)) \text{ and of depth } O(\log \deg(f_n))\}$.

Clearly, we have the inclusion $\text{homF}[s(n)] \subseteq \text{lowSynDegF}[s(n)]$. Also, in this paper, we have shown the inclusion $\text{lowSynDegF}[s(n)] \subseteq \text{lowDepthF}[s(n)]$. Consequently,

$$\text{homF}[\text{poly}(n)] \subseteq \text{lowSynDegF}[\text{poly}(n)] \subseteq \text{lowDepthF}[\text{poly}(n)] \subseteq \text{VF},$$

and we do not know if these inclusions are strict or not.

The complexity of the Elementary Symmetric Polynomials

A particularly interesting special case of the questions above comes from the example of the *Elementary Symmetric Polynomials*. Given parameters d, n with $d \leqslant n$, recall that the Elementary Symmetric polynomial $S_n^d(x_1, \ldots, x_n)$ is the sum of all the multilinear monomials of degree exactly d. A simple and elegant construction of Ben-Or (see [26]) shows that for any d, the polynomial S_n^d has an *inhomogeneous* formula of depth-3 and size $O(n^2)$. This puts this family of polynomials in the class $\text{lowDepthF}[\text{poly}(n)]$. Further, Shpilka and Wigderson [26, Theorem 5.3], showed that S_n^d has depth-6 formulas of syntactic degree at most $\text{poly}(d)$, putting it in the class $\text{lowSynDegF}[\text{poly}(n)]$.

However, as far as we know, there are no known $\text{poly}(n)$-sized homogeneous formulas for this family of polynomials.[13] In fact, under some further restrictions, Hrubeš and Yehudayoff showed [10] a superpolynomial homogeneous formula lower bound when $d = n/2$. Removing these restrictions would show a separation between $\text{homF}[\text{poly}(n)]$ and $\text{lowSynDegF}[\text{poly}(n)]$. On the other hand, if indeed the elementary symmetric polynomials have $\text{poly}(n)$-sized homogeneous formulas, then this can be used to argue[14] the same for any polynomial computed by a depth-3 formula of polynomial size, hinting at a possible collapse between $\text{homF}[\text{poly}(n)]$ and $\text{lowSynDegF}[\text{poly}(n)]$.

[13] A strong form of this was conjectured by Nisan and Wigderson [18], which was subsequently refuted by Hrubeš and Yehudayoff [10]. However, this still does not yield polynomial-sized homogeneous formulas for all elementary symmetric polynomials.

[14] see e.g. [16, Section III] for the standard argument

Lower bounds for higher-depth formulas

Due to the recent lower bound results of [16], we know that there is an explicit homogeneous polynomial $P(X)$ of degree d on n variables that cannot be computed by any formula of size $\text{poly}(n)$ and depth $\varepsilon \cdot \log\log d$, for some absolute constant $\varepsilon > 0$. It turns out that the polynomial P *is* computable by an *algebraic branching program* and therefore, lies in the complexity class called VBP.

It is known that VF is contained in VBP. However, we do not know whether this containment is strict or not. Our lower bound result helps us pose a refined version of this question. Specifically, it shows that if the lower bound from [16] can be improved from $\Omega(\log\log d)$ to $\omega(\log d)$, then we will have separated VF from VBP.

The fact that our depth reduction carries over to the non-commutative setting, makes a compelling case for revisiting the VF vs. VBP question in the non-commutative setting. Specifically, a recent result of [30] shows that there is an explicit non-commutative polynomial $P(X)$ of degree d on n variables that cannot be computed by any non-commutative formula of size $\text{poly}(n)$ and depth $\varepsilon \cdot \sqrt{\log d}$. So, improving the lower bound in this case from $\Omega(\sqrt{\log d})$ to $\omega(\log d)$ would separate VF from VBP in the non-commutative setting.

References

1 Manindra Agrawal and V. Vinay. Arithmetic circuits: A chasm at depth four. In *proceedings of Foundations of Computer Science (FOCS)*, pages 67–75, 2008. doi:10.1109/FOCS.2008.32.

2 Maria Luisa Bonet and Samuel R. Buss. Size-depth tradeoffs for boolean formulae. *Information Processing Letters*, 49(3):151–155, 1994. doi:10.1016/0020-0190(94)90093-0.

3 R. Brent, D. Kuck, and K. Maruyama. The parallel evaluation of arithmetic expressions without division. *IEEE Transactions on Computers*, C-22(5):532–534, 1973. doi:10.1109/T-C.1973.223757.

4 Richard P. Brent. The parallel evaluation of general arithmetic expressions. *Journal of the ACM*, 21(2):201–206, April 1974. doi:10.1145/321812.321815.

5 Nader H. Bshouty, Richard Cleve, and Wayne Eberly. Size-depth tradeoffs for algebraic formulas. *SIAM J. Comput.*, 24(4):682–705, 1995. doi:10.1137/S0097539792232586.

6 Peter Bürgisser. Cook's versus Valiant's hypothesis. *Theoretical Computer Science*, 235(1):71–88, 2000.

7 Suryajith Chillara, Mrinal Kumar, Ramprasad Saptharishi, and V. Vinay. The chasm at depth four, and tensor rank: Old results, new insights. *CoRR*, abs/1606.04200, 2016. arXiv:1606.04200.

8 Suryajith Chillara, Nutan Limaye, and Srikanth Srinivasan. Small-depth multilinear formula lower bounds for iterated matrix multiplication, with applications. In *STACS*, volume 96 of *LIPIcs*, pages 21:1–21:15. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2018.

9 Ankit Gupta, Pritish Kamath, Neeraj Kayal, and Ramprasad Saptharishi. Arithmetic circuits: A chasm at depth 3. *SIAM Journal of Computing*, 45(3):1064–1079, 2016. doi:10.1137/140957123.

10 Pavel Hrubeš and Amir Yehudayoff. Homogeneous formulas and symmetric polynomials. *Comput. Complexity*, 20(3):559–578, 2011. doi:10.1007/s00037-011-0007-3.

11 Neeraj Kayal, Chandan Saha, and Ramprasad Saptharishi. A super-polynomial lower bound for regular arithmetic formulas. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*, pages 146–153. ACM, 2014. doi:10.1145/2591796.2591847.

12 Pascal Koiran. Arithmetic circuits: The chasm at depth four gets wider. *Theor. Comput. Sci.*, 448:56–65, 2012. doi:10.1016/j.tcs.2012.03.041.

13 S Rao Kosaraju. Parallel evaluation of division-free arithmetic equations. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, pages 231–239, 1986.

14 Mrinal Kumar, Rafael Mendes de Oliveira, and Ramprasad Saptharishi. Towards optimal depth reductions for syntactically multilinear circuits. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, volume 132 of *LIPIcs*, pages 78:1–78:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.78.

15 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. *Electron. Colloquium Comput. Complex.*, TR21-081, 2021. URL: https://eccc.weizmann.ac.il/report/2021/081.

16 Nutan Limaye, Srikanth Srinivasan, and Sébastien Tavenas. Superpolynomial lower bounds against low-depth algebraic circuits. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 804–814. IEEE, 2021. doi:10.1109/FOCS52979.2021.00083.

17 Noam Nisan. Lower bounds for non-commutative computation. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 410–418, 1991.

18 Noam Nisan and Avi Wigderson. Lower bounds on arithmetic circuits via partial derivatives. *Computational Complexity*, 6(3):217–234, 1997. doi:10.1007/BF01294256.

19 Franco P. Preparata and David E. Muller. The time required to evaluate division-free arithmetic expressions. *Inf. Process. Lett.*, 3(5):144–146, 1975. doi:10.1016/0020-0190(75)90028-9.

20 Franco P. Preparata and David E. Muller. Efficient parallel evaluation of boolean expression. *IEEE Trans. Computers*, 25(5):548–549, 1976. doi:10.1109/TC.1976.1674647.

21 Ran Raz. Separation of multilinear circuit and formula size. *Theory of Computing*, 2(1):121–135, 2006. doi:10.4086/toc.2006.v002a006.

22 Ran Raz. Tensor-rank and lower bounds for arithmetic formulas. *Journal of the ACM*, 60(6):40:1–40:15, 2013. doi:10.1145/2535928.

23 Ran Raz and Amir Yehudayoff. Balancing syntactically multilinear arithmetic circuits. *Computational Complexity*, 17(4):515–535, 2008. doi:10.1007/s00037-008-0254-0.

24 Ramprasad Saptharishi. A survey of lower bounds in arithmetic circuit complexity. Github survey, 2015. URL: https://github.com/dasarpmar/lowerbounds-survey/releases/.

25 Eli Shamir and Marc Snir. On the depth complexity of formulas. *Math. Syst. Theory*, 13:301–322, 1980. doi:10.1007/BF01744302.

26 Amir Shpilka and Avi Wigderson. Depth-3 arithmetic circuits over fields of characteristic zero. *Computational Complexity*, 10(1):1–27, 2001.

27 Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends in Theoretical Computer Science*, 5:207–388, March 2010. doi:10.1561/0400000039.

28 Philip M. Spira. On time hardware complexity tradeoffs for Boolean functions. In Shu Lin, editor, *Proceedings of the Fourth Hawaii International Conference on System Sciences*, pages 525–527. Western Periodicals Company, North Hollywood, California, 1971.

29 Sébastien Tavenas. Improved bounds for reduction to depth 4 and depth 3. *Information and Computation*, 240:2–11, 2015. doi:10.1016/j.ic.2014.09.004.

30 Sébastien Tavenas, Nutan Limaye, and Srikanth Srinivasan. Set-multilinear and non-commutative formula lower bounds for iterated matrix multiplication. In *STOC*, pages 416–425. ACM, 2022.

31 Leslie G. Valiant, Sven Skyum, Stuart J. Berkowitz, and Charles Rackoff. Fast Parallel Computation of Polynomials Using Few Processors. *SIAM Journal of Computing*, 12(4):641–644, 1983. doi:10.1137/0212043.

32 SV Yablonskii and VP Kozyrev. Mathematical problems of cybernetics. *Information Materials of Scientific Council of Akad. Nauk SSSR on Complex Problem "Kibernetika"*, 19:3–15, 1968.

A The Bonet-Buss depth-reduction

In this section, we give a proof of Theorem 10. This is essentially the same proof as in the work of Bonet and Buss [2], but since the results in that paper are only stated for Boolean formulas, we give the proof here for completeness.

We show that for some large enough absolute constant $C > 0$, any polynomial P computed by a formula F of size s can also be computed by a formula F' of size at most $s^{1+\varepsilon}$, depth at most $2^{O(1/\varepsilon)} \cdot \log s$, and fan-in 2. Further, if F is homogeneous/monotone/non-commutative, then so is F'. This latter claim will follow directly from the construction.

We assume here that each gate in F has fan-in 2 to begin with. We can make this modification to F in the beginning at the expense of increasing the depth, and without increasing the size.

We prove the claim by induction on the size s of the formula. Let $T(s)$ and $D(s)$ denote the maximum size and depth (respectively) of F' thus obtained, assuming that F has size at most s. Let $k = 2^{C/\varepsilon}$ where $C > 0$ is an absolute constant we will choose below. We assume throughout that $\varepsilon < 1$, which is without loss of generality.

The base case of the formula is when $s \leqslant k$, in which case the claim is trivial, as any formula of size s has depth at most s, which in this case is at most $2^{C/\varepsilon}$.

Now, assume that $s > k$. In this case, we find a gate α such that the size of the subformula F_α rooted at α has size at least $s - s/k$, while the children β and γ of α do not satisfy this property. It is easy to observe that there is a unique α with this property. Let $*$ denote the operation (either $+$ or $\times$) labelling α.

We replace α by a fresh variable y in F to get a formula F_y. Note that F_y has size at most $s_1 = \frac{2s}{k}$. Since F_y has at most one occurrence of y, we can write

$$F_y = A \cdot y \cdot B + C$$

where A is the product of all subformulas that multiply y on the left in F along the path to the output, B is similarly the product of all subformulas that multiply y on the right, and C is the polynomial computed by F_y when we set y to 0. Clearly, A, B and C have formulas F_A, F_B and F_C of size at most s_1. Finally, we get the formula F' as in Figure 1.

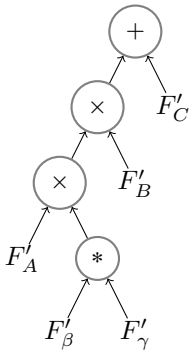


Figure 1 Constructing F'.

Here, $F'_A, F'_B, F'_C, F'_\beta, F'_\gamma$ are the formulas obtained by recursively applying the same procedure to $F_A, F_B, F_C, F_\beta, F_\gamma$.

We can bound the size and depth of the depth-reduced formula by induction. We have

$$D(s) \leqslant D(s - s/k) + O(1)$$

leading easily to an overall depth bound of $O(k \log s) = 2^{O(1/\varepsilon)} \cdot \log s$ for any choice of the constant C.

For the size, we have the following recursion.

$$T(s) \leqslant \max_{\substack{s_\beta, s_\gamma, s_1: \\ s_\beta, s_\gamma \leqslant s(1-1/k) \\ s_1 \leqslant 2s/k \\ s_\beta + s_\gamma + s_1 \leqslant s}} T(s_\beta) + T(s_\gamma) + 3T(s_1)$$

where s_β and s_γ are the sizes of F_β and F_γ respectively. We now use induction to bound $T(s)$ as follows. (We omit the conditions on s_β, s_γ and s_1 for notational simplicity.)

$$\begin{aligned} T(s) &\leqslant \max_{s_\beta, s_\gamma, s_1} s_\beta^{1+\varepsilon} + s_\gamma^{1+\varepsilon} + 3s_1^{1+\varepsilon} \\ &\leqslant \left(s\left(1 - \frac{1}{k}\right)\right)^{1+\varepsilon} + \left(\frac{s}{k}\right)^{1+\varepsilon} + 3\left(\frac{2s}{k}\right)^{1+\varepsilon} \end{aligned}$$

where we used the fact that $s_1 \leqslant 2s/k$ and the fact that, using the convexity of the map $x \mapsto x^{1+\varepsilon}$, the function $s_\beta^{1+\varepsilon} + s_\gamma^{1+\varepsilon}$ is maximized when $\max\{s_\beta, s_\gamma\} = s - s/k$, meaning that $\min\{s_\beta, s_\gamma\} \leqslant s/k$.

Continuing the computation, we get

$$T(s) \leqslant s^{1+\varepsilon}\left(\left(1 - \frac{1}{k}\right) + \frac{1}{k^{1+\varepsilon}} + 3\frac{4}{k^{1+\varepsilon}}\right) \leqslant s^{1+\varepsilon} \cdot \left(1 - \frac{1}{k} + \frac{C'}{k^{1+\varepsilon}}\right)$$

for some large enough absolute constant C'. Setting $k = 2^{C/\varepsilon}$ for a large enough absolute constant C gives us $T(s) \leqslant s^{1+\varepsilon}$, proving the inductive claim.

Constant-Depth Circuits vs. Monotone Circuits

Bruno P. Cavalar
University of Warwick, Coventry, UK

Igor C. Oliveira
University of Warwick, Coventry, UK

Abstract

We establish new separations between the power of monotone and general (non-monotone) Boolean circuits:

- For every $k \geq 1$, there is a monotone function in AC^0 (constant-depth poly-size circuits) that requires monotone circuits of depth $\Omega(\log^k n)$. This significantly extends a classical result of Okol'nishnikova [49] and Ajtai and Gurevich [1]. In addition, our separation holds for a monotone graph property, which was unknown even in the context of AC^0 versus mAC^0.
- For every $k \geq 1$, there is a monotone function in $\mathsf{AC}^0[\oplus]$ (constant-depth poly-size circuits extended with parity gates) that requires monotone circuits of size $\exp(\Omega(\log^k n))$. This makes progress towards a question posed by Grigni and Sipser [32].

These results show that constant-depth circuits can be more efficient than monotone formulas and monotone circuits when computing monotone functions.

In the opposite direction, we observe that non-trivial simulations are possible in the absence of parity gates: every monotone function computed by an AC^0 circuit of size s and depth d can be computed by a monotone circuit of size $2^{n-n/O(\log s)^{d-1}}$. We show that the existence of significantly faster monotone simulations would lead to breakthrough circuit lower bounds. In particular, if every monotone function in AC^0 admits a polynomial size monotone circuit, then NC^2 is not contained in NC^1.

Finally, we revisit our separation result against monotone circuit size and investigate the limits of our approach, which is based on a monotone lower bound for constraint satisfaction problems (CSPs) established by Göös, Kamath, Robere and Sokolov [31] via lifting techniques. Adapting results of Schaefer [67] and Allender, Bauland, Immerman, Schnoor and Vollmer [4], we obtain an unconditional classification of the monotone circuit complexity of Boolean-valued CSPs via their polymorphisms. This result and the consequences we derive from it might be of independent interest.

2012 ACM Subject Classification Theory of computation → Circuit complexity

Keywords and phrases circuit complexity, monotone circuit complexity, bounded-depth circuis, constraint-satisfaction problems

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.29

Related Version *Full Version*: `https://eccc.weizmann.ac.il/report/2023/069/`

Funding This work received support from the Royal Society University Research Fellowship URF\R1\191059, the EPSRC New Horizons Grant EP/V048201/1, and the Centre for Discrete Mathematics and its Applications (DIMAP) at the University of Warwick.

Acknowledgements We thank Arkadev Chattopadhyay for several conversations about the AC^0 versus $\mathsf{mSIZE}[\mathsf{poly}]$ problem and related questions. We are also grateful to Denis Kuperberg for explaining to us the results from [46, 47]. The first author thanks Ninad Rajgopal for helpful discussions about depth reduction. Finally, we thank Gernot Salzer for the code used to generate Figures 1, 2, and 3.

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 29; pp. 29:1–29:37

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

1 Introduction

A Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ is monotone if $f(x) \leq f(y)$ whenever $x_i \leq y_i$ for each coordinate $1 \leq i \leq n$. Monotone Boolean functions, and the monotone Boolean circuits[1] that compute them, have been extensively investigated for decades due to their relevance in circuit complexity [58], cryptography [10], learning theory [14], proof complexity [45, 54], property testing [28], pseudorandomness [22], optimisation [30], hazard-free computations [37], and meta-complexity [36], among other topics. In addition, over the last few years a number of results have further highlighted the importance of monotone complexity as a central topic in the study of propositional proofs, total search problems, communication protocols, and related areas (see [26] for a recent survey).

Some of the most fundamental results about monotone functions deal with their complexities with respect to different classes of Boolean circuits, such as the monotone circuit lower bound of Razborov [59] for Matching and the constant-depth circuit lower bound of Rossman [64] for k-Clique. Particularly important to our discussion is a related strand of research that contrasts the computational power of monotone circuits relative to general (non-monotone) AND/OR/NOT circuits, which we review next.

Weakness of Monotone Circuits. The study of monotone simulations of non-monotone computations and associated separation results has a long and rich history. In a sequence of celebrated results, [59, 8, 7, 69] showed the existence of monotone functions that can be computed by circuits of polynomial size but require monotone circuits of size $2^{n^{\Omega(1)}}$. In other words, the use of negations can significantly speedup the computation of monotone functions. More recently, Göös, Kamath, Robere and Sokolov [31] considerably strengthened this separation by showing that some monotone functions in NC^2 (poly-size $O(\log^2 n)$-depth fan-in two circuits) require monotone circuits of size $2^{n^{\Omega(1)}}$. (An earlier weaker separation against monotone depth $n^{\Omega(1)}$ was established in [57].) Therefore, negations can also allow monotone functions to be efficiently computed in parallel.

Similar separations about the limitations of monotone circuits are also known at the low-complexity end of the spectrum: Okol'nishnikova [49] and (independently) Ajtai and Gurevich [1] exhibited monotone functions in AC^0 (i.e., constant-depth poly-size AND/OR/NOT circuits) that require monotone AC^0 circuits (composed of only AND/OR gates) of super-polynomial size.[2] This result has been extended to an exponential separation in [24], which shows the existence of a monotone function in AC^0 that requires monotone depth-d circuits of size $2^{\widetilde{\Omega}(n^{1/d})}$ even if MAJ (majority) gates are allowed in addition to AND/OR gates.[3]

Strength of Monotone Circuits. In contrast to these results, in many settings negations do not offer a significant speedup and monotone computations can be unexpectedly powerful. For instance, monotone circuits are able to efficiently implement several non-trivial algorithms, such as solving constraint satisfaction problems using treewidth bounds (see, e.g., [50,

[1] Recall that in a monotone Boolean circuit the gate set is limited to $\{\mathsf{AND}, \mathsf{OR}\}$ and input gates are labelled by elements from $\{x_1, \ldots, x_n, 0, 1\}$.

[2] We refer to [13] for an alternate exposition of this result.

[3] Separations between monotone and non-monotone devices have also been extensively investigated in other settings. This includes average-case complexity [12], different computational models, such as span programs [9, 62] and algebraic complexity (see [21] and references therein), and separations in first-order logic [68, 46, 47]. We restrict our attention to worst-case separations for Boolean circuits in this paper.

Chapter 3]). As another example, in the context of cryptography, it has been proved that if one-way functions exist, then there are monotone one-way functions [29]. Below we describe results that are more closely related to the separations investigated in our paper.

In the extremely constrained setting of depth-2 circuits, Quine [55] showed that monotone functions computed by size-s DNFs (resp., CNFs) can always be computed by size-s monotone DNFs (resp., CNFs). Some results along this line are known for larger circuit depth, but with respect to more structured classes of monotone Boolean functions. Rossman [63, 66] showed that any homomorphism-preserving graph property computed by AC^0 circuits is also computed by monotone AC^0 circuits.[4] Under no circuit depth restriction, Berkowitz [11] proved that the monotone and non-monotone circuit size complexities of every slice function are polynomially related.[5]

Despite much progress and sustained efforts, these two classes of results leave open tantalising problems about the power of cancellations in computation.[6] In particular, they suggest the following basic question about the contrast between the weakness of monotone computations and the strength of negations:

What is the largest computational gap between the power of monotone and general (non-monotone) Boolean circuits?

A concrete formalisation of this question dates back to the seminal work on monotone complexity of Grigni and Sipser [32] in the early nineties. They asked if there are monotone functions in AC^0 that require super-polynomial size monotone Boolean circuits, i.e., if $\mathsf{AC}^0 \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[\mathsf{poly}]$. In case this separation holds, it would exhibit the largest qualitative gap between monotone and general Boolean circuits, i.e., even extremely parallel non-monotone computations can be more efficient than arbitrary monotone computations.

1.1 Results

Our results show that, with respect to the computation of monotone functions, highly parallel (non-monotone) Boolean circuits can be super-polynomially more efficient than unrestricted monotone circuits. Before providing a precise formulation of these results, we introduce some notation.

For a function $d\colon \mathbb{N} \to \mathbb{N}$, let $\mathsf{mDEPTH}[d]$ denote the class of Boolean functions computed by monotone fan-in two AND/OR Boolean circuits of depth $O(d(n))$. Similarly, we use $\mathsf{mSIZE}[s]$ to denote the class of Boolean functions computed by monotone circuits of size $O(s(n))$. More generally, for a circuit class $\mathcal{C}$, we let $\mathsf{m}\mathcal{C}$ denote its natural monotone analogue. Finally, for a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, we use $\mathsf{mSIZE}(f)$ and $\mathsf{mDEPTH}(f)$ to denote its monotone circuit size and depth complexities, respectively. We refer to Jukna [42] for standard background on circuit complexity theory.

---

[4] A function $f : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ is called a *graph property* if $f(G) = f(H)$ whenever G and H are isomorphic graphs, and *homomorphism-preserving* if $f(G) \leq f(H)$ whenever there is a graph homomorphism from G to H. It is easy to see that every homomorphism-preserving graph property is monotone.

[5] A function $f : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ is a *slice function* if there is $i \geq 0$ such that $f(x)$ is 0 on inputs of Hamming weight less than i and 1 on inputs of Hamming weight larger than i.

[6] Any non-monotone circuit can be written as an XOR (parity) of distinct monotone sub-circuits (see, e.g., [33, Appendix A.1]), so negations can be seen as a way of combining, or cancelling, different monotone computations. See also a related discussion in Valiant [71].

1.1.1 Constant-depth circuits vs. monotone circuits

Recall that the Okol'nishnikova-Ajtai-Gurevich [49, 1] theorem states that $\mathsf{AC}^0 \cap \mathsf{Mono} \not\subseteq \mathsf{mAC}^0$. In contrast, as our main result, we establish a separation between constant-depth Boolean circuits and monotone circuits of much larger depth. In particular, we show that constant-depth circuits with negations can be significantly more efficient than monotone formulas.

▶ **Theorem 1** (Polynomial-size constant-depth vs. larger monotone depth)**.** *For every $k \geq 1$, we have $\mathsf{AC}^0 \cap \mathsf{Mono} \not\subseteq \mathsf{mDEPTH}[(\log n)^k]$. Moreover, this separation holds for a monotone graph property.*

In a more constrained setting, Kuperberg [46, 47] exhibited a monotone graph property expressible in first-order logic that cannot be expressed in positive first-order logic. A separation that holds for a monotone graph property was unknown even in the context of AC^0 versus mAC^0.

Let $\mathsf{HomPreserving}$ denote the class of all homomorphism-preserving graph properties, and recall that Rossman [63, 66] established that $\mathsf{AC}^0 \cap \mathsf{HomPreserving} \subseteq \mathsf{mAC}^0$. Theorem 1 implies that this efficient monotone simulation does not extend to the larger class of monotone graph properties, even if super-logarithmic depth is allowed.

Our argument is completely different from those of [49, 1, 13, 24] and their counterparts in first-order logic [68, 46, 47]. In particular, it allows us to break the $O(\log n)$ monotone depth barrier present in previous separations with an AC^0 upper bound, which rely on lower bounds against monotone circuits of depth d and size (at most) $2^{n^{O(1/d)}}$. We defer the discussion of our techniques to Section 1.2.

In our next result, we consider monotone circuits of unbounded depth.

▶ **Theorem 2** (Polynomial-size constant-depth vs. larger monotone size)**.** *For every $k \geq 1$, we have $\mathsf{AC}^0[\oplus] \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{(\log n)^k}]$.*

Theorem 1 and Theorem 2 are incomparable: while the monotone lower bound is stronger in the latter, its constant-depth upper bound requires parity gates. Theorem 2 provides the first separation between constant-depth circuits and monotone circuits of polynomial size, coming remarkably close to a solution to the question considered by Grigni and Sipser [32].

We note that in both of our results the family of monotone functions is explicit and has a simple description (see Section 1.2).

1.1.2 Non-trivial monotone simulations and their consequences

While Theorem 1 and Theorem 2 provide more evidence for the existence of monotone functions in AC^0 which require monotone circuits of super-polynomial size, they still leave open the intriguing possibility that unbounded fan-in $\oplus$-gates might be crucial to achieve the utmost cancellations (speedups) provided by constant-depth circuits. This further motivates the investigation of efficient monotone simulations of constant-depth circuits without parity gates, which we consider next.

For convenience, let $\mathsf{AC}^0_d[s]$ denote the class of Boolean functions computed by AC^0 circuits of depth $\leq d$ and size $\leq s(n)$. (We might omit $s(n)$ and/or d when implicitly quantifying over all families of polynomial size circuits and/or all constant depths.)

We observe that a non-trivial monotone simulation is possible in the absence of parity gates. Indeed, by combining existing results from circuit complexity theory, it is not hard to show that $\mathsf{AC}^0_d[s] \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[2^{n(1-1/O(\log s)^{d-1})}]$ (see Section 4.1). Moreover, this upper

bound is achieved by monotone DNFs of the same size. This is the best upper bound we can currently show for the class of all monotone functions when the depth $d \geq 3$. (Negations offer no speedup at depths $d \leq 2$ [55].) In contrast, we prove that a significantly faster monotone simulation would lead to new (non-monotone) lower bounds in complexity theory. Recall that it is a notorious open problem to obtain explicit lower bounds against depth-d circuits of size $2^{\omega(n^{1/(d-1)})}$, for any fixed $d \geq 3$. We denote by GraphProperties the set of all Boolean functions which are graph properties.

▶ **Theorem 3** (New circuit lower bounds from monotone simulations). *There exists $\varepsilon > 0$ such that the following holds.*

1. *If* $\mathsf{AC}^0_3 \cap \mathsf{Mono} \subseteq \mathsf{mNC}^1$, *then* $\mathsf{NP} \not\subseteq \mathsf{AC}^0_3[2^{o(n)}]$.
2. *If* $\mathsf{AC}^0_4 \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, *then* $\mathsf{NP} \not\subseteq \mathsf{AC}^0_4[2^{o(\sqrt{n}/\log n)}]$.
3. *If* $\mathsf{AC}^0 \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, *then* $\mathsf{NC}^2 \not\subseteq \mathsf{NC}^1$.
4. *If* $\mathsf{NC}^1 \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[2^{O(n^{\varepsilon})}]$, *then* $\mathsf{NC}^2 \not\subseteq \mathsf{NC}^1$.
5. *If* $\mathsf{AC}^0 \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, *then* $\mathsf{NP} \not\subseteq \mathsf{NC}^1$.
6. *If* $\mathsf{NC}^1 \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, *then* $\mathsf{L} \not\subseteq \mathsf{NC}^1$.

Item (3) of Theorem 3 implies in particular that, if the upper bound of Theorem 2 cannot be improved to AC^0 (i.e., the question asked by [32] has a negative answer), then $\mathsf{NC}^2 \not\subseteq \mathsf{NC}^1$. It also improves a result from [23] showing the weaker conclusion $\mathsf{NP} \not\subseteq \mathsf{NC}^1$ under the same assumption.

Even if it's impossible to efficiently simulate AC^0 circuits computing monotone functions using unbounded depth monotone circuits, it could still be the case that a simulation exists for certain classes of monotone functions with additional structure. As explained above, Rossman's result [63, 66] achieves this for graph properties that are preserved under homomorphisms. Items (5) and (6) of Theorem 3 show that a simulation that holds for all monotone graph properties is sufficient to get new separations in computational complexity.

1.1.3 Monotone complexity of constraint satisfaction problems

Recall that [31] showed the existence of a monotone function f^{GKRS} in NC^2 that is not in $\mathsf{mSIZE}[2^{n^{\Omega(1)}}]$. As opposed to classical results [59, 8, 7, 69] that rely on the approximation method, their monotone circuit lower bound employs a lifting technique from communication complexity. It is thus natural to consider if their approach can be adapted to provide a monotone function g that is efficiently computable by constant-depth circuits but is not in $\mathsf{mSIZE}[\mathsf{poly}]$.

As remarked in [31, 26], all monotone lower bounds obtained from lifting theorems so far also hold for monotone encodings of constraint satisfaction problems (CSPs). Next, we introduce a class of monotone Boolean functions $\mathsf{CSP\text{-}SAT}_S$ which capture the framework and lower bound of [31].

Encoding CSPs as monotone Boolean functions. Let $R \subseteq \{0,1\}^k$ be a relation. We call k the *arity* of R. Let $V = (i_1, \ldots, i_k) \in [n]^k$, and let $f_{R,V} : \{0,1\}^n \to \{0,1\}$ be the function that accepts a string $x \in \{0,1\}^n$ if $(x_{i_1}, \ldots, x_{i_k}) \in R$. We call $f_{R,V}$ a *constraint application* of R on n variables. (A different choice of the sequence V gives a different constraint application of R.) If S is a finite set of Boolean relations, we call any set of constraint applications of relations from S on a fixed set of variables an *S-formula.* In particular, we can describe an S-formula through a set of pairs (V, R). We say that an S-formula F is *satisfiable* if there exists an assignment to the variables of F which satisfies all the constraints of F.

Let $S = \{R_1, \ldots, R_k\}$ be a finite set of Boolean relations. Let ℓ_i be the arity of the relation R_i. Note that there are n^{ℓ_i} possible constraint applications of the relation R_i on n variables. Let $N := \sum_{i=1}^{k} n^{\ell_i}$. We can identify each S-formula F on a fixed set of n variables with a corresponding string $w^F \in \{0,1\}^N$, where $w^F_j = 1$ if and only if the j-th possible constraint application (corresponding to one of the N pairs (V, R)) appears in F. Let $\mathsf{CSP\text{-}SAT}^n_S : \{0,1\}^N \to \{0,1\}$ be the Boolean function which accepts a given S-formula F if F is *unsatisfiable*. Note that this is a monotone function. When n is clear from the context or we view $\{\mathsf{CSP\text{-}SAT}^n_S\}_{n\geq 1}$ as a sequence of functions, we simply write $\mathsf{CSP\text{-}SAT}_S$.

The function f^{GKRS} from [31] is simply $\mathsf{CSP\text{-}SAT}_S$ for $S = \{\oplus^0_3, \oplus^1_3\}$, where we write $\oplus^b_3(x_1, x_2, x_3) = 1$ if and only if $\sum_i x_i = b \pmod 2$. More generally, for any finite set S of Boolean relations, their framework shows how to lift a Resolution width (resp. depth) lower bound for an arbitrary unsatisfiable S-formula F over m variables into a corresponding monotone circuit size (resp. depth) lower bound for $\mathsf{CSP\text{-}SAT}^n_S$, where $n = \mathsf{poly}(m)$.

Despite the generality of the technique from [31] and the vast number of possibilities for S, we prove that a direct application of their approach cannot establish Theorem 1 and Theorem 2. This is formalised as follows. (We refer to Section 5 for much stronger forms of the result.)

▶ **Theorem 4** (Limits of the direct approach via lifting and CSPs). *Let S be a finite set of Boolean relations. The following holds.*

1. *If* $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mSIZE}[\mathsf{poly}]$ *then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\oplus\mathsf{L}$*-hard under* $\leq^{\mathsf{AC}^0}_m$ *reductions.*
2. *If* $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mNC}^1$ *then* $\mathsf{CSP\text{-}SAT}_S$ *is* L*-hard under* $\leq^{\mathsf{AC}^0}_m$ *reductions.*

In particular, since there are functions (e.g., Majority) computable in logarithmic space that are not in $\mathsf{AC}^0[\oplus]$, Theorem 4 (Part 2) implies that any $\mathsf{CSP\text{-}SAT}_S$ function that is hard for poly-size monotone formulas (mNC^1) must lie outside $\mathsf{AC}^0[\oplus]$. Observe that this can also be interpreted as a *monotone simulation*: for any finite set S of Boolean relations, if $\mathsf{CSP\text{-}SAT}_S \in \mathsf{AC}^0[\oplus]$ then $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNC}^1$.[7]

Theorem 4 is a corollary of a general result that completely classifies the monotone circuit complexity of Boolean-valued constraint satisfaction problems based on the set $\mathsf{Pol}(S)$ of *polymorphisms* of S, a standard concept in the investigation of CSPs.[8] We present next a simplified version of this result, which shows a dichotomy for the monotone circuit size and depth of Boolean-valued constraint satisfaction problems. We refer to Section 5 for a more general formulation and additional consequences.

▶ **Theorem 5** (Dichotomies for the monotone complexity of Boolean-valued CSPs). *Let S be a finite set of Boolean relations. The following holds.*

1. Monotone Size Dichotomy: *If* $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$ *there is* $\varepsilon > 0$ *such that* $\mathsf{mSIZE}(\mathsf{CSP\text{-}SAT}_S) = 2^{\Omega(n^\varepsilon)}$. *Otherwise,* $\mathsf{mSIZE}(\mathsf{CSP\text{-}SAT}_S) = n^{O(1)}$.
2. Monotone Depth Dichotomy: *If* $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$ *or* $\mathsf{Pol}(S) \subseteq \mathrm{V}_2$ *or* $\mathsf{Pol}(S) \subseteq \mathrm{E}_2$*, there is* $\varepsilon > 0$ *such that* $\mathsf{mDEPTH}(\mathsf{CSP\text{-}SAT}_S) = \Omega(n^\varepsilon)$. *Otherwise,* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNC}^2$.

---

[7] Jumping ahead, our proof of Theorem 2 still relies in a crucial way on the monotone lower bound obtained by [31]. However, our argument requires an extra ingredient and does not follow from a direct application of their template. We provide more details about it in Section 1.2 below. Interestingly, the proof of Theorem 1 was discovered by trying to avoid the "barrier" posed by Theorem 4.

[8] Roughly speaking, $\mathsf{Pol}(S)$ captures the amount of symmetry in S, and a larger set $\mathsf{Pol}(S)$ implies that solving $\mathsf{CSP\text{-}SAT}_S$ is computationally easier. We refer the reader to Section 5 for more details and for a discussion of Post's lattice, which is relevant in the next statement.

We note that previous papers of Schaefer [67] and Allender, Bauland, Immerman, Schnoor and Vollmer [4] provided a *conditional* classification of the complexity of such CSPs. Theorem 5 and its extensions, which build on their results and techniques, paint a complete and *unconditional* picture of their monotone complexity.[9]

1.2 Techniques

Our arguments combine in novel ways several previously unrelated ideas from the literature. The exposition below follows the order in which the results appear above, except for the overview of the proof of Theorem 1, which appears last. We discuss this result after explaining the proof of Theorem 2 and the classification of the monotone complexity of CSPs (Theorem 4 and Theorem 5), as this sheds light into how the proof of Theorem 1 was discovered and into the nature of the argument.

A monotone circuit size lower bound for a function in $\mathbf{AC^0}[\oplus]$. We first give an overview of the proof of Theorem 2.

The lower bound of [31]. We begin by providing more details about the aforementioned monotone circuit lower bound of [31], since their result is a key ingredient in our separation (see [26] for a more detailed overview). Recall that their function f^{GKRS} corresponds to $\mathsf{CSP\text{-}SAT}_S$ for $S = \{\oplus_3^0, \oplus_3^1\}$. Following their notation, this is simply the Boolean function $\mathsf{3\text{-}XOR\text{-}SAT}_n \colon \{0,1\}^{2n^3} \to \{0,1\}$ which uses each input bit to indicate the presence of a linear equation with exactly 3 variables. This (monotone) function accepts a given linear system over $\mathbb{F}_2$ if the system is *unsatisfiable*. As one of their main results, [31] employed a lifting technique from communication complexity to show the existence of a constant $\varepsilon > 0$ such that $\mathsf{mSIZE}(\mathsf{3\text{-}XOR\text{-}SAT}_n) = 2^{n^\varepsilon}$. (We show in Appendix A that a weaker super-polynomial monotone circuit size lower bound for $\mathsf{3\text{-}XOR\text{-}SAT}_n$ can also be obtained using the approximation method and a reduction.)

Sketch of the proof of Theorem 2. Since $\mathsf{3\text{-}XOR\text{-}SAT}_n \in \mathsf{NC}^2$ (see, e.g, [31]), their result implies that $\mathsf{NC}^2 \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{n^{\Omega(1)}}]$. On the other hand, we are after a separation between *constant-depth* (non-monotone) circuits and *polynomial-size* (unbounded depth) monotone circuits. There are two natural ways that one might try to approach this challenge, as discussed next.

First, the lifting framework explored by [31] offers in principle the possibility that by carefully picking a different set S of Boolean relations, one might be able to reduce the non-monotone depth complexity of $\mathsf{CSP\text{-}SAT}_S$ while retaining super-polynomial monotone hardness. However, Theorem 4 shows that this is impossible, as explained above.

A second possibility is to combine the *exponential* 2^{n^ε} monotone circuit size lower bound for $\mathsf{3\text{-}XOR\text{-}SAT}_n$ and a padding argument, since we only need *super-polynomial* hardness. Indeed, this argument can be used to define a monotone function $g \colon \{0,1\}^n \to \{0,1\}$ that is computed by polynomial-size fan-in two circuits of depth $\mathsf{poly}(\log\log n)$ but requires monotone circuit of size $n^{\omega(1)}$. However, it is clear that no padding argument alone can reduce the non-monotone circuit depth bound to $O(1)$ while retaining the desired monotone hardness.

[9] We remark that only recently has Schaefer's classification been extended to the non-Boolean case [72, 15]. Though the refined classification of [4] is conjectured to hold analogously in the case of non-Boolean CSPs [48], this is still open (see the discussion in [16, Section 7]).

Given that both the classical widely investigated approximation method for monotone lower bounds and the more recent lifting technique do not appear to work in their current forms, for some time it seemed to us that, if true, a significantly new technique would be needed to establish a separation similar to the one in Theorem 2.

Perhaps surprisingly, it turns out that a more clever approach that combines padding with a non-trivial circuit upper bound can be used to obtain the result. The first key observation, already present in [31] and other papers, is that $\mathsf{3\text{-}XOR\text{-}SAT}_n$ can be computed not only in NC^2 but actually by polynomial-size span programs over $\mathbb{F}_2$. On the other hand, it is known that this model is equivalent in power to parity branching programs [44], which correspond to the non-uniform version of $\oplus\mathsf{L}$, i.e., counting modulo 2 the number of accepting paths of a nondeterministic Turing machine that uses $O(\log n)$ space. A second key idea is that such a computation can be simulated by $\mathsf{AC}^0[\oplus]$ circuits of sub-exponential size and large depth. More precisely, similarly to an existing simulation of NL (nondeterministic logspace) by AC^0 circuits of depth d and size $2^{n^{O(1/d)}}$ via a "guess-and-verify" approach, it is possible to achieve an analogous simulation of $\oplus\mathsf{L}$ using $\mathsf{AC}^0[\oplus]$ circuits (this folklore result appears implicit in [6] and [51]). Putting everything together, it follows that for a large enough but constant depth, $\mathsf{3\text{-}XOR\text{-}SAT}_n$ can be computed by $\mathsf{AC}^0[\oplus]$ circuits of size $2^{n^{\varepsilon/2}}$. Since this function is hard against monotone circuits of size $2^{n^{\varepsilon}}$, a padding argument can now be used to establish a separation between $\mathsf{AC}^0[\oplus]$ and $\mathsf{mSIZE}[\mathsf{poly}]$. (A careful choice of parameters provides the slightly stronger statement in Theorem 2.)

Non-trivial monotone simulations and their consequences. In order to conclude that significantly stronger monotone simulations imply new complexity separations (Theorem 3), we argue contrapositively. By supposing a complexity collapse, we can exploit known monotone circuit lower bounds to conclude that a hard monotone function exists in a lower complexity class. For instance, if $\mathsf{NC}^2 \subseteq \mathsf{NC}^1$, then $\mathsf{3\text{-}XOR\text{-}SAT} \in \mathsf{NC}^1$, and we can conclude by standard depth-reduction for NC^1 and padding, together with the exponential lower bound for $\mathsf{3\text{-}XOR\text{-}SAT}$ due to [31], that there exists a monotone function in AC^0 which is hard for polynomial-size monotone circuits. The other implications are argued in a similar fashion. In particular, we avoid the more complicated use of hardness magnification from [23] to establish this kind of result, while also getting a stronger consequence.

A little more work is required in the case of graph properties (Theorem 3 Items 5 and 6), as padding the function computing a graph property does not yield a graph property. We give a general lemma that allows us to pad monotone graph properties while preserving their structure (Lemma 12). We then argue as in the case for general functions, using known monotone lower bounds for graph properties. We note that Lemma 12 is also important in the proof of Theorem 1, which will be discussed below. We believe that our padding technique for graph properties might find additional applications.

Monotone complexity of CSPs. These are the most technical results of the paper. Since explaining the corresponding proofs requires more background and case analysis, here we only briefly describe the main ideas and references behind Theorem 4, Theorem 5, and the extensions discussed in Section 5.

A seminal work of Schaefer [67] proved that any Boolean CSP is either solvable in polynomial-time or it is NP-complete. Later, Jeavons [38] observed that the complexity of deciding if a given set of constraint applications of S is satisfiable depends exclusively on the set $\mathsf{Pol}(S)$ of *polymorphisms* of S. Intuitively, the set of polymorphisms of a set of relations is a measure of its symmetry. The more symmetric a set of relations is, the

lesser is its expressive power. Jeavons formally proves this intuition by showing that, if $\mathsf{Pol}(S) \subseteq \mathsf{Pol}(S')$, then the problem of deciding the satisfiability of a given S'-formula can be reduced in polynomial-time to that of deciding the satisfiability of a given S-formula. This allows Jeavons to reprove Schaefer's result.

Existing proofs and classification results for constraint satisfaction problems do not encode the satisfiability problem as a monotone Boolean function $\mathsf{CSP\text{-}SAT}_S$, in the way we described above. We reexamine Schaefer's and Jeavons's proofs and establish that the reduction from $\mathsf{CSP\text{-}SAT}_{S'}$ to $\mathsf{CSP\text{-}SAT}_S$ can also be done with efficient monotone circuits. Making use of and adapting parts of the refined results and analysis of [4], which builds on the earlier dichotomy result of [67] and provides a detailed picture of the computational complexity of Boolean-valued CSPs, we prove in fact that the underlying reductions can all be done in monotone nondeterministic logspace.

Finally, using known upper and lower bounds for monotone circuits together with a direct analysis of some basic cases, and inspecting Post's lattice [53, 18, 19], we are able to show that $\mathsf{CSP\text{-}SAT}_S$ is hard for monotone circuits only when $\mathsf{CSP\text{-}SAT}_S$ is $\oplus\mathsf{L}$-complete, as in Theorem 4 Part 1.

A monotone circuit depth lower bound for a function in AC^0. Next, we combine insights obtained from the monotone lower bound of [31], our proof of Theorem 2 via a guess-and-verify depth reduction and padding, and the statement of Theorem 4 (limits of the direct approach via CSPs) to get the separation in Theorem 1. As alluded to above, our approach differs from those of [49, 1, 13, 24] and related results in the context of first-order logic [68, 46, 47].

Recall that the [31] framework lifts a Resolution width lower bound for an unsatisfiable S-formula F into a corresponding monotone circuit size lower bound for $\mathsf{CSP\text{-}SAT}_S$. On the other hand, Theorem 4 rules out separating constant-depth circuits from monotone circuits of polynomial size via $\mathsf{CSP\text{-}SAT}_S$ functions. In particular, we cannot directly apply the chain of reductions from [31] to obtain the desired separation result. Instead, we extract from the specific S-formula F that they use a *structural property* that will allow us to improve the $\mathsf{AC}^0[\oplus]$ upper from Theorem 2 to the desired AC^0 upper bound in Theorem 1.

In [31] the formula F is a *Tseitin* contradiction, a well-known class of unsatisfiable CNFs with a number of applications in proof complexity. For an undirected graph G, the Tseitin formula $T(G)$ encodes a system of linear equations modulo 2 as follows: each edge $e \in E(G)$ becomes a Boolean variable x_e, and each vertex $v \in V(G)$ corresponds to a constraint (linear equation) C_v stating that $\sum_{u \in N_G(v)} x_{\{v,u\}} = 1 \pmod 2$, where $N_G(v)$ denotes the set of neighbours of v in G. Crucially, $T(G)$ does not encode an arbitrary system of linear equations, i.e., the following key structural property holds: every variable x_e appears in exactly 2 equations.

On a technical level, this property is not preserved when obtaining a (total) monotone function $\mathsf{CSP\text{-}SAT}_S$ by the gadget composition employed in the lifting framework and its reductions. However, we can still hope to explore this property in a somewhat different argument with the goal of obtaining CSP instances that lie in a complexity class weaker than $\oplus\mathsf{L}$, which is the main bottleneck in the proof of Theorem 2 yielding $\mathsf{AC}^0[\oplus]$ circuits instead of AC^0. At the same time, considering this structural property immediately takes us outside the domain of Theorem 4, which does not impose structural conditions over the CSP instances.

We can capture the computational problem corresponding to this type of system of linear equations using the following Boolean function. Let $\mathsf{OddFactor}_n : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ be the function that accepts a given graph G if the formula $T(G)$ described above is *satisfiable*.

(Equivalently, if G admits a spanning subgraph in which the degree of every vertex is odd.) Note that $\mathsf{OddFactor}_n$ is a monotone Boolean function: adding edges to G cannot make a satisfiable system unsatisfiable, since we can always set a new edge variable x_e to 0.

While 3-XOR-SAT (the corresponding $\mathsf{CSP\text{-}SAT}_S$ function obtained from an appropriate Tseitin formula via the framework of [31]) admits a $\oplus\mathsf{L}$ upper bound, we observe that $\mathsf{OddFactor}_n$ can be computed in L thanks to its more structured class of input instances. Indeed, one can prove that the formula $T(G)$ is satisfiable if and only if every connected component of G has an even number of vertices.[10] In turn, the latter condition can be checked in logarithmic space using Reingold's algorithm for undirected s-t-connectivity [60]. (We note that related ideas appear in an unpublished note of Johannsen [40].) This is the first application of Reingold's algorithm to this kind of separation.

At the same time, $\mathsf{OddFactor}_n$ retains at least part of the monotone hardness of 3-XOR-SAT. Using a different reduction from a communication complexity lower bound, [9] proved that the monotone circuit depth of $\mathsf{OddFactor}_n$ is $n^{\Omega(1)}$. Altogether, we obtain a monotone Boolean function (indeed a graph property) that lies in L but is not in $\mathsf{mDEPTH}[n^{o(1)}]$. Applying a guess-and-verify depth reduction for L and using (graph) padding (analogously to the proof sketch of Theorem 2), we get a monotone graph property in AC^0 that is not in $\mathsf{mDEPTH}[\log^k n]$. This completes the sketch of the proof of Theorem 1.

1.3 Directions and open problems

Constant-depth circuits and monotone circuits are possibly the two most widely investigated models in circuit complexity theory. Although our results provide new insights about the relation between them, there are exceptionally basic questions that remain open.

While [55] showed that negations can be efficiently eliminated from circuits of depth $d \leq 2$ that compute monotone functions, already at depth $d = 3$ the situation is much less clear. Theorem 19 (see Section 4.1) implies that every monotone function in depth-3 AC^0 admits a monotone circuit of size $2^{n-\Omega(n/\log^2 n)}$. It is unclear to us if this is optimal. While [24] rules out an efficient *constant-depth* monotone simulation, it is still possible (and consistent with Theorem 1) that $\mathsf{AC}^0_3 \cap \mathsf{Mono} \subseteq \mathsf{mNC}^1$. Is there a significantly better monotone circuit size upper bound for monotone functions computed by polynomial-size depth-3 circuits?

Our results come close to solving the question posed by Grigni and Sipser [32]. Using our approach, it would be sufficient to show that $\mathsf{OddFactor}_n$ requires monotone circuits of size $\exp(n^{\Omega(1)})$. This is closely related to the challenge of obtaining an exponential monotone circuit size lower bound for $\mathsf{Matching}_n$, a longstanding open problem in monotone complexity (see [42, Section 9.11]).[11] Indeed, it's possible to reduce OddFactor to Matching using monotone AC^0 circuits (see [3, Lemma 6.18]).

Incidentally, the algebraic complexity variant of the AC^0 vs. $\mathsf{mSIZE}[\mathsf{poly}]$ problem has been recently settled in a strong way through a new separation result obtained by Chattopadhyay, Datta, and Mukhopadhyay [21]. Could some of their techniques be useful to attack the more elusive Boolean case?

---

[10] A simple parity argument shows that odd-sized components cannot be satisfied. On the other hand, we can always satisfy an even-sized component by starting with an arbitrary assignment, which must satisfy an even number of constraints by a parity argument, and flipping the values of the edges in a path between unsatisfied nodes, until all nodes in the connected component are satisfied.

[11] Note that in OddFactor we are concerned with the existence of a spanning subgraph where the degree of every vertex is odd, while in Matching the degree should be exactly 1.

Finally, it would be interesting to develop a more general theory able to explain when cancellations can speedup the computation of monotone Boolean functions. Our investigation of monotone simulations and separations for different classes of monotone functions (graph properties and constraint satisfaction problems) can be seen as a further step in this direction.

2 Preliminaries

2.1 Notation

Boolean functions. We denote by Mono the set of all monotone Boolean functions. We define $\mathsf{poly} = \{n \mapsto n^C : C \in \mathbb{N}\}$. A Boolean function $f : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ is said to be a graph property if $f(G) = f(H)$ for any two isomorphic graphs G and H. Let $\mathcal{F} = \{f_n\}_{n\in\mathbb{N}}$ be a sequence of graph properties, where f_n is defined over undirected graphs on n vertices. We say that $\mathcal{F}$ is *preserved under homomorphisms* if, whenever there is a homomorphism from a graph G to a graph H, we have $\mathcal{F}(G) \leq \mathcal{F}(H)$. We denote by $\mathsf{HomPreserving}$ the set of all graph properties which are preserved under homomorphisms. Note that $\mathsf{HomPreserving} \subseteq \mathsf{Mono}$.

Boolean circuits. We denote by $\mathsf{AC}^0_d[s]$ the family of Boolean functions computed by size-s, depth-d Boolean circuits with unbounded fan-in $\{\wedge, \vee\}$-gates and input literals from $\{x_1, \overline{x_1}, \ldots, x_n, \overline{x_n}\}$. We write $\mathsf{AC}^0[s]$ as a shorthand for $\bigcup_{d=1}^{\infty} \mathsf{AC}^0_d[s]$, and AC^0 as a shorthand of $\mathsf{AC}^0[n^{O(1)}] = \mathsf{AC}^0[\mathsf{poly}]$. We will also refer to $\mathsf{AC}^0_d[\mathsf{poly}]$ by AC^0_d. We write $\mathsf{DNF}[s]$ to denote the family of Boolean functions computed by size-s DNFs, where size is measured by number of terms. We write $\mathsf{CNF}[s]$ analogously. We write $\mathsf{SIZE}[s]$ to denote the family of Boolean functions computed by size-s circuits. We write $\mathsf{DEPTH}[d]$ to denote the family of Boolean functions computed by fan-in 2 circuits of depth d. We denote by $\mathsf{AC}^0[\oplus]$ the family of Boolean functions computed by polynomial-size AC^0 circuits with unbounded fan-in $\oplus$-gates.

We denote by L the family of Boolean functions computed by logspace machines, and by NL the family of Boolean functions computed by polynomial-time nondeterministic logspace machines. Moreover, we denote by $\oplus\mathsf{L}$ the family of Boolean functions computed by polynomial-time nondeterministic logspace machines with a *parity* acceptance condition (i.e., an input is accepted if the number of accepting paths is odd).

Circuit complexity. Given a circuit class $\mathcal{C}$, we write $\mathsf{m}C$ to denote the monotone version of $\mathcal{C}$. Given a function f, we write $\mathsf{mSIZE}(f)$ to denote the size of the smallest monotone circuit computing f and $\mathsf{mDEPTH}(f)$ to denote the smallest depth of a fan-in 2 monotone circuit computing f. Given two Boolean functions f, g, we write $f \leq_m^{\mathsf{mProj}} g$ if there exists a many-one reduction from f to g in which each bit of the reduction is a monotone projection[12] of the input.

Miscellanea. Let $\alpha \in \{0,1\}^n$. We define $|\alpha|_1 := \sum_{i=1}^n \alpha_i$. We call $|\alpha|_1$ the *Hamming weight* of α. We let $\mathrm{supp}(\alpha) = \{i \in [n] : \alpha_i = 1\}$. We let $\mathrm{THR}_{k,n} : \{0,1\}^n \to \{0,1\}$ be the Boolean function such that $\mathrm{THR}_{k,n}(x) = 1 \iff |x|_1 \geq k$.

[12] A monotone projection is a projection without negations.

2.2 Background results

The next lemma, which is proved via a standard "guess-and-verify" approach, shows that nondeterministic logspace computations can be simulated by circuits of size 2^{n^ε} and of depth $d = O_\varepsilon(1)$.

▶ **Lemma 6** (Folklore; see, e.g., [5, Lemma 8.1]). *For all $\varepsilon > 0$, we have* $\mathsf{NL} \subseteq \mathsf{AC}^0[2^{n^\varepsilon}]$.

3 Constant-Depth Circuits vs. Monotone Circuits

In this section, we prove Theorems 1 and 2. For the upper bounds, we require the logspace graph connectivity algorithm due to [60] and the $\oplus\mathsf{L}$ algorithm for solving linear systems over $\mathbb{F}_2$ due to [17], as well as the depth-reduction techniques of [6, 5]. On the lower bounds side, our proofs rely on previous monotone circuit and depth lower bounds from [9, 31]. In order to obtain a monotone formula lower bound for a graph property, we prove a graph padding lemma in Section 3.2.

3.1 A monotone size lower bound for a function in $\mathsf{AC}^0[\oplus]$

In this section, we prove Theorem 2. We first recall the monotone circuit lower bound of [31] and a depth-reduction lemma implicit in [6] and [51], whose full proof we give below for completeness. We remark that similar arguments can be employed to prove Lemma 6, essentially by replacing the $\oplus$ gates by $\vee$ gates.

As explained in Section 1.2, in its strongest form the separation result from [31] can be stated as follows.

▶ **Theorem 7** ([31]). *There exists $\varepsilon > 0$ such that $\oplus\mathsf{L} \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{o(n^\varepsilon)}]$. Moreover, this separation is witnessed by* 3-XOR-SAT.

▶ **Lemma 8** (Folklore; see, e.g., [6, 51]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function computed by a $\oplus\mathsf{L}$ machine. For every $\delta > 0$, there exists an $\mathsf{AC}^0[\oplus]$ circuit of size 2^{n^δ} that computes f.*

Proof. Let M be a $\oplus\mathsf{L}$-machine computing f. Without loss of generality, we may assume that each configuration in the configuration graph G of M is *time-stamped* – in other words, each configuration carries the information of the number of computational steps it takes to arrive at it.[13] We may also assume that every accepting computation takes exactly the same amount of time, which means that every path from the starting configuration v_{start} to the accepting configuration v_{accept} has the same length in the configuration graph. These assumptions imply that the configuration graph is *layered* (because a configuration with time-stamp t can only point to configurations with time-stamp $t+1$) and acyclic. Note that, for a fixed machine, the configuration graph can be computed from the input string using a projection.

Let $m = n^{O(1)}$ be the time that an accepting computation takes. We now show how to count (modulo 2) the number of accepting paths from v_{start} to v_{accept} with a depth-d $\mathsf{AC}^0[\oplus]$ circuit. First, choose $m^{1/d} - 1$ configurations $v_1, \ldots, v_{m^{1/d}-1}$ (henceforth called

[13] Formally, we can define a $\oplus\mathsf{L}$-machine M' such that the configurations of M' are (C,t), where C is a configuration of M, and $t = 0, 1, \ldots, m = n^{O(1)}$ is a number denoting the time in which the configuration was achieved. A configuration (C,t) can only reach a configuration $(C', t+1)$ in the configuration graph of M'.

"checkpoints") from $V(G)$, such that the configuration v_i is at the level $i \cdot m^{1-1/d}$ in the configuration graph (i.e., it takes $i \cdot m^{1-1/d}$ time steps to arrive at v_i). For convenience, we let $v_0 = v_{\mathsf{start}}$ and $v_{m^{1/d}} = v_{\mathsf{accept}}$. We then count the number of paths from from v_{start} to v_{accept} that go through $v_1, \ldots, v_{m^{1/d}-1}$, and sum over all possible choices of the checkpoints. Since the graph is layered and each path from v_0 to $v_{m^{1/d}}$ has length exactly m, there is only one choice of checkpoints that witnesses a given path from v_0 to $v_{m^{1/d}}$, so no path is counted twice in this summation. Letting $\#\mathsf{paths}(s, t, \ell)$ denote the number of paths between configurations s and t with distance exactly ℓ, we obtain

$$\#\mathsf{paths}(v_0, v_{m^{1/d}}, m) = \sum_{v_1,\ldots,v_{m^{1/d}-1}} \prod_{i=0}^{m^{1/d}-1} \#\mathsf{paths}(v_i, v_{i+1}, m^{1-1/d}).$$

The above calculation can be done in modulo 2 with an unbounded fan-in XOR gate (replacing the summation) and an unbounded fan-in AND gate (replacing the product). Note that the formula above is recursive. Repeating the same computation for calculating (modulo 2) the expression $\#\mathsf{paths}(v_i, v_{i+1}, m^{1-1/d})$ for each i, we obtain a depth-$2d$ $\mathsf{AC}^0[\oplus]$ circuit for calculating the number of paths from v_{start} to v_{accept} (modulo 2). Clearly, the total size of the circuit is $2^{O(m^{1/d} \cdot \log m)}$, which is smaller than 2^{n^δ} for a large enough constant d. ◀

We now restate Theorem 2 and prove it by combining Theorem 7 and Lemma 8 with a padding trick.

▶ **Theorem 2** (Polynomial-size constant-depth vs. larger monotone size)**.** *For every $k \geq 1$, we have* $\mathsf{AC}^0[\oplus] \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{(\log n)^k}]$.

Proof. By Theorem 7, there exists $\varepsilon > 0$ and a monotone function $f \in \oplus\mathsf{L}$ such that any monotone circuit computing f has size $2^{\Omega(n^\varepsilon)}$.

Let $\delta = \varepsilon/k$ and let $m = 2^{n^\delta}$. Let $g : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ be the Boolean function defined as $g(x, y) = f(x)$. Note that g is a function on $N := m + n = 2^{\Theta(n^\delta)}$ bits. By Lemma 8, there exists an $\mathsf{AC}^0[\oplus]$ circuit computing f of size $2^{n^\delta} = N^{O(1)}$. The same circuit computes g. On the other hand, any monotone circuit computing g has size $2^{\Omega(n^\varepsilon)} = 2^{\Omega((\log N)^{\varepsilon/\delta})} = 2^{\Omega((\log N)^k)}$. ◀

3.2 A monotone depth lower bound for a graph property in AC^0

In this section, we prove Theorem 1. We prove moreover that the function that separates $\mathsf{AC}^0 \cap \mathsf{Mono}$ and mNC^i can be taken to be a graph property. We state our result in its full generality below.

▶ **Theorem 9.** *For every $i \geq 1$, we have* $\mathsf{AC}^0 \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \not\subseteq \mathsf{mDEPTH}[(\log n)^i]$. *In particular, we have* $\mathsf{AC}^0 \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \not\subseteq \mathsf{mNC}^i$.

First, we recall a result of [9], which proves monotone lower bounds for the following function. Let $\mathsf{OddFactor}_n : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ be the function that accepts a given graph if it contains an *odd factor* – in other words, a spanning subgraph in which the degree of every vertex is odd. Babai, Gál and Wigderson [9] proved the following result:

▶ **Theorem 10** ([9])**.** *Any monotone formula computing* $\mathsf{OddFactor}_n$ *has size* $2^{\Omega(n)}$*, and any monotone circuit computing* $\mathsf{OddFactor}_n$ *has size* $n^{\Omega(\log n)}$.

The proof in [9] is actually for the case of *bipartite graphs*, but it easily extends to general graphs, since the bipartite case reduces to the general case by a monotone projection. The formula lower bound stated above is slightly stronger because it makes use of asymptotically optimal lower bounds on the randomized communication complexity of DISJ_n [43], which were not available to [9]. We remark that, with a different language, a monotone circuit lower bound for OddFactor is also implicitly proved in Feder and Vardi [27, Theorem 30].

We now recall an upper bound for OddFactor, implicitly proved in an unpublished note due to Johannsen [40].

▶ **Theorem 11** ([40]). *We have* $\mathsf{OddFactor} \in \mathsf{L}$.

Proof. We first recall the following observation about the OddFactor function, which appears in different forms in the literature (see [70, Lemma 4.1] or [42, Lemma 18.16]; see also [40, Proposition 1] for a different proof.)

▷ Claim. A graph G has an odd factor if and only if every connected component of G has an even number of vertices.

Proof. If a graph G has an odd factor, we can conclude that every connected component of G has an even number of vertices from the well-known observation that in every graph there is an even number of vertices of odd degree.

Now suppose that every connected component of G has an even number of vertices. We will iteratively construct an odd factor F of G. We begin with the empty graph. We take any two vertices u, v in the same connected component of G which currently have even degree in F, and consider any path $P = (x_1, \ldots, x_k)$ between u and v, where $x_1 = u$ and $x_k = v$. If the edge $x_i x_{i+1}$ is currently in F, we remove $x_i x_{i+1}$ from F; otherwise, we add $x_i x_{i+1}$ to F. It's easy to check that, in every iteration of this procedure, only the vertices u and v have the parity of their degree changed in F; the degree of every other vertex stays the same (modulo 2). Since every connected component has an even number of vertices, this means that, eventually, every vertex in F will have odd degree. ◁

Now it's easy to check in logspace if every connected component of G has an even number of vertices using Reingold's algorithm for undirected connectivity [60]. It suffices to check if, for every vertex v of G, the number of vertices reachable from v is odd. ◀

Now, if we only desire to obtain a function in AC^0 not computed by monotone circuits of depth $(\log n)^i$, we can follow the same argument of Theorem 2, using Lemma 6 instead of Lemma 8. In order to obtain moreover a monotone *graph property* witnessing this separation, we will need the following lemma, which enables us to obtain a graph property after "padding" a graph property. We defer the proof of this lemma to the end of this section.

▶ **Lemma 12.** *Let* $f : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ *be a monotone graph property on graphs of* n *vertices. The following holds.*

1. *If* $f \in \mathsf{NC}^i$ *for some* $i > 1$*, then there exists a monotone graph property* g *on graphs of* $N = 2^{(\log n)^i}$ *vertices such that* $g \in \mathsf{NC}^1$ *and* $f \leq_m^{\mathsf{mProj}} g$.
2. *If* $f \in \mathsf{NL}$*, then for all* $\varepsilon > 0$ *there exists a monotone graph property* g *on graphs of* $N = 2^{n^\varepsilon}$ *vertices such that* g *can be computed by* AC^0 *circuits of size* $N^{2+o(1)}$ *and* $f \leq_m^{\mathsf{mProj}} g$.
3. *If* $f \in \oplus\mathsf{L}$*, then for all* $\varepsilon > 0$ *there exists a monotone graph property* g *on graphs of* $N = 2^{n^\varepsilon}$ *vertices such that* g *can be computed by* $\mathsf{AC}^0[\oplus]$ *circuits of size* $N^{2+o(1)}$ *and* $f \leq_m^{\mathsf{mProj}} g$.

We are now ready to prove Theorem 9.

Proof of Theorem 9. Fix $n \in \mathbb{N}$ and take an $\varepsilon < 1/i$. Observing that $\mathsf{L} \subseteq \mathsf{NL}$, from Theorem 11 and item (2) of Lemma 12 we conclude that there exists a monotone graph property f on $N = 2^{n^\varepsilon}$ vertices such that $f \in \mathsf{AC}^0$ and $\mathsf{OddFactor}_n \leq_m^{\mathsf{mProj}} f$. By Theorem 10, any monotone circuit computing f has depth $\Omega(n) = \Omega((\log N)^{1/\varepsilon}) \gg (\log N)^i$. ◀

Raz and Wigderson [57] observed that there exists a monotone function $f \in \mathsf{NC}^1 \setminus \mathsf{mNC}$. Using Lemma 12, we observe moreover that it's possible to obtain this separation with a monotone graph property.

▶ **Proposition 13.** *We have* $\mathsf{NC}^1 \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \not\subseteq \mathsf{mNC}$.

Proof. Observing that $\mathsf{L} \subseteq \mathsf{NC}^2$, we conclude from Theorem 11 and item (1) of Lemma 12 that there exists a monotone graph property f on $N = 2^{(\log n)^2}$ vertices such that $f \in \mathsf{NC}^1$ and $\mathsf{OddFactor}_n \leq_m^{\mathsf{mProj}} f$. By Theorem 10, any monotone circuit computing f has depth $\Omega(n) = \Omega(2^{\sqrt{\log N}})$, which implies $f \notin \mathsf{mNC}$. ◀

3.3 Efficient monotone padding for graph properties

We will now prove Lemma 12. We first recall some low-depth circuits for computing threshold functions, which we will use to design a circuit for efficiently computing the adjacency matrix of induced subgraphs.

▶ **Theorem 14** ([35]). *Let $d > 0$ be a constant. The function $\mathrm{THR}_{(\log n)^d, n}$ can be computed by an AC^0 circuit of size $n^{o(1)}$ and depth $d + O(1)$.*

▶ **Theorem 15** ([2]). *For every $k \in [n]$, the function $\mathrm{THR}_{k,n}$ can be computed by a circuit of depth $O(\log n)$ and size $n^{O(1)}$.*

▶ **Lemma 16.** *There exists a circuit C_n^k with $\binom{n}{2} + n$ inputs and $\binom{k}{2}$ outputs which, when given as input an adjacency matrix of a graph G on n vertices and a characteristic vector of a set $S \subseteq [n]$ such that $|S| \leq k$, outputs the adjacency matrix of the graph $G[S]$, padded with isolated vertices when $|S| < k$. The circuit has constant-depth and size $n^{2+o(1)}$ when $k = \mathrm{polylog}(n)$, and size $n^{O(1)}$ and depth $O(\log n)$ otherwise.*

Proof. Let $\{x_{ij}\}_{i,j \in [n]}$ encode the adjacency matrix of G. Let $\alpha \in \{0,1\}^n$ be the characteristic vector of S. Let $i, j \in [k]$. Note that $\{i, j\} \in E(G[S])$ if and only if there exists $a, b \in [n]$ such that

- α_a is the i-th non-zero entry of α,
- α_b is the j-th non-zero entry of α, and
- $x_{ab} = 1$ (i.e., a and b are connected in G).

We first consider the case $k = \mathrm{polylog}(n)$. In this case, the first two conditions can be checked with circuits of size $n^{o(1)}$ using Theorem 14. Therefore, we can compute if i and j are adjacent using $n^{2+o(1)}$ gates and constant depth. As there are at most $(\log n)^{O(1)}$ such pairs, we can output $G[S]$ with at most $n^{2+o(1)}$ gates.

For any k, the first two conditions can be checked with an NC^1 circuit by Theorem 15. Since there are at most n^2 pairs i, j, the entire adjacency matrix can be computed with a $O(\log n)$-depth and polynomial-size circuit. ◀

We are ready to prove Lemma 12.

Proof of Lemma 12. We first prove (1). Fix $n \in \mathbb{N}$ and let $N = 2^{(\log n)^i}$. For a graph G on N vertices such that $|E(G)| \leq \binom{n}{2}$, let G_{clean} be the graph obtained from G by removing isolated vertices from G one-by-one, in lexicographic order, until one of the following two conditions are satisfied: (1) there are no more isolated vertices in G_{clean}, *or* (2) G_{clean} has exactly n vertices. Let $g : \{0,1\}^{\binom{N}{2}} \to \{0,1\}$ be the monotone graph property defined as follows:

$$g(G) := \left(|E(G)| > \binom{n}{2}\right) \vee (|V(G_{\mathsf{clean}})| > n) \vee (f(G_{\mathsf{clean}}) = 1).$$

Note that g accepts a graph G if and only if at least one of the following three conditions are satisfied:

1. G has at most $\binom{n}{2}$ edges, G_{clean} has exactly n vertices and $f(G_{\mathsf{clean}}) = 1$, or
2. G has more than $\binom{n}{2}$ edges, or
3. G_{clean} has more than n vertices.

We observe that the monotonicity of g follows from the monotonicity of f. We also claim that g is a graph property. Indeed, the graph G_{clean} is the same (up to isomorphism), irrespective of the order according to which the isolated vertices are removed from G. Moreover, the function f is also a graph property. Because of this, all the three conditions above are preserved under isomorphisms.

We first observe that f is a monotone projection of g. Indeed, given a graph G on n vertices, we can easily construct by a monotone projection a graph G' on N vertices and at most $\binom{n}{2}$ edges such that $f(G) = g(G')$. We just let G' have a planted copy of G, and all other vertices are isolated. Then $G'_{\mathsf{clean}} = G$ (up to isomorphism) and $g(G') = f(G_{\mathsf{clean}}) = f(G)$.

We now show how to compute g in NC^1. Let $\{x_{ij}\}_{i,j\in[N]}$ be the input bits of g, corresponding to the adjacency matrix of a graph G. The circuit computes as follows.

1. If $|E(G)| > \binom{n}{2}$, accept the graph G.
2. Compute the characteristic vector $\alpha \in \{0,1\}^N$ of the set of all non-isolated vertices of G. If $|\alpha|_1 > n$, accept the graph G.
3. Compute G_{clean} and output $f(G_{\mathsf{clean}})$.

Note that checking if $|E(G)| > \binom{n}{2}$ can be done in NC^1 by Theorem 15. Moreover, for all $i \in [N]$, we have $\alpha_i = \bigvee_{j\in[N]} x_{ij}$, and therefore α_i can be computed by a circuit of depth $O(\log N)$ and $O(N)$ gates. In total, the vector α can be computed with $O(N^2)$ gates and $O(\log N)$ depth. Finally, we can check if $|\alpha|_1 > n$ in NC^1 with a threshold circuit.

For the final step, we compute G_{clean}. If $|\alpha|_1 = n$, note that $G_{\mathsf{clean}} = G[\mathrm{supp}(\alpha)]$. When $|\alpha|_1 < n$, then G_{clean} is $G[\mathrm{supp}(\alpha)]$ padded with isolated vertices. We can therefore compute G_{clean} with the circuit C^n_N of Lemma 16. Moreover, since $f \in \mathsf{NC}^i$, we have that f can be computed by a circuit of size $n^{O(1)} = N^{o(1)}$ and depth $O((\log n)^i) = O(\log N)$. Therefore, computing $f(G_{\mathsf{clean}})$ can be done in NC^1. Overall, we get that $g \in \mathsf{NC}^1$.

In order to prove (2), it suffices to modify the proof above. The modification can be briefly described as follows. We let $N = 2^{n^\varepsilon}$. Every time Lemma 16 is applied, we use the AC^0 circuit instead of the NC^1 circuit, since $n = \mathrm{polylog}(N)$. This ammounts to $N^{2+o(1)}$ many gates with unbounded fan-in. Moreover, since by assumption $f \in \mathsf{NL}$, applying Lemma 6 we obtain an AC^0 circuit for f of size $2^{n^{\varepsilon/2}} = N^{o(1)}$, so we can compute $f(G_{\mathsf{clean}})$ in constant depth with $N^{o(1)}$ gates.

Finally, for (3) it suffices to apply the same argument used for (2), replacing an application of Lemma 6 by an application of Lemma 8. ◀

4 Non-Trivial Monotone Simulations and Their Consequences

In contrast to Section 3, in this section we observe that a non-trivial simulation of AC^0 circuits by monotone circuits is possible. This follows from a refined version of the switching lemma proved by Rossman [65]. As a proof of concept, we use this simulation result to reprove a well-known AC^0 lower bound for Majority.

In the second part of this section, we show that if much faster simulations are possible, then even stronger non-monotone circuit lower bounds follow. We also show that this implication is true even if the simulation only holds for *graph properties*. Monotone simulations for graph properties are motivated by a result of Rossman [63], which shows that very strong monotone simulations are possible for *homomorphism-preserving graph properties*. The lower bounds from monotone simulations are proved with the simulation result and padding argument used in the previous section (Lemmas 6 and 12).

4.1 A non-trivial simulation for bounded-depth circuits

The earliest monotone simulation result was proved for DNFs by Quine [55].

▶ **Theorem 17** (Quine [55]). *For all $s : \mathbb{N} \to \mathbb{N}$, we have $\mathsf{DNF}[s] \cap \mathsf{Mono} \subseteq \mathsf{mDNF}[s]$.*

Proof. If a given DNF computes a monotone Boolean function, simply removing the negative literals continues to compute the same function. ◀

Let $\mathsf{DT}_{\mathsf{size}}(f)$ denote the size of a smallest decision-tree computing f. We will need a result obtained by Rossman [65].

▶ **Theorem 18** ([65]). *If $f : \{0,1\}^n \to \{0,1\}$ is computable by an AC^0 circuit of depth d and size s, then $\mathsf{DT}_{\mathsf{size}}(f) = 2^{(1-1/O(\log s)^{d-1})n}$.*

▶ **Theorem 19.** *Let $s : \mathbb{N} \to \mathbb{N}$ and $d \geq 1$. We have $\mathsf{AC}^0_d[s] \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[t]$, where $t = n \cdot 2^{n(1-1/O(\log s)^{d-1})}$. Moreover, this upper bound is achieved by monotone DNFs of size t/n.*

Proof. Let f be a monotone function computable by an AC^0 circuit of depth d and size s. By Theorem 18, there exists a decision tree of size $2^{(1-1/O(\log s)^{d-1})n}$ computing f. Therefore, there exists a DNF of the same size computing f, which can be taken to be monotone by by Theorem 17. This can be converted into a monotone circuit of size $n \cdot 2^{(1-1/O(\log s)^{d-1})n}$. ◀

We observe that it is possible to immediately deduce an AC^0 lower bound for Majority using this simulation theorem. Even though near-optimal lower bounds for Majority have been known for a long time [34] and the proof of the main technical tool (Theorem 18) behind our simulation result is similar to the one used by [34], the argument below illustrates how a monotone simulation can lead to non-monotone circuit lower bounds.

▶ **Corollary 20.** *Any depth-d AC^0 circuit computing Majority has size $2^{\Omega((n/\log n)^{1/(d-1)})}$.*

Proof. Note that Majority has $\binom{n}{n/2} = \Omega(2^n/\sqrt{n})$ minterms. Therefore, any monotone DNF computing Majority has size at least $\Omega(2^n/\sqrt{n})$. By Theorem 19, it follows that the size s of a depth-d AC^0 computing Majority satisfies the following inequality:

$$2^{n(1-1/O(\log s)^{d-1})} = \Omega(2^{n-\frac{1}{2}\log n}).$$

From this equation we obtain $s = 2^{\Omega((n/\log n)^{1/(d-1)})}$. ◀

4.2 Non-monotone lower bounds from monotone simulations

We now show that if monotone circuits are able to efficiently simulate non-monotone circuits computing monotone Boolean functions, then striking complexity separations follow. We also show a result of this kind for simulations of graph properties. We first prove a lemma connecting the simulation of AC^0 circuits with the simulation of NL machines.

▶ **Lemma 21.** *For all constants $\varepsilon > 0$ and $C \geq 1$, if $\mathsf{AC}^0 \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[2^{O((\log n)^C)}]$, then $\mathsf{NL} \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[2^{o(n^\varepsilon)}]$.*

Proof. We prove the contrapositive. Suppose that there exists $\varepsilon > 0$ such that $\mathsf{NL} \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{o(n^\varepsilon)}]$. This means that there exists a monotone function f such that $f \in \mathsf{NL}$ and any monotone circuit computing f has size $2^{\Omega(n^\varepsilon)}$.

Let $\delta = \varepsilon/(2C)$ and let $m = 2^{n^\delta}$. Let $g : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ be the Boolean function defined as $g(x,y) = f(x)$. Note that g is a function on $N := m + n = 2^{\Theta(n^\delta)}$ bits. By Lemma 6, there exists an AC^0 circuit computing f of size $2^{n^\delta} = N^{O(1)}$. Moreover, any monotone circuit computing g has size $2^{\Omega(n^\varepsilon)} = 2^{\Omega((\log N)^{\varepsilon/\delta})} = 2^{\Omega((\log N)^{2C})}$. ◀

Next, we recall the strongest known monotone circuit and formula lower bounds for a monotone function in NP.

▶ **Theorem 22** ([52])**.** $\mathsf{NP} \cap \mathsf{Mono} \not\subseteq \mathsf{mDEPTH}[o(n)]$.

▶ **Theorem 23** ([20])**.** $\mathsf{NP} \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{o(\sqrt{n}/\log n)}]$.

We are now ready to state and prove our first result regarding new complexity separations from monotone simulations. Recall that obtaining explicit lower bounds against depth-3 AC^0 circuits of size $2^{\omega(n^{1/2})}$ is a major challenge in circuit complexity theory, while the best lower bound on the size of depth-4 AC^0 circuits computing a function in NP is currently $2^{\Omega(n^{1/3})}$ [34]. Moreover, no strict separation is known in the following sequence of inclusions of complexity classes: $\mathsf{ACC} \subseteq \mathsf{TC}^0 \subseteq \mathsf{NC}^1 \subseteq \mathsf{L} \subseteq \mathsf{NL} \subseteq \oplus\mathsf{L} \subseteq \mathsf{NC}^2$. We show that efficient monotone simulations would bring new results in both of these fronts. (We stress that all lower bound consequences appearing below refer to separations against non-uniform circuits.)[14]

▶ **Theorem 24.** *Let $\mathcal{C}$ be a class of circuits. There exists $\varepsilon > 0$ such that the following holds:*

1. *If $\mathsf{AC}^0_3 \cap \mathsf{Mono} \subseteq \mathsf{mNC}^1$, then $\mathsf{NP} \not\subseteq \mathsf{AC}^0_3[2^{o(n)}]$.*
2. *If $\mathsf{AC}^0_4 \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, then $\mathsf{NP} \not\subseteq \mathsf{AC}^0_4[2^{o(\sqrt{n}/\log n)}]$.*
3. *If $\mathcal{C} \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[2^{O(n^\varepsilon)}]$, then $\mathsf{NC}^2 \not\subseteq \mathcal{C}$.*
4. *If $\mathsf{AC}^0 \cap \mathsf{Mono} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, then $\mathsf{NC}^2 \not\subseteq \mathsf{NC}^1$.*

Proof. We will prove each item separately.

Proof of 1. Let us assume that $\mathsf{AC}^0_3 \cap \mathsf{Mono} \subseteq \mathsf{mNC}^1$. Let f be the function of Theorem 22. For a contradiction, suppose that $f \in \mathsf{AC}^0_3[2^{o(n)}]$. Let $\alpha : \mathbb{N} \to \mathbb{N}$ be such that $\alpha(n) \to_n \infty$ and f has a depth-3 AC^0 circuit of size $2^{n/\alpha}$. Let $m = 2^{n/(10\cdot\alpha)}$ and let $g : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}$ be the function $g(x,y) = f(x)$. Let $N = n + m = (1 + o(1))2^{n/(10\cdot\alpha)}$. Clearly, the function g has a depth-3 AC^0 circuit of size $2^{n/\alpha} = N^{O(1)}$. Since g is monotone, we conclude from the assumption that g is computed by a polynomial-size monotone formula. Now, since $f(x) = g(x, 1^m)$, we obtain a monotone formula of size $N^{O(1)} = 2^{o(n)}$ for computing f, which contradicts the lower bound of Theorem 22.

[14] In other words, all upper bounds are *uniform*, but the lower bounds hold even for *non-uniform* circuits. Note that this is stronger than lower bounds for uniform circuits.

Proof of 2. Similar to the proof of item (1), but using Theorem 23 instead.

Proof of 3. Suppose that $\mathsf{NC}^2 \subseteq \mathcal{C}$. By Theorem 7, there exists a monotone function $f \in \mathsf{NC}^2$ on n bits and a number $\varepsilon > 0$ such that $f \notin \mathsf{mSIZE}[2^{o(n^\varepsilon)}]$. Therefore, for any $\delta > 0$ such that $\delta < \varepsilon$, we have $f \notin \mathsf{mSIZE}[2^{O(n^\delta)}]$. Since, by assumption, we have $f \in \mathsf{NC}^2 \subseteq \mathcal{C}$, we obtain $\mathcal{C} \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{O(n^\delta)}]$.

Proof of 4. If $\mathsf{NC}^2 \subseteq \mathsf{NC}^1$, then, by item (3), we get $\mathsf{NC}^1 \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[2^{o(n^\varepsilon)}]$. From Lemma 21, we obtain $\mathsf{AC}^0 \cap \mathsf{Mono} \not\subseteq \mathsf{mSIZE}[\mathsf{poly}]$. ◀

As a motivation to the ensuing discussion, we recall a result of Rossman [63], who showed that any homomorphism-preserving graph property computed by AC^0 circuits is also computed by monotone AC^0 circuits.

▶ **Theorem 25** ([63]). $\mathsf{AC}^0 \cap \mathsf{HomPreserving} \subseteq \mathsf{mDNF}[\mathsf{poly}]$.

This inspires the question of whether general graph properties can also be efficiently simulated by monotone circuits. We show that, if true, such simulations would imply strong complexity separations. Let us first recall an exponential monotone circuit lower bound for monotone graph properties, and we will be ready to state and prove our main result.

▶ **Theorem 26** ([7]). *There exists* $\varepsilon > 0$ *such that* $\mathsf{NP} \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \not\subseteq \mathsf{mSIZE}[2^{o(n^\varepsilon)}]$.

▶ **Theorem 27.** *Let* $\mathcal{C}$ *be a class of circuits. The following holds:*

1. *If* $\mathcal{C} \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, *then* $\mathsf{L} \not\subseteq \mathcal{C}$.
2. *If* $\mathcal{C} \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \subseteq \mathsf{mDEPTH}[o(\sqrt{n})]$, *where* n *denotes the number of input bits, then* $\mathsf{L} \not\subseteq \mathcal{C}$.
3. *If* $\mathsf{AC}^0 \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$, *then* $\mathsf{NP} \not\subseteq \mathsf{NC}^1$.

Proof. We will prove each item separately.

Proof of 1. Suppose that $\mathsf{L} \subseteq \mathcal{C}$. By Theorem 10, the monotone graph property $\mathsf{OddFactor}$ satisfies $\mathsf{OddFactor} \notin \mathsf{mSIZE}[\mathsf{poly}]$. Moreover, we have the upper bound $\mathsf{OddFactor} \in \mathsf{L}$ by Theorem 11. Since, by assumption, we have $\mathsf{OddFactor} \in \mathsf{L} \subseteq \mathcal{C}$, we obtain $\mathcal{C} \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \not\subseteq \mathsf{mSIZE}[\mathsf{poly}]$.

Proof of 2. Suppose that $\mathsf{L} \subseteq \mathcal{C}$. By Theorems 10 and 11, there exists a monotone graph property $f \in \mathsf{L}$ such that $f \notin \mathsf{mDEPTH}[o(\sqrt{n})]$. Since, by assumption, we have $f \in \mathsf{L} \subseteq \mathcal{C}$, we obtain $\mathcal{C} \cap \mathsf{Mono} \cap \mathsf{GraphProperties} \not\subseteq \mathsf{mDEPTH}[o(\sqrt{n})]$.

Proof of 3. Suppose that $\mathsf{NP} \subseteq \mathsf{NC}^1$. By Theorem 26, there exists a monotone graph property $f \in \mathsf{NC}^1$ such that $\mathsf{mSIZE}(f) = 2^{\Omega(n^\varepsilon)}$ for some $\varepsilon > 0$. Let $\delta = \varepsilon/2$. By Lemma 12 (Item 2), there exists a monotone graph property g on $N = 2^{n^\delta}$ vertices computed by an AC^0 circuit of size $N^{2+o(1)}$ such that f is a monotone projection of g. Theorem 26 implies that any monotone circuit computing f has size $2^{\Omega(n^\varepsilon)} = 2^{\Omega((\log N)^2)} = N^{\omega(1)}$. ◀

5 Monotone Complexity of Constraint Satisfaction Problems

In this section, we study the monotone complexity of Boolean-valued CSPs. Our goal is to classify which types of Boolean CSPs are hard for monotone circuit size and monotone circuit depth, eventually proving Theorems 4 and 5.

We will first spend some time recalling standard definitions and concepts in the theory of CSPs (Section 5.1), as well as a few results about CSPs that were proved in previous works [67, 38, 18, 19, 4] (Section 5.2). We will then prove Theorem 5 in Section 5.3, and we will finally prove Theorem 4 in Section 5.5 after proving some auxiliary results in Section 5.4.

5.1 Definitions

For a good introduction to the concepts defined below, we refer the reader to [18, 19]. We also refer the reader to Section 1.1.3 for the definition of the family of functions $\mathsf{CSP\text{-}SAT}_S$, as well as the terms *constraint application, S-formula* and *satisfiable formula*.

We denote by $p_i^n : \{0,1\}^n \to \{0,1\}$ the i-th *projection function* on n variables, whose operation is defined as $p_i^n(x) = x_i$. For a set of Boolean functions B, we denote by $[B]$ the *closure* of B, defined as follows: a Boolean function f is in $[B]$ if and only if $f \in B \cup \{\text{Identity}\}$ or if there exists $g \in B$ and $h_1, \ldots, h_k$ such that $f = g(h_1, \ldots, h_k)$, where each h_i is either a projection function or a function from $[B]$. We can equivalently define $[B]$ as the set of all Boolean functions that can be computed by circuits using the functions of B as gates. Note that $[B]$ necessarily contains an infinite number of Boolean functions, since $p_1^n \in [B]$ for every $n \in \mathbb{N}$; moreover, the constant functions are not necessarily in $[B]$. We say that B is a *clone* if $B = [B]$. A few prominent examples of clones are the set of all Boolean functions (equal to $[\{\wedge, \neg\}]$), monotone functions (equal to $[\{\wedge, \vee, 0, 1\}]$), and linear functions (equal to $[\{\oplus, 1\}]$).

▶ Remark 28. The set of all clones forms a lattice, known as *Post's lattice*, under the operations $[A] \sqcap [B] := [A] \cap [B]$ and $[A] \sqcup [B] := [A \cup B]$. From the next section onwards, we will refer to the clones defined in [18] (such as I_0, I_1, etc.), assuming the reader is familiar with them. For the unfamiliar reader, we refer to Appendix C and Figures 1 and 4, which contain all the definitions of the clones we will need, as well as the entire Post's lattice in graphical representation.

To avoid confusion, we will always refer to clones with normal-Roman font (e.g., $\mathrm{S}_1, \mathrm{I}_0$, etc).

Let S be a finite set of Boolean relations. We denote by $\mathsf{CNF}(S)$ the set of all S-formulas. We denote by $\mathsf{COQ}(S)$ the set of all relations which can be expressed with the following type of formula φ:

$$\varphi(x_1, \ldots, x_k) = \exists y_1, \ldots, y_\ell \, \psi(x_1, \ldots, x_k, y_1, \ldots, y_\ell),$$

where $\psi \in \mathsf{CNF}(S)$. The relations in $\mathsf{COQ}(S)$ will also be referred as *conjunctive queries* over S. We denote by $\langle S \rangle$ the set of relations defined as $\langle S \rangle := \mathsf{COQ}(S \cup \{=\})$. If $S = \langle S \rangle$, we say that S is a *co-clone*. We define

$$\mathsf{CSP} = \{\mathsf{CSP\text{-}SAT}_S : S \text{ is a finite set of relations}\}.$$

We say that $\mathsf{CSP\text{-}SAT}_S$ is *trivial* if $\mathsf{CSP\text{-}SAT}_S$ is a constant function.

Let R be a k-ary Boolean relation and let $f : \{0,1\}^\ell \to \{0,1\}$ be a Boolean function. For $x \in R$ and $i \in [k]$, we denote by $x[i]$ the i-th bit of x.

▶ **Definition 29.** *We say that f is a* polymorphism *of R, and R is* an invariant of f*, if, for all $x_1, \ldots, x_\ell \in R$, we have*

$$(f(x_1[1], \ldots, x_\ell[1]), f(x_2[2], \ldots, x_\ell[2]), \ldots, f(x_k[k], \ldots, x_\ell[k])) \in R.$$

We denote the set of all polymorphisms of R by $\mathsf{Pol}(R)$. For a set of relations S, we denote by $\mathsf{Pol}(S)$ the set of Boolean functions which are polymorphisms of all the relations of S. For a set of Boolean functions, we denote by $\mathsf{Inv}(B)$ the set of all Boolean relations which are invariant under all functions of B (i.e., $\mathsf{Inv}(B) = \{R : B \subseteq \mathsf{Pol}(R)\}$).

The following summarises the important facts about clones, co-clones and polymorphisms that are relevant to the study of CSPs [39].

▶ **Lemma 30.** *Let S and S' be sets of Boolean relations and let B and B' be sets of Boolean functions. We have*

(i) $\mathsf{Pol}(S)$ *is a clone and* $\mathsf{Inv}(B)$ *is a co-clone;*
(ii) *If* $S \subseteq S'$*, then* $\mathsf{Pol}(S') \subseteq \mathsf{Pol}(S)$*;*
(iii) *If* $B \subseteq B'$*, then* $\mathsf{Inv}(B') \subseteq \mathsf{Inv}(B)$*;*
(iv) $\mathsf{COQ}(\mathsf{COQ}(S)) = \mathsf{COQ}(S)$*;*
(v) *If* $S \subseteq S'$*, then* $\mathsf{COQ}(S) \subseteq \mathsf{COQ}(S')$*;*
(vi) $\mathsf{Inv}(\mathsf{Pol}(S)) = \langle S \rangle$*;*
(vii) $\mathsf{Pol}(\mathsf{Inv}(B)) = [B]$.

We now define different types of reductions. We say that a reduction is a *monotone OR-reduction* if every bit of the reduction is either constant or can be computed by a monotone disjunction on the input variables. We write $f \leq_m^{\mathsf{mOR}} g$ if there exists a many-one monotone OR-reduction from f to g. We also write $f \leq_m^{\mathsf{AC}^0} g$ if there exists a many-one AC^0 reduction from f to g, and $f \leq_m^{\mathsf{mNL}} g$ if there exists a many-one mNL reduction from f to g[15]. Unless otherwise specified, every reduction we consider will generate an instance of polynomial size on the length of the input.

Finally, we denote by OR^k and NAND^k the k-ary OR and NAND relations, respectively.

5.2 Basic facts about CSP-SAT

We state here basic facts about the CSP-SAT function. These facts are proved in the original paper of Schaefer [67], as well as in later papers [38, 18, 19, 4].

Lemma 31 below is one of the most important lemmas of this section and will be used many times. It states that $\mathsf{Pol}(S)$ characterises the monotone complexity of $\mathsf{CSP\text{-}SAT}_S$, in the sense that the sets of relations with few polymorphisms give rise to the hardest instances of CSPs. A non-monotone version of this result was proved in [38, 19, Theorem 2.4], and we check in Appendix B.2 that their proofs also hold in the monotone case.

▶ **Lemma 31** (Polymorphisms characterise the complexity of CSPs [38, 19, Theorem 2.4])**.** *If* $\mathsf{Pol}(S_2) \subseteq \mathsf{Pol}(S_1)$*, then* $\mathsf{CSP\text{-}SAT}_{S_1}^n \leq_m^{\mathsf{mNL}} \mathsf{CSP\text{-}SAT}_{S_2}^{\mathsf{poly}(n)}$.

Theorem 32 gives monotone circuit upper bounds for some instances of $\mathsf{CSP\text{-}SAT}_S$. Non-monotone variants of this upper bound were originally obtained in the seminal paper of Schaefer [67], and we again check that the monotone variants work in Appendix B.3.

▶ **Theorem 32** (Monotone version of the upper bounds for CSP-SAT [67, 4])**.** *Let S be a finite set of relations. The following holds.*

1. *If* $\mathrm{E}_2 \subseteq \mathsf{Pol}(S)$ *or* $\mathrm{V}_2 \subseteq \mathsf{Pol}(S)$*, then* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mSIZE}[\mathsf{poly}]$.
2. *If* $\mathrm{D}_2 \subseteq \mathsf{Pol}(S)$*, or* $\mathrm{S}_{00} \subseteq \mathsf{Pol}(S)$*, or* $\mathrm{S}_{10} \subseteq \mathsf{Pol}(S)$*, then* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNL}$.

Finally, we state here a result of [4], which classifies the *non-monotone* complexity of $\mathsf{CSP\text{-}SAT}_S$ under $\leq_m^{\mathsf{AC}^0}$ reductions. The classification of the complexity of $\mathsf{CSP\text{-}SAT}_S$ is based solely on $\mathsf{Pol}(S)$. See Figure 1 for a graphical representation.

[15] A many-one AC^0 (resp. mNL) reduction is one in which each bit of the reduction is either constant or can be computed with a polynomial-size AC^0 circuit (resp. monotone nondeterministic branching program). Recall that a *monotone nondeterministic branching program* is a directed acyclic graph G with two distinguished vertices s and t, in which each edge e is labelled with an input function $\rho_e \in \{1, x_1, \ldots, x_n\}$. Given an input x, the program accepts if there exists a path from s to t in the subgraph G_x of G in which an edge e appears if $\rho_e(x) = 1$.

▶ **Theorem 33** (Refined classification of CSP problems [4, Theorem 3.1]). *Let S be a finite set of Boolean relations. The following holds.*

- *If* $\mathrm{I}_0 \subseteq \mathsf{Pol}(S)$ *or* $\mathrm{I}_1 \subseteq \mathsf{Pol}(S)$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is trivial.*
- *If* $\mathsf{Pol}(S) \in \{\mathrm{I}_2, \mathrm{N}_2\}$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\leq_m^{\mathsf{AC}^0}$*-complete for* NP*.*
- *If* $\mathsf{Pol}(S) \in \{\mathrm{V}_2, \mathrm{E}_2\}$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\leq_m^{\mathsf{AC}^0}$*-complete for* P*.*
- *If* $\mathsf{Pol}(S) \in \{\mathrm{L}_2, \mathrm{L}_3\}$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\leq_m^{\mathsf{AC}^0}$*-complete for* $\oplus\mathsf{L}$*.*
- *If* $\mathrm{S}_{00} \subseteq \mathsf{Pol}(S) \subseteq {\mathrm{S}_{00}}^2$ *or* $\mathrm{S}_{10} \subseteq \mathsf{Pol}(S) \subseteq {\mathrm{S}_{10}}^2$ *or* $\mathsf{Pol}(S) \in \{\mathrm{D}_2, \mathrm{M}_2\}$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\leq_m^{\mathsf{AC}^0}$*-complete for* NL*.*
- *If* $\mathsf{Pol}(S) \in \{\mathrm{D}_1, \mathrm{D}\}$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\leq_m^{\mathsf{AC}^0}$*-complete for* L*.*
- *If* $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$ *or* $\mathrm{S}_{12} \subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$*, then either* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{AC}^0$ *or* $\mathsf{CSP\text{-}SAT}_S$ *is* $\leq_m^{\mathsf{AC}^0}$*-complete for* L*.*

5.3 A monotone dichotomy for CSP-SAT

In this section, we prove Theorem 5. We first prove Part (1) of the theorem (the dichotomy for circuit size), and then we prove Part (2) of the theorem (the dichotomy for circuit depth).

Dichotomy for circuits. To prove the dichotomy for circuits, we first show that, for any set of relations S whose set of polymorphisms is contained in L_3, we can monotonically reduce 3-XOR-SAT to $\mathsf{CSP\text{-}SAT}_S$.

▶ **Lemma 34.** *Let S be a finite set of relations. If* $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$*, then* 3-XOR-SAT $\leq_m^{\mathsf{mNL}}$ $\mathsf{CSP\text{-}SAT}_S$*.*

Proof. Inspecting Post's lattice (Figure 1), note that the only clones strictly contained in L_3 are $\mathrm{L}_2, \mathrm{N}_2$ and I_2. We will first show that the reduction holds for the case $\mathsf{Pol}(S) = \mathrm{L}_2$ and then prove that the reduction also holds for the case $\mathsf{Pol}(S) = \mathrm{L}_3$. Lemma 31 will then imply the cases $\mathsf{Pol}(S) \in \{\mathrm{N}_2, \mathrm{I}_2\}$, since $\mathrm{I}_2 \subseteq \mathrm{N}_2 \subseteq \mathrm{L}_3$.

It's not hard to check that, if $\mathsf{Pol}(S) = \mathrm{L}_2$, then $\mathsf{Pol}(S) \subseteq \mathsf{Pol}(\text{3-XOR-SAT})$ (it suffices to observe that bitwise XORing three satisfying assignments to a linear equation gives rise to a new satisfying assignment to the same equation). Therefore, from Lemma 31 we deduce that 3-XOR-SAT admits a reduction to $\mathsf{CSP\text{-}SAT}_S$ in mNL. In order to prove the case $\mathsf{Pol}(S) = \mathrm{L}_3$, we first prove the following claim.

▷ Claim ([4, Lemma 3.11]). Let S be a finite set of relations such that $\mathsf{Pol}(S) = \mathrm{L}_2$. There exists a finite set of relations S' such that $\mathsf{Pol}(S') = \mathrm{L}_3$ and $\mathsf{CSP\text{-}SAT}_S^n \leq_m^{\mathsf{mProj}} \mathsf{CSP\text{-}SAT}_{S'}^{n+1}$.

Proof. We describe the proof of Lemma 3.11 in [4] and observe that it gives a monotone reduction.

For a relation $R \in S$, let $R' = \{(\neg x_1, \ldots, \neg x_k) : (x_1, \ldots, x_k) \in R\}$. Let also $S' = \{R' : R \in S\}$. It's not hard to check that $\mathsf{Pol}(S') = \mathrm{L}_3$, since S' is an invariant of L_2 and N_2, and L_3 is the smallest clone containing both L_2 and N_2; moreover, if $\rho \in \mathsf{Pol}(S')$ and ρ is a Boolean function on at least two bits, then $\rho \in \mathsf{Pol}(S) = \mathrm{L}_2$.

Now let F be a instance of $\mathsf{CSP\text{-}SAT}_S^n$. For every constraint $C = R(x_1, \ldots, x_k)$ in F, we add the constraint $C' = R'(\alpha, x_1, \ldots, x_k)$ to the S'-formula F', where α is a new variable. Note that F' is a S'-formula, defined on $n+1$ variables, which is satisfiable if and only if F is satisfiable. Moreover, the construction of F' from F can be done with a monotone projection. ◁

Since the case $\mathsf{Pol}(S) = \mathrm{L}_2$ holds, the case $\mathsf{Pol}(S) = \mathrm{L}_3$ now follows from Lemma 31 and the Claim. Finally, from Lemma 31 we conclude that the reduction also holds for the case $\mathsf{Pol}(S) \in \{\mathrm{N}_2, \mathrm{I}_2\}$, since $\mathrm{I}_2 \subseteq \mathrm{N}_2 \subseteq \mathrm{L}_3$. ◀

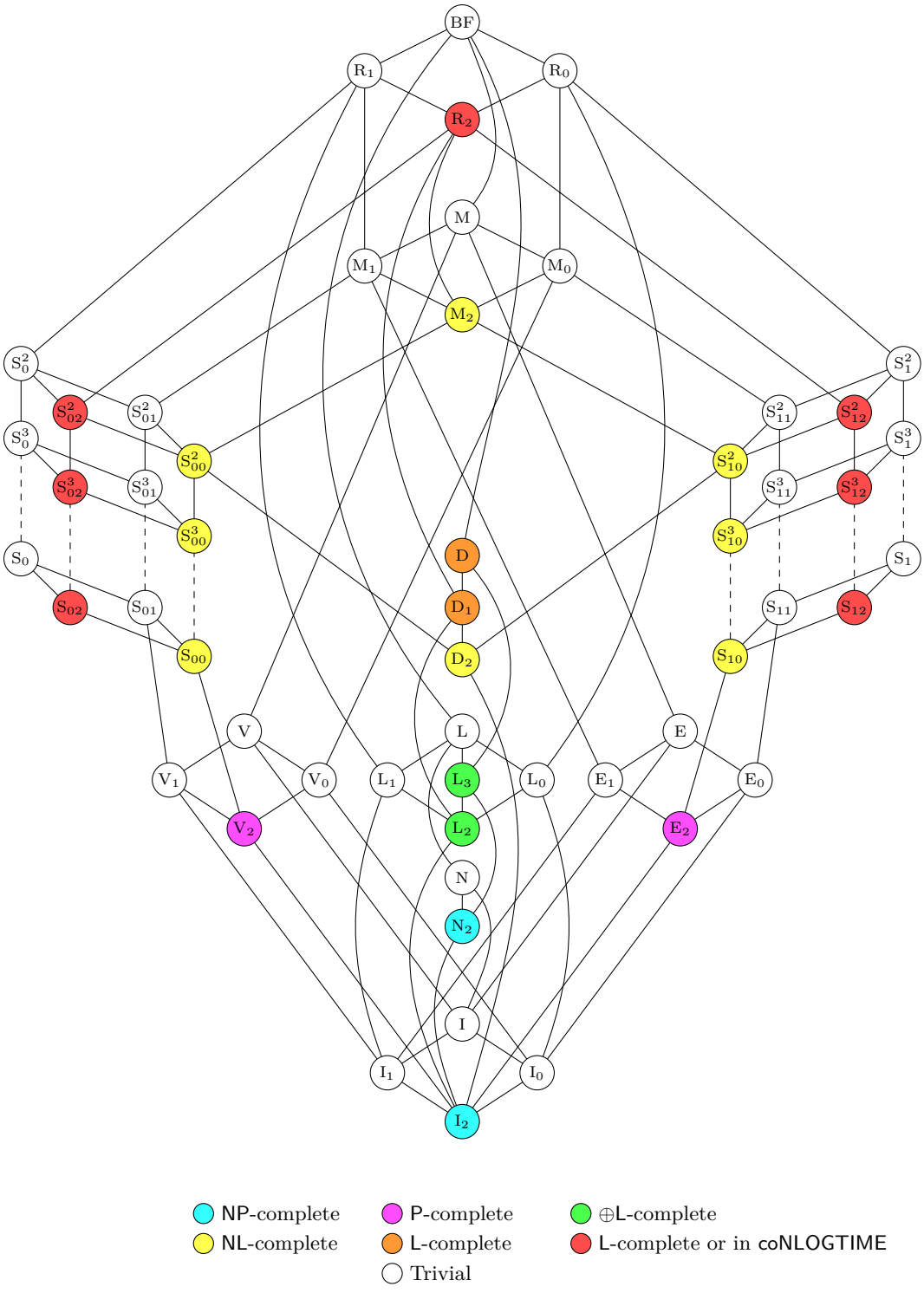


Figure 1 Graph of all closed classes of Boolean functions. The vertices are colored with the complexity of deciding CSPs whose set of polymorphisms corresponds to the label of the vertex. Trivial CSPs are those that correspond to constant functions. Every hardness result is proved under $\leq_m^{\mathsf{AC}^0}$ reductions. See Theorem 33 for details. A similar figure appears in [4, Figure 1].

▶ **Theorem 35** (Dichotomy for monotone circuits). *Let S be a finite set of relations. If* $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$ *then there exists a constant* $\varepsilon > 0$ *such that* $\mathsf{mSIZE}(\mathsf{CSP\text{-}SAT}_S) = 2^{\Omega(n^\varepsilon)}$. *Otherwise, we have* $\mathsf{mSIZE}(\mathsf{CSP\text{-}SAT}_S) = n^{O(1)}$.

Proof. If $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$, the lower bound follows from the 'moreover' part of Theorem 7, and Lemma 34. For the upper bound, we inspect Post's lattice (Figure 1). Observe that, if $\mathsf{Pol}(S) \not\subseteq \mathrm{L}_3$, the following are the only possible cases:

1. $\mathrm{I}_0 \subseteq \mathsf{Pol}(S)$ or $\mathrm{I}_1 \subseteq \mathsf{Pol}(S)$. In both cases, any $\mathsf{CNF}(S)$ is trivially satisfiable.
2. $\mathrm{E}_2 \subseteq \mathsf{Pol}(S)$ or $\mathrm{V}_2 \subseteq \mathsf{Pol}(S)$. In this case, $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mSIZE}[\mathsf{poly}]$ by Theorem 32.
3. $\mathrm{D}_2 \subseteq \mathsf{Pol}(S)$. In this case, $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNL} \subseteq \mathsf{mSIZE}[\mathsf{poly}]$ by Theorem 32. ◀

▶ Remark 36. We remark that the lifting theorem of [31] (which is an ingredient in the proof of Theorem 7) is only used to prove that the monotone complexity of $\mathsf{CSP\text{-}SAT}_S$ is exponential when $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$. If we only care to show a superpolynomial separation, then it suffices to apply the superpolynomial lower bound for CSPs with counting proved in [27, 9] using the approximation method. Indeed, we give an explicit proof in Appendix A. The same holds for the consequences of this theorem (see Theorem 46).

Dichotomy for formulas. Define $\mathsf{3\text{-}Horn\text{-}SAT}_n : \{0,1\}^{2n^3+n} \to \{0,1\}$ as $\mathsf{3\text{-}Horn\text{-}SAT}_n = \mathsf{CSP\text{-}SAT}^n_{\mathcal{H}^3}$, where

$$\mathcal{H}^3 = \{(\neg x_1 \vee \neg x_2 \vee x_3), (\neg x_1 \vee \neg x_2 \vee \neg x_3), (x)\}\,.$$

The following is proved in [56, 31].

▶ **Theorem 37** ([56, 31]). *There exists* $\varepsilon > 0$ *such that* $\mathsf{3\text{-}Horn\text{-}SAT} \in \mathsf{mSIZE}[\mathsf{poly}] \setminus \mathsf{mDEPTH}[o(n^\varepsilon)]$.

Proof sketch. Since $\mathrm{E}_2 \subseteq \mathsf{Pol}(\mathcal{H}^3)$ (see, e.g., [25, Lemma 4.8]), the upper bound follows from Theorem 32. The lower bound follows from a lifting theorem of [56, 31]. They show that the monotone circuit-depth of $\mathsf{3\text{-}Horn\text{-}SAT}$ is at least the depth of the smallest Resolution-tree refuting a so-called *pebbling formula*. Since this formula requires Resolution-trees of depth n^ε, the lower bound follows. ◀

Analogously to the previous section, we show that $\mathsf{3\text{-}Horn\text{-}SAT}$ reduces to $\mathsf{CSP\text{-}SAT}_S$ whenever $\mathsf{Pol}(S)$ is small enough, in a precise sense stated below. We then deduce the dichotomy for formulas with a similar argument.

▶ **Lemma 38.** *Let S be a finite set of relations. If* $\mathsf{Pol}(S) \subseteq \mathrm{E}_2$ *or* $\mathsf{Pol}(S) \subseteq \mathrm{V}_2$, *then* $\mathsf{3\text{-}Horn\text{-}SAT} \leq^{\mathsf{mNL}}_m \mathsf{CSP\text{-}SAT}_S$.

Proof. We first consider the case $\mathsf{Pol}(S) \subseteq \mathrm{E}_2$. Note that $\mathrm{E}_2 \subseteq \mathsf{Pol}(\mathsf{3\text{-}Horn\text{-}SAT})$ (see, e.g., [25, Lemma 4.8]). Therefore, from Lemma 31 we deduce that $\mathsf{3\text{-}Horn\text{-}SAT}$ admits a reduction to $\mathsf{CSP\text{-}SAT}_S$ in mNL.

Now let $\mathsf{3\text{-}AntiHorn\text{-}SAT} = \mathsf{CSP\text{-}SAT}_{\mathcal{A}^3}$, where $\mathcal{A}^3 = \{(x_1 \vee x_2 \vee \neg x_3), (x_1 \vee x_2 \vee x_3), (\neg x)\}$. Observe that a $\mathcal{H}^3$-formula φ is satisfiable if and only if the $\mathcal{A}^3$-formula $\varphi(\neg x_1, \ldots, \neg x_n)$ is satisfiable. Therefore, we have $\mathsf{3\text{-}Horn\text{-}SAT} \leq^{\mathsf{mProj}}_m \mathsf{3\text{-}AntiHorn\text{-}SAT}$. Observing that $\mathrm{V}_2 \subseteq \mathsf{Pol}(\mathcal{A}^3)$ (again, see e.g. [25, Lemma 4.8]), the result now follows from Lemma 31 and the previous paragraph. ◀

▶ **Theorem 39** (Dichotomy for monotone formulas). *Let S be a finite set of relations. If* $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$, *or* $\mathsf{Pol}(S) \subseteq \mathrm{V}_2$, *or* $\mathsf{Pol}(S) \subseteq \mathrm{E}_2$, *then there is a constant* $\varepsilon > 0$ *such that* $\mathsf{mDEPTH}(\mathsf{CSP\text{-}SAT}_S) = \Omega(n^\varepsilon)$. *Otherwise, we have* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNL} \subseteq \mathsf{mNC}^2 \subseteq \mathsf{mDEPTH}[\log^2 n]$.

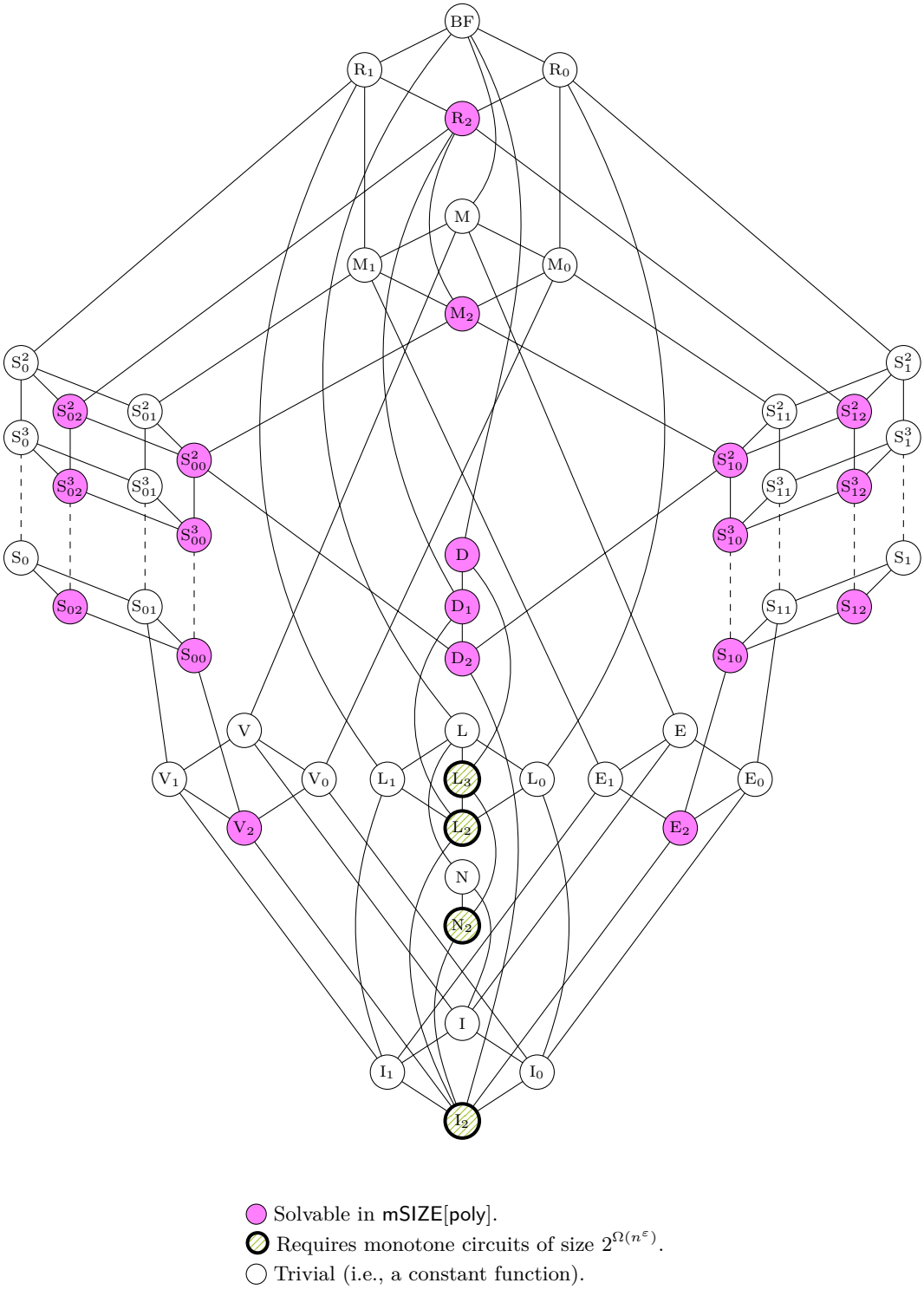


Figure 2 Illustration of Theorem 35. The vertices are colored with the *monotone circuit size* complexity of deciding CSPs whose set of polymorphisms corresponds to the label of the vertex.

Proof. We will first prove the lower bound. If $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$, the lower bound follows from Theorem 35. If $\mathsf{Pol}(S) \subseteq \mathrm{V}_2$ or $\mathsf{Pol}(S) \subseteq \mathrm{E}_2$, the lower bound follows from Theorem 37 and Lemma 38.

By inspecting Post's lattice (Remark 28), we see that the remaning cases are:

1. $\mathrm{I}_0 \subseteq \mathsf{Pol}(S)$ or $\mathrm{I}_1 \subseteq \mathsf{Pol}(S)$. In both cases, any $\mathsf{CNF}(S)$ is trivially satisfiable.
2. $\mathrm{S}_{00} \subseteq \mathsf{Pol}(S)$, or $\mathrm{S}_{10} \subseteq \mathsf{Pol}(S)$, or $\mathrm{D}_2 \subseteq \mathsf{Pol}(S)$. In all of those three cases, we have $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNL}$ by Theorem 32. ◀

5.4 Some auxiliary results

In this section, we prove auxiliary results needed in the proof of a more general form of Theorem 4. In particular, we will prove that all $\mathsf{CSP\text{-}SAT}_S$ which are in AC^0 are also contained in $\mathsf{mAC}^0 \subseteq \mathsf{mNC}^1$. Moreover, we show that, if $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mNC}^1$, then $\mathsf{CSP\text{-}SAT}_S$ is L-hard under $\leq_m^{\mathsf{AC}^0}$ reductions.

We first observe that, when $\mathsf{COQ}(S_1) \subseteq \mathsf{COQ}(S_2)$, there exists an efficient low-depth reduction from $\mathsf{CSP\text{-}SAT}_{S_1}$ to $\mathsf{CSP\text{-}SAT}_{S_2}$. This reduction, which will be useful in this section, is more refined than the one given by Lemma 31. A proof of the non-monotone version of this statement is found in [19, Proposition 2.3], and we give a monotone version of this proof in Appendix B.2.

▶ **Lemma 40** ([19, Proposition 2.3]). *If* $\mathsf{COQ}(S_1) \subseteq \mathsf{COQ}(S_2)$*, then there exists a constant* $C \in \mathbb{N}$ *such that* $\mathsf{CSP\text{-}SAT}^n_{S_1} \leq_m^{\mathsf{mOR}} \mathsf{CSP\text{-}SAT}^{Cn}_{S_2}$.

Proof. We defer the proof to Appendix B.2. ◀

We now recall some lemmas from [4], and prove a few consequences from them. We say that a set S of relations *can express equality* if $\{=\} \subseteq \mathsf{COQ}(S)$.

▶ **Lemma 41** ([4]). *Let* S *be a finite set of relations. Suppose* $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S)$ *(*$\mathrm{S}_{12} \subseteq \mathsf{Pol}(S)$*, resp.) and that* S *cannot express equality. Then there exists* $k \geq 2$ *such that* $S \subseteq \mathsf{COQ}(\left\{\mathsf{OR}^k, x, \neg x\right\})$ *(*$S \subseteq \mathsf{COQ}(\left\{\mathsf{NAND}^k, x, \neg x\right\})$*, resp.).*

Proof. Follows from the proof of Lemma 3.8 of [4]. ◀

▶ **Lemma 42.** *Let* S *be a finite set of relations such that* $\mathsf{Pol}(S) \subseteq \mathrm{R}_2$*. If* $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S)$ *or* $\mathrm{S}_{12} \subseteq \mathsf{Pol}(S)$*, and* S *cannot express equality, then* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mAC}^0_3$.

Proof. We write the proof in the case $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S)$. The other case is analogous.

From Lemmas 40 and 41 and Items iv and v of Lemma 30, we get that there is a monotone OR-reduction from $\mathsf{CSP\text{-}SAT}_S$ to $\mathsf{CSP\text{-}SAT}_{\left\{\mathsf{OR}^k, x, \neg x\right\}}$ for some k. However, an $\left\{\mathsf{OR}^k, x, \neg x\right\}$-formula is unsatisfiable iff there exists a literal and its negation as a constraint in the formula, or if there exists a disjunction in the formula such that every one of its literals appears negatively as a constraint. This condition can be easily checked by a polynomial-size monotone DNF. Composing the monotone DNF with the monotone OR-reduction, we obtain a depth-3 AC^0 circuit computing $\mathsf{CSP\text{-}SAT}_S$. ◀

▶ **Lemma 43** ([4, Lemma 3.8]). *Let* S *be a finite set of relations such that* $\mathsf{Pol}(S) \subseteq \mathrm{R}_2$*. If* $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S)$ *or* $\mathrm{S}_{12} \subseteq \mathsf{Pol}(S)$*, and* S *can express equality, then* $\mathsf{CSP\text{-}SAT}_S$ *is* L*-hard under* $\leq_m^{\mathsf{AC}^0}$ *reductions.*

▶ **Lemma 44.** *Let* S *be a finite set of relations. If* $\mathrm{S}_{02} \not\subseteq \mathsf{Pol}(S)$ *and* $\mathrm{S}_{12} \not\subseteq \mathsf{Pol}(S)$*, then* $\mathsf{CSP\text{-}SAT}_S$ *is* L*-hard or trivial.*

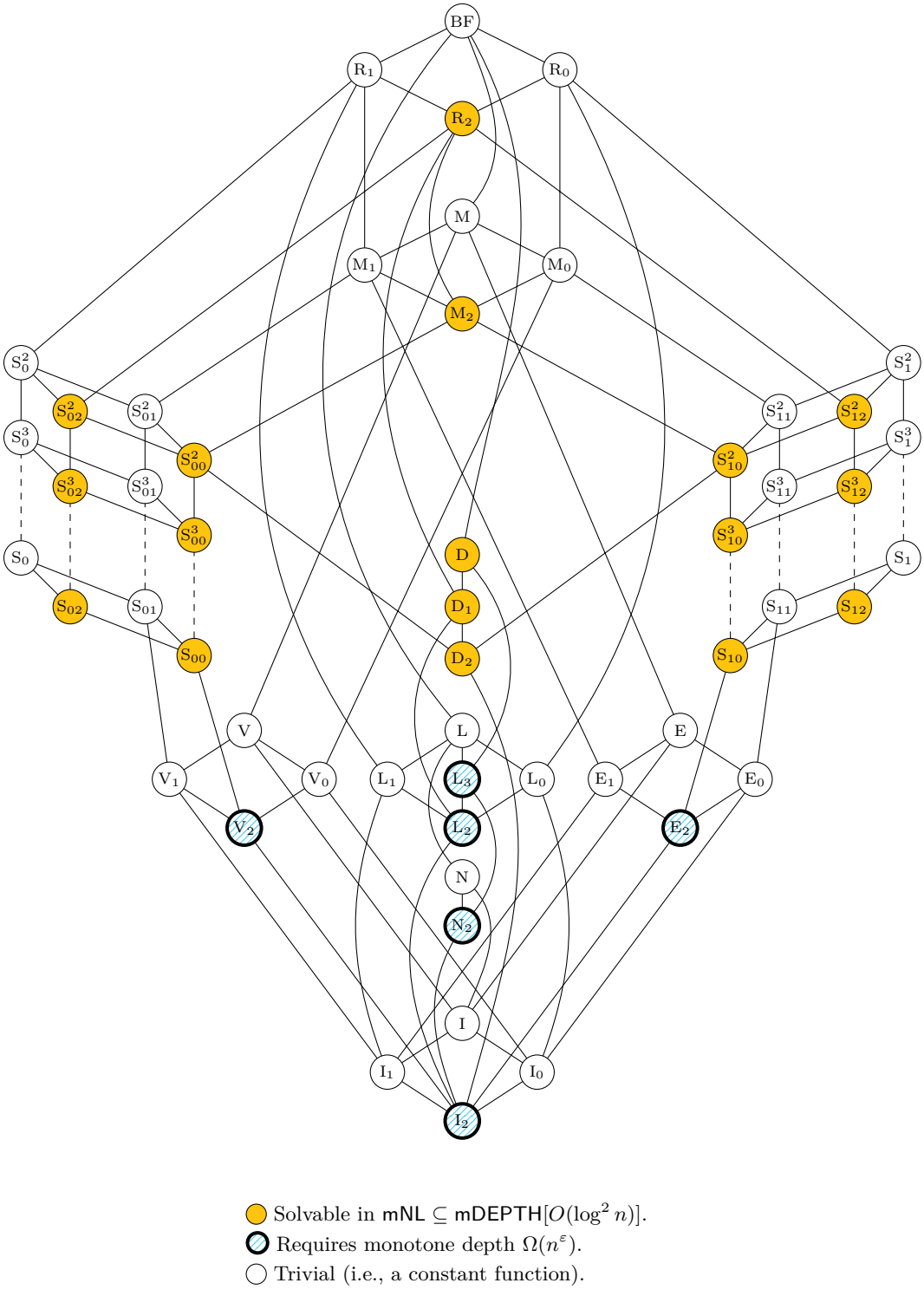


Figure 3 Illustration of Theorem 39. The vertices are colored with the *monotone circuit depth* complexity of deciding CSPs whose set of polymorphisms corresponds to the label of the vertex.

Proof. This follows by inspecting Post's lattice (Figure 1) and the classification theorem (Theorem 33). ◀

We may now prove the main result of this subsection.

▶ **Theorem 45.** *We have* $\mathsf{CSP} \cap \mathsf{AC}^0 \subseteq \mathsf{mAC}^0_3$. *Moreover, if* $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mAC}^0_3$, *then* $\mathsf{CSP\text{-}SAT}_S$ *is* L*-hard under* $\leq^{\mathsf{AC}^0}_m$ *reductions.*

Proof. Let S be a finite set of relations. If $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mAC}^0_3$, then, by Lemma 42, at least one of the following cases hold:

1. $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$ or $\mathrm{S}_{12} \subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$, and S can express the equality relation;
2. $\mathrm{S}_{02} \not\subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$ and $\mathrm{S}_{12} \not\subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$.
3. $\mathsf{Pol}(S) \not\subseteq \mathrm{R}_2$.

Since $\mathsf{CSP\text{-}SAT}_S$ is not trivial, we obtain that $\mathsf{CSP\text{-}SAT}_S$ is L-hard in the first two cases by Lemmas 43 and 44, and it's easy to check that $\mathsf{CSP\text{-}SAT}_S$ is also L-hard in the third case by inspecting Post's lattice (Figure 1) and the classification theorem (Theorem 33). Since $\mathsf{L} \not\subseteq \mathsf{AC}^0$, this also implies that, if $\mathsf{CSP\text{-}SAT}_S \in \mathsf{AC}^0$, then $\mathrm{S}_{02} \subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$ or $\mathrm{S}_{12} \subseteq \mathsf{Pol}(S) \subseteq \mathrm{R}_2$, and S cannot express the equality relation. Lemma 42 again gives $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mAC}^0_3$. ◀

5.5 Consequences for monotone circuit lower bounds via lifting

We now prove a stronger form of Theorem 4. In the previous section, we showed that $\mathsf{CSP} \cap \mathsf{AC}^0 \subseteq \mathsf{mAC}^0$. In particular, this means that there does not exist a finite set of relations S such that $\mathsf{CSP\text{-}SAT}_S$ separates AC^0 and mNC^1, a separation which we proved in Theorem 1. We will also observe that, if $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mNC}^2$, then $\mathsf{CSP\text{-}SAT}_S$ is $\oplus\mathsf{L}$-hard.

▶ **Theorem 46.** *Let* S *be a finite set of Boolean relations.*

1. *If* $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mAC}^0_3$ *then* $\mathsf{CSP\text{-}SAT}_S$ *is* L*-hard under* $\leq^{\mathsf{AC}^0}_m$ *reductions.*
2. *If* $\mathsf{CSP\text{-}SAT}_S \notin \mathsf{mNC}^2$, *then* $\mathsf{CSP\text{-}SAT}_S$ *is* $\oplus\mathsf{L}$*-hard under* $\leq^{\mathsf{AC}^0}_m$ *reductions.*

Proof. Item (1) follows from Theorem 45. To prove item (2), suppose that $\mathsf{mDEPTH}(\mathsf{CSP\text{-}SAT}_S) = \omega(\log^2 n)$. Then, by Theorem 39, we conclude that $\mathsf{Pol}(S) \subseteq \mathrm{L}_3$, or $\mathsf{Pol}(S) \subseteq \mathrm{V}_2$, or $\mathsf{Pol}(S) \subseteq \mathrm{E}_2$. Theorem 33 implies that $\mathsf{CSP\text{-}SAT}_S$ is $\oplus\mathsf{L}$-hard. ◀

Further Discussion. We recall the discussion of Section 1.1.3. We introduced and defined the functions $\mathsf{CSP\text{-}SAT}_S$ in that section, as a way to capture monotone circuit lower bounds proved via lifting. This in particular captures the monotone function 3-XOR-SAT, which was proved in [31] to require monotone circuit lower bounds of size $2^{n^{\Omega(1)}}$ to compute, even though $\oplus\mathsf{L}$-machines running in polynomial-time can compute it. Theorem 46 proves that this separation between monotone and non-monotone circuit lower bounds cannot be improved by varying the set of relations S, as we argue below.

There are two ways one could try to find a function in AC^0 with large monotone complexity using a CSP-SAT function. First, one could try to define a set of relations S such that $\mathsf{CSP\text{-}SAT}_S \in \mathsf{AC}^0$, but the monotone complexity of $\mathsf{CSP\text{-}SAT}_S$ is large. However, Item (1) of Theorem 46 proves that this is impossible, as any CSP-SAT function outside of mAC^0 is L-hard under simple reductions and, therefore, cannot be computed in AC^0.

Secondly, one could try to be apply the arguments of Section 3, consisting of a padding trick and a simulation theorem. When S is the set of 3XOR relations, then indeed we obtain a function in $\mathsf{AC}^0[\oplus]$ with superpolynomial monotone circuit complexity, as proved in Theorem 7. However, Item (2) of Theorem 46 proves that this is best possible, as any

CSP-SAT function which admits a superpolynomial monotone circuit lower bound must be ⊕L-hard and, therefore, at least as hard as 3-XOR-SAT for non-monotone circuits. Item (2) also shows that even CSP-SAT functions with a $\omega(\log^2 n)$ monotone depth lower bound must be ⊕L-hard, which suggests that the arguments of Section 3 applied to a CSP-SAT function are not able to prove the separation of Theorem 9.

A caveat to these impossibility results is in order. First, we only study Boolean-valued CSPs here, though the framework of lifting can also be applied in the context of non-Boolean CSPs. It's not clear if non-Boolean CSPs exhibit the same dichotomies for monotone computation we proved in this section. We remark that Schaefer's dichotomy for Boolean-valued CSPs [67] has been extended to non-Boolean CSPs [72, 15].

Secondly, the instances of CSP-SAT generated by lifting do not cover the entirety of the minterms and maxterms of CSP-SAT. In particular, our results do not rule out the possibility that a clever interpolation of the instances generated by lifting may give rise to a function that is easier to compute by non-monotone circuits, and therefore bypasses the hardness results of Theorem 46. One example is the Tardós function [69]. A lifting theorem applied to a Pigeonhole Principle formula can be used to prove a lower bound on the size of monotone circuits that accept cliques of size k and reject graphs that are $(k-1)$-colorable, for some $k = n^{\varepsilon}$ [56, 61]. A natural interpolation for these instances would be the k-Clique function, which, being NP-complete, would be related to an NP-complete CSP-SAT. However, as proved by [69], there is a monotone function in P which has the same output behaviour over these instances.

References

1. Miklós Ajtai and Yuri Gurevich. Monotone versus positive. *J. Assoc. Comput. Mach.*, 34(4):1004–1015, 1987. doi:10.1145/31846.31852.
2. Miklós Ajtai, János Komlós, and Endre Szemerédi. An $O(n \log n)$ sorting network. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing, 25-27 April, 1983, Boston, Massachusetts, USA*, pages 1–9. ACM, 1983. doi:10.1145/800061.808726.
3. Jin Akiyama and Mikio Kano. *Factors and Factorizations of Graphs*, volume 2031 of *Lecture Notes in Mathematics*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. doi:10.1007/978-3-642-21919-1.
4. Eric Allender, Michael Bauland, Neil Immerman, Henning Schnoor, and Heribert Vollmer. The complexity of satisfiability problems: Refining Schaefer's theorem. *J. Comput. Syst. Sci.*, 75(4):245–254, 2009. doi:10.1016/j.jcss.2008.11.001.
5. Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing disjunctive normal form formulas and AC^0 circuits given a truth table. *SIAM J. Comput.*, 38(1):63–84, 2008. doi:10.1137/060664537.
6. Eric Allender, Michal Koucký, Detlef Ronneburger, Sambuddha Roy, and V. Vinay. Time-space tradeoffs in the counting hierarchy. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity, Chicago, Illinois, USA, June 18-21, 2001*, pages 295–302. IEEE Computer Society, 2001. doi:10.1109/CCC.2001.933896.
7. Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7(1):1–22, 1987.
8. Alexander E Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. *Doklady Akademii Nauk SSSR*, 282:1033–1037, 1985.
9. László Babai, Anna Gál, and Avi Wigderson. Superpolynomial lower bounds for monotone span programs. *Combinatorica*, 19(3):301–319, 1999. doi:10.1007/s004930050058.
10. Josh Cohen Benaloh and Jerry Leichter. Generalized secret sharing and monotone functions. In *Advances in Cryptology* (CRYPTO), pages 27–35, 1988.

11 S. J. Berkowitz. On some relationships between monotone and non-monotone circuit complexity. Technical report, University of Toronto, 1982.

12 Eric Blais, Johan Håstad, Rocco A. Servedio, and Li-Yang Tan. On DNF approximators for monotone boolean functions. In *International Colloquium on Automata, Languages, and Programming* (ICALP), pages 235–246, 2014.

13 Eric Blais, Dominik Scheder, and Li-Yang Tan. Ajtai-gurevich redux. Manuscript, 2013.

14 Nader H. Bshouty and Christino Tamon. On the Fourier spectrum of monotone functions. *J. ACM*, 43(4):747–770, 1996. doi:10.1145/234533.234564.

15 Andrei A. Bulatov. A dichotomy theorem for nonuniform csps. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 319–330. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.37.

16 Andrei A. Bulatov. Constraint satisfaction problems: complexity and algorithms. *ACM SIGLOG News*, 5(4):4–24, 2018. doi:10.1145/3292048.3292050.

17 Gerhard Buntrock, Carsten Damm, Ulrich Hertrampf, and Christoph Meinel. Structure and importance of logspace-mod class. *Math. Syst. Theory*, 25(3):223–237, 1992. doi:10.1007/BF01374526.

18 Elmar Böhler, Nadia Creignou, Steffen Reith, and Heribert Vollmer. Playing with boolean blocks, part i: Post's lattice with applications to complexity theory. *SIGACT News*, 2003.

19 Elmar Böhler, Nadia Creignou, Steffen Reith, and Heribert Vollmer. Playing with boolean blocks, part ii: Constraint satisfaction problems. *ACM SIGACT-Newsletter*, 35, 2004.

20 Bruno Pasqualotto Cavalar, Mrinal Kumar, and Benjamin Rossman. Monotone circuit lower bounds from robust sunflowers. In *LATIN 2020: Theoretical Informatics - 14th Latin American Symposium, São Paulo, Brazil, January 5-8, 2021, Proceedings*, volume 12118 of *Lecture Notes in Computer Science*, pages 311–322. Springer, 2020. doi:10.1007/978-3-030-61792-9_25.

21 Arkadev Chattopadhyay, Rajit Datta, and Partha Mukhopadhyay. Lower bounds for monotone arithmetic circuits via communication complexity. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 786–799. ACM, 2021. doi:10.1145/3406325.3451069.

22 Eshan Chattopadhyay and David Zuckerman. Explicit two-source extractors and resilient functions. In *Symposium on Theory of Computing* (STOC), pages 670–683, 2016.

23 Lijie Chen, Shuichi Hirahara, Igor C. Oliveira, Ján Pich, Ninad Rajgopal, and Rahul Santhanam. Beyond Natural Proofs: Hardness Magnification and Locality. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151, pages 70:1–70:48, 2020.

24 Xi Chen, Igor C. Oliveira, and Rocco A. Servedio. Addition is exponentially harder than counting for shallow monotone circuits. In *STOC'17 – Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1232–1245. ACM, New York, 2017. doi:10.1145/3055399.3055425.

25 Nadia Creignou, Sanjeev Khanna, and Madhu Sudan. *Complexity classifications of Boolean constraint satisfaction problems*, volume 7 of *SIAM monographs on discrete mathematics and applications*. SIAM, 2001.

26 Susanna F. de Rezende, Mika Göös, and Robert Robere. Guest column: Proofs, circuits, and communication. *SIGACT News*, 53(1):59–82, 2022. doi:10.1145/3532737.3532746.

27 Tomás Feder and Moshe Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1998. doi:10.1137/S0097539794266766.

28 Oded Goldreich, Shafi Goldwasser, Eric Lehman, and Dana Ron. Testing monotonicity. In *Symposium on Foundations of Computer Science,* (FOCS), pages 426–435, 1998.

29 Oded Goldreich and Rani Izsak. Monotone circuits: One-way functions versus pseudorandom generators. *Theory Comput.*, 8(1):231–238, 2012. doi:10.4086/toc.2012.v008a010.

30 Mika Göös, Rahul Jain, and Thomas Watson. Extension complexity of independent set polytopes. *SIAM J. Comput.*, 47(1):241–269, 2018. doi:10.1137/16M109884X.

31 Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in monotone complexity and TFNP. In *10th Innovations in Theoretical Computer Science*, volume 124 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. No. 38, 19. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2019.

32 Michelangelo Grigni and Michael Sipser. Monotone complexity. In *Proceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 57–75, USA, 1992. Cambridge University Press.

33 Siyao Guo, Tal Malkin, Igor C. Oliveira, and Alon Rosen. The power of negations in cryptography. In *Theory of Cryptography Conference* (TCC), pages 36–65, 2015.

34 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing, May 28-30, 1986, Berkeley, California, USA*, pages 6–20, 1986. doi:10.1145/12130.12132.

35 Johan Håstad, Ingo Wegener, Norbert Wurm, and Sang-Zin Yi. Optimal depth, very small size circuits for symmetric functions in AC^0. *Inf. Comput.*, 108(2):200–211, 1994. doi:10.1006/inco.1994.1008.

36 Shuichi Hirahara. NP-hardness of learning programs and partial MCSP. In *Symposium on Foundations of Computer Science* (FOCS), 2022.

37 Christian Ikenmeyer, Balagopal Komarath, Christoph Lenzen, Vladimir Lysikov, Andrey Mokhov, and Karteek Sreenivasaiah. On the complexity of hazard-free circuits. *J. ACM*, 66(4):25:1–25:20, 2019. doi:10.1145/3320123.

38 Peter Jeavons. On the algebraic structure of combinatorial problems. *Theor. Comput. Sci.*, 200(1-2):185–204, 1998. doi:10.1016/S0304-3975(97)00230-2.

39 Peter Jeavons, David Cohen, and Marc Gyssens. Closure properties of constraints. *J. ACM*, 44(4):527–548, July 1997. doi:10.1145/263867.263489.

40 Jan Johannsen. The complexity of satisfiability problems with two occurrences, 2003. Unpublished Manuscript.

41 Neil D. Jones, Y. Edmund Lien, and William T. Laaser. New problems complete for nondeterministic log space. *Math. Syst. Theory*, 10:1–17, 1976. doi:10.1007/BF01683259.

42 Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*. Springer, 2012.

43 Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Discret. Math.*, 5(4):545–557, 1992. doi:10.1137/0405044.

44 Mauricio Karchmer and Avi Wigderson. On span programs. In *Structure in Complexity Theory Conference* (CCC), pages 102–111, 1993. doi:10.1109/SCT.1993.336536.

45 Jan Krajíček. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *J. Symbolic Logic*, 62(2):457–486, 1997.

46 Denis Kuperberg. Positive first-order logic on words. In *Symposium on Logic in Computer Science* (LICS), pages 1–13, 2021.

47 Denis Kuperberg. Positive first-order logic on words and graphs. *CoRR*, abs/2201.11619, 2022. arXiv:2201.11619.

48 Benoît Larose and Pascal Tesson. Universal algebra and hardness results for constraint satisfaction problems. *Theor. Comput. Sci.*, 410(18):1629–1647, 2009. doi:10.1016/j.tcs.2008.12.048.

49 E. A. Okol'nishnikova. The effect of negations on the complexity of realization of monotone Boolean functions by formulas of bounded depth. *Metody Diskret. Analiz.*, pages 74–80, 1982.

50 Igor C. Oliveira. Unconditional lower bounds in complexity theory, 2015. (PhD Thesis, Columbia University).

51 Igor C. Oliveira, Rahul Santhanam, and Srikanth Srinivasan. Parity helps to compute majority. In *34th Computational Complexity Conference, CCC 2019, July 18-20, 2019, New Brunswick, NJ, USA*, pages 23:1–23:17, 2019. doi:10.4230/LIPIcs.CCC.2019.23.

52 Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *Symposium on Theory of Computing* (STOC), pages 1246–1255, 2017. doi:10.1145/3055399.3055478.

53 Emil L. Post. *The Two-Valued Iterative Systems of Mathematical Logic. (AM-5)*. Princeton University Press, 1941. URL: http://www.jstor.org/stable/j.ctt1bgzb1r.

54 Pavel Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *J. Symb. Log.*, 62(3):981–998, 1997. doi:10.2307/2275583.

55 W. V. Quine. Two theorems about truth functions. *Bol. Soc. Mat. Mexicana*, 10:64–70, 1953.

56 Ran Raz and Pierre McKenzie. Separation of the monotone NC hierarchy. *Combinatorica*, 19(3):403–435, 1999. doi:10.1007/s004930050062.

57 Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992. doi:10.1145/146637.146684.

58 A. A. Razborov. Lower bounds on the monotone complexity of some Boolean functions. *Dokl. Akad. Nauk SSSR*, 281(4):798–801, 1985.

59 Alexander A Razborov. Lower bounds on monotone complexity of the logical permanent. *Mathematical Notes of the Academy of Sciences of the USSR*, 37(6):485–493, 1985.

60 Omer Reingold. Undirected st-connectivity in log-space. In Harold N. Gabow and Ronald Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 376–385. ACM, 2005. doi:10.1145/1060590.1060647.

61 Susanna Rezende. Personal communication, 2023.

62 Robert Robere, Toniann Pitassi, Benjamin Rossman, and Stephen A. Cook. Exponential lower bounds for monotone span programs. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 406–415. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.51.

63 Benjamin Rossman. Homomorphism preservation theorems. *J. ACM*, 55(3):Art. 15, 53, 2008. doi:10.1145/1379759.1379763.

64 Benjamin Rossman. On the constant-depth complexity of k-clique. In *Symposium on Theory of Computing* (STOC), pages 721–730, 2008.

65 Benjamin Rossman. An entropy proof of the switching lemma and tight bounds on the decision-tree size of AC^0, 2017.

66 Benjamin Rossman. An improved homomorphism preservation theorem from lower bounds in circuit complexity. In *Innovations in Theoretical Computer Science Conference* (ITCS), pages 27:1–27:17, 2017.

67 Thomas J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing, May 1-3, 1978, San Diego, California, USA*, pages 216–226. ACM, 1978. doi:10.1145/800133.804350.

68 Alexei P. Stolboushkin. Finitely monotone properties. In *Symposium on Logic in Computer Science* (LICS), pages 324–330, 1995.

69 É. Tardos. The gap between monotone and nonmonotone circuit complexity is exponential. *Combinatorica*, 8(1):141–142, 1988. doi:10.1007/BF02122563.

70 Alasdair Urquhart. Hard examples for resolution. *J. ACM*, 34(1):209–219, 1987. doi:10.1145/7531.8928.

71 Leslie G. Valiant. Negation can be exponentially powerful. *Theor. Comput. Sci.*, 12:303–314, 1980.

72 Dmitriy Zhuk. A proof of CSP dichotomy conjecture. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 331–342. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.38.

A A Lower Bound for 3-XOR-SAT Using the Approximation Method

As discussed in Section 1.1.3, [31] obtained an exponential lower bound on the monotone circuit size of the function 3-XOR-SAT using techniques from communication complexity and lifting. Here we observe that a weaker but still super-polynomial lower bound can be proved using the approximation method.

First, we recall the function $\mathsf{OddFactor}_n : \{0,1\}^{\binom{n}{2}} \to \{0,1\}$ of Section 3.2, which accepts a given graph if the graph contains an *odd factor*, which is a spanning subgraph in which the degree of every vertex is odd. For convenience, in this section we consider a weaker version of OddFactor, which takes as an input a *bipartite* graph with n vertices on each part, and accepts if the graph contains an odd factor. Let $\mathsf{Bipartite\text{-}OddFactor}_n : \{0,1\}^{n^2} \to \{0,1\}$ be this function. We remark that the lower bounds of Babai, Gál and Wigderson [9] for OddFactor (Theorem 10) also hold for Bipartite-OddFactor. The proof of the monotone circuit lower bound in particular is essentially Razborov's lower bound for Matching via the approximation method [59].

▶ **Theorem 47** ([9]). *We have*

$$\mathsf{mSIZE}(\mathsf{Bipartite\text{-}OddFactor}_n) = n^{\Omega(\log n)} \quad \textit{and} \quad \mathsf{mDEPTH}(\mathsf{Bipartite\text{-}OddFactor}_n) = \Omega(n).$$

We can reduce Bipartite-OddFactor to 3-XOR-SAT by noting that computing $\mathsf{Bipartite\text{-}OddFactor}_n(M)$ on a given matrix $M \in \{0,1\}^{n^2}$ is computationally equivalent to deciding the satisfiability of the following $\mathbb{F}_2$ linear system over variables $\{x_{ij}\}$:

- For all $i \in [n]$: $\bigoplus_{k=1}^{n} x_{ik} = 1$;
- For all $j \in [n]$: $\bigoplus_{k=1}^{n} x_{kj} = 1$;
- For all $i, j \in [n]$ such that $M_{ij} = 0$: $x_{ij} = 0$.

We can then use a circuit for 3-XOR-SAT to solve this system by using a standard trick of introducing new variables to reduce the number of variables that appear in each equation, as done in the textbook reduction from SAT to 3-SAT. As the corresponding reductions turn out to be monotone, this implies monotone circuit and formula lower bounds for 3-XOR-SAT. We note that a somewhat similar argument (in the non-monotone setting) appears in Feder and Vardi [27, Theorem 30] regarding constraint satisfaction problems with the ability to count.

In order to formalise this argument, we will need the following definition and results.

▶ **Definition 48.** *Let f be a Boolean function. We define* $\mathsf{dual}(f) : x \mapsto \neg f(\neg x)$ *as the* dual *of f.*

▶ **Lemma 49.** *Let f be a monotone Boolean function. We have* $\mathsf{mSIZE}(f) = \mathsf{mSIZE}(\mathsf{dual}(f))$ *and* $\mathsf{mDEPTH}(f) = \mathsf{mDEPTH}(\mathsf{dual}(f))$.

Proof. The idea is to push negations to the bottom and eliminate double negations at the input layer. In other words, applying De Morgan rules, we can convert any $\{\wedge, \vee\}$-circuit computing f into a circuit computing $\mathsf{dual}(f)$ by swapping $\wedge$-gates for $\vee$-gates, and vice-versa. Moreover, this transformation preserves the depth of the circuit. ◀

We are ready to describe a monotone reduction from the function $\mathsf{Bipartite\text{-}OddFactor}_n$ to 3-XOR-SAT, which implies the desired lower bounds.

▶ **Theorem 50.** *There exists $\varepsilon > 0$ such that*

$$\mathsf{mSIZE}(\text{3-XOR-SAT}) = n^{\Omega(\log n)} \quad \textit{and} \quad \mathsf{mDEPTH}(\text{3-XOR-SAT}) = \Omega(n^{\varepsilon}).$$

Proof. Recall that the value of the function $\mathsf{Bipartite\text{-}OddFactor}_n(M)$ on a given matrix $M \in \{0,1\}^{n^2}$ is equal to 1 if the following system is satisfiable:

- For all $i \in [n]$: $\bigoplus_{k=1}^{n} x_{ik} = 1$;
- For all $j \in [n]$: $\bigoplus_{k=1}^{n} x_{kj} = 1$;
- For all $i, j \in [n]$ such that $M_{ij} = 0$: $x_{ij} = 0$.

We introduce some extra variables to reduce the number of variables in each equation in the following way. For every $i \in [n]$, introduce variables $z_{i1}, \ldots, z_{i(n-1)}$ and the equations

$$\begin{aligned} z_{i1} &= x_{i1} \oplus x_{i2}, \\ z_{i2} &= z_{i1} \oplus x_{i3}, \\ &\ldots \\ z_{i,(n-1)} &= z_{i,(n-2)} \oplus x_{i,n}, \\ z_{i,(n-1)} &= 1. \end{aligned}$$

Now note that these equations imply $z_{i,(n-2)} = \bigoplus_{k=1}^{n} x_{ik} = 1$. For each "column" equation $\bigoplus_{k=1}^{n} x_{kj} = 1$, we also add variables $w_{j1}, \ldots, w_{j(n-1)}$ as above. In total, we add at most $2n^2$ variables and $2n^2$ equations. Therefore, there is a system of linear equations on $O(n^2)$ variables, where each constraint contains at most 3 variables, which is satisfiable if and only if $\mathsf{Bipartite\text{-}OddFactor}_n(M) = 1$. Moreover, it is easy to see that the characteristic vector α of the set of equations of this system can be computed from M by an *anti-monotone* projection, as we activate a constraint that depends on the input when $M_{ij} = 0$.

Now let $f = \mathsf{dual}(\mathsf{3\text{-}XOR\text{-}SAT})$ and $\beta = \neg\alpha$. Since, by definition, 3-XOR-SAT accepts *unsatisfiable* systems, we get $\mathsf{Bipartite\text{-}OddFactor}_n(M) = \neg\mathsf{3\text{-}XOR\text{-}SAT}(\alpha) = f(\beta)$ and that β is a *monotone* projection of M. Therefore, by Lemma 49, we obtain

$$\mathsf{mSIZE}(\mathsf{Bipartite\text{-}OddFactor}_n) \leq \mathsf{mSIZE}(\mathsf{3\text{-}XOR\text{-}SAT})$$

and

$$\mathsf{mDEPTH}(\mathsf{Bipartite\text{-}OddFactor}_n) \leq \mathsf{mDEPTH}(\mathsf{3\text{-}XOR\text{-}SAT}). \qquad \blacktriangleleft$$

B Schaefer's Theorem in Monotone Complexity

B.1 Connectivity and generation functions

We recall the definitions of two prominent monotone Boolean functions that have efficient monotone circuits. Let $\mathsf{ST\text{-}CONN} : \{0,1\}^{n^2} \to \{0,1\}$ be the function that outputs 1 on a given directed graph G if there exists a path from 1 to n in G. Let $\mathsf{GEN} : \{0,1\}^{n^3} \to \{0,1\}$ be the Boolean function which receives a set T of triples $(i,j,k) \in [n^3]$, and outputs 1 if $n \in S$, where $S \subseteq [n]$ is the set generated with the following rules:

- *Axiom:* $1 \in S$,
- *Generation:* If $i, j \in S$ and $(i,j,k) \in T$, then $k \in S$.

The following upper bounds are well-known and easy to prove.

▶ **Theorem 51** ([42, Exercise 7.3], [56]). *We have* $\mathsf{ST\text{-}CONN} \in \mathsf{mNL}$ *and* $\mathsf{GEN} \in \mathsf{mSIZE}[\mathsf{poly}]$.

B.2 Proof of reduction lemmas

Here we present monotonised versions of the proofs of [19, Propositions 2.2 - 2.4], which give a simplified presentation of the results of [67].

▶ **Lemma 40** ([19, Proposition 2.3]). *If* $\mathsf{COQ}(S_1) \subseteq \mathsf{COQ}(S_2)$*, then there exists a constant* $C \in \mathbb{N}$ *such that* $\mathsf{CSP\text{-}SAT}^n_{S_1} \leq_m^{\mathsf{mOR}} \mathsf{CSP\text{-}SAT}^{Cn}_{S_2}$.

Proof. If $\mathsf{COQ}(S_1) \subseteq \mathsf{COQ}(S_2)$, then each relation of S_1 can be represented as a conjunctive query over S_2. Let F_1 be a S_1-formula. For each constraint C_1 of F_1, there exists a formula $\varphi(C_1)$ in $\mathsf{CNF}(S_2)$ such that C_1 is a projection of $\varphi(C_1)$ (i.e., C_1 is a conjunctive query of

$\varphi(C_1)$). However, note that C_1 is satisfiable if and only if $\varphi(C_1)$ is satisfiable. So we can replace the constraint C_1 by the *set* of constraints in $\varphi(C_1)$. Doing this for every constraint in F_1, we obtain an S_2-formula F_2 which is satisfiable iff F_1 is satisfiable.

Now note that, to decide if a given constraint application C of S_2 is in the reduction, it suffices to check if there exists a S_1-constraint C_1 in F_1 such that C is in $\varphi(C_1)$. Using non-uniformity, this can be easily done by an OR over the relevant input bits.

Finally, we observe that, since the arities of each relation in S_1 and S_2 are constant, we only add a constant number of variables for each constraint to represent S_1-formulas with conjunctive queries over S_2-formulas. ◀

▶ **Lemma 52.** *Let S be a set of Boolean relations. We have* $\mathsf{CSP\text{-}SAT}_{S\cup\{=\}} \leq_m^{\mathsf{mNL}} \mathsf{CSP\text{-}SAT}_S$.

Proof. Let F be a $(S \cup \{=\})$-formula on n variables given as an input. Remember that F is given as a Boolean vector α, where each bit of α represents the presence of a constraint application on n variables from $S \cup \{=\}$. We first build an undirected graph G with the variables $x_1, \ldots, x_n$ as vertices, and we put an edge between x_i and x_j if the constraint $x_i = x_j$ appears in F. Note that G can be constructed by a monotone projection from F.

Let $R \in S$ and let $C = R(x_1, \ldots, x_n)$ be a constraint application of R. If C appears in F, we add to the system every constraint of the form $C' = R(y_1, \ldots, y_n)$ such that, for every $i \in [n]$, there exists a path from x_i to y_i in the graph G. In this case, we say that C *generates* C'. Let F_2 be the formula that contains all non-equality constraints of F, and all the non-equality constraints generated by a constraint in F. It's not hard to see that F is satisfiable if and only if F_2 is satisfiable, and therefore the reduction is correct.

Moreover, the reduction can be done in monotone NL using the monotone NL algorithm for $\mathsf{ST\text{-}CONN}$ (Theorem 51). Indeed, there are at most n^k constraint applications of a given relation R of arity k. Therefore, to decide if a constraint $C' = R(y_1, \ldots, y_n)$ appears in F_2, it suffices to check if there exists a constraint application of R in F which generates C'. This can be checked with n^k calls to $\mathsf{ST\text{-}CONN}$. ◀

▶ **Lemma 31** (Polymorphisms characterise the complexity of CSPs [38, 19, Theorem 2.4])**.** *If* $\mathsf{Pol}(S_2) \subseteq \mathsf{Pol}(S_1)$*, then* $\mathsf{CSP\text{-}SAT}_{S_1}^n \leq_m^{\mathsf{mNL}} \mathsf{CSP\text{-}SAT}_{S_2}^{\mathsf{poly}(n)}$.

Proof. If $\mathsf{Pol}(S_2) \subseteq \mathsf{Pol}(S_1)$, then from Lemma 30 (Items iii, v, and vi) we obtain $\mathsf{COQ}(S_1) \subseteq \langle S_1 \rangle \subseteq \langle S_2 \rangle = \mathsf{COQ}(S_2 \cup \{=\})$. Therefore, by Lemmas 40 and 52 we can do the following chain of reductions in monotone NL:

$$\mathsf{CSP\text{-}SAT}_{S_1} \leq_m^{\mathsf{mOR}} \mathsf{CSP\text{-}SAT}_{S_2\cup\{=\}} \leq_m^{\mathsf{mNL}} \mathsf{CSP\text{-}SAT}_{S_2}.$$ ◀

B.3 Monotone circuit upper bounds

We restate and prove the theorem.

▶ **Theorem 32** (Monotone version of the upper bounds for CSP-SAT [67, 4])**.** *Let S be a finite set of relations. The following holds.*

1. *If* $\mathrm{E}_2 \subseteq \mathsf{Pol}(S)$ *or* $\mathrm{V}_2 \subseteq \mathsf{Pol}(S)$*, then* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mSIZE}[\mathsf{poly}]$.
2. *If* $\mathrm{D}_2 \subseteq \mathsf{Pol}(S)$*, or* $\mathrm{S}_{00} \subseteq \mathsf{Pol}(S)$*, or* $\mathrm{S}_{10} \subseteq \mathsf{Pol}(S)$*, then* $\mathsf{CSP\text{-}SAT}_S \in \mathsf{mNL}$.

Proof. We prove each case separately.

Proof of 1. We first observe that 3-Horn-SAT (see definition in Section 5.3, *Dichotomy for formulas*) can be solved by a reduction to $\mathsf{GEN} \in \mathsf{mSIZE}[\mathsf{poly}]$. Indeed, we interpret each constraint of the form $(\neg x_i \vee \neg x_j \vee x_k)$ (which is equivalent to $x_i \wedge x_j \implies x_k$) as a triple (i, j, k), and constraints of the form x_i as a triple $(0, 0, i)$. Let $S \subseteq \{0, 1, 2 \ldots, n\}$ be the

set generated by these triples, applying the generation rules of GEN, using $0 \in S$ as the axiom. It suffices to check that there exists some constraint of the form $\neg x_i \vee \neg x_j \vee \neg x_k$, such that $\{i, j, k\} \subseteq S$. This process can be done with polynomial-size monotone circuits, invoking GEN. Therefore, it follows from Theorem 51 that 3-Horn-SAT $\in$ mSIZE[poly].

Moreover, we recall that, if $\mathrm{E}_2 \subseteq \mathsf{Pol}(S)$, then $S \subseteq \mathsf{COQ}(\mathcal{H}^3)$ (in other words, every S-formula can be written as a set of 3-Horn equations) – see, e.g, [25, Lemma 4.8]. Therefore, from Items iv and v of Lemma 30 and Lemma 40, we conclude that $\mathsf{CSP\text{-}SAT}_S \leq_m^{\mathsf{mOR}}$ 3-Horn-SAT $\in$ mSIZE[poly].

Now recall that, if $\mathrm{V}_2 \subseteq \mathsf{Pol}(S)$, then $S \subseteq \mathsf{COQ}(\mathcal{A}^3)$, where $\mathcal{A}^3$ is the set of width-3 Anti-Horn relations (i.e., $\mathcal{A}^3 = \{(x_1 \vee x_2 \vee \neg x_3), (x_1 \vee x_2 \vee x_3), (\neg x)\}$; see [25, Lemma 4.8] for a proof of this observation). But note that an $\mathcal{A}^3$-formula φ is satisfiable if and only if the $\mathcal{H}^3$-formula $\varphi(\neg x_1, \ldots, \neg x_n)$ is satisfiable. Therefore by Lemma 40 and Items iv and v of Lemma 30, we have $\mathsf{CSP\text{-}SAT}_S \leq_m^{\mathsf{mOR}} \mathsf{CSP\text{-}SAT}_{\mathcal{A}^3} \leq_m^{\mathsf{mProj}}$ 3-Horn-SAT $\in$ mSIZE[poly].

Proof of 2. We first prove the case $\mathrm{D}_2 \subseteq \mathsf{Pol}(S)$. Let 2-SAT $= \mathsf{CSP\text{-}SAT}_\Gamma$, where $\Gamma = \{(x_1 \vee x_2), (x_1 \vee \neg x_2), (\neg x_1 \vee \neg x_2)\}$. It's easy to check that the standard reduction from 2-SAT to ST-CONN can be done in monotone NL (see [41, Theorem 4]). Therefore, it follows from Theorem 51 that 2-SAT $\in$ mNL. Now, recall that, if $\mathrm{D}_2 \subseteq \mathsf{Pol}(S)$, then $S \subseteq \mathsf{COQ}(\Gamma)$ (see, e.g., [25, Lemma 4.9]). Therefore, from Lemma 40 and Items iv and v of Lemma 30, we conclude $\mathsf{CSP\text{-}SAT}_S \in$ mNL.

We now suppose that $\mathrm{S}_{00} \subseteq \mathsf{Pol}(S)$. We check that the proof of [4, Lemma 3.4] gives a monotone circuit. If $\mathrm{S}_{00} \subseteq \mathsf{Pol}(S)$, then there exists $k \geq 2$ such that ${\mathrm{S}_{00}}^k \subseteq \mathsf{Pol}(S)$ (that's because there does not exist a finite set of relations S such that $\mathsf{Pol}(S) = \mathrm{S}_{00}$). Note that ${\mathrm{S}_{00}}^k = \mathsf{Pol}(\Gamma)$, where $\Gamma = \left\{\mathsf{OR}^k, x, \neg x, \rightarrow, =\right\}$. We show below how to decide if a Γ-formula is unsatisfiable in monotone NL. The result then follows from Lemma 31. Let F be a given Γ-formula with n variables. We first construct a directed graph G, with vertex set $\{x_1, \ldots, x_n\}$, and with arcs (x_i, x_j) if $x_i \rightarrow x_j$ is a constraint of F, and arcs (x_i, x_j) and (x_j, x_i) if $x_i = x_j$ is a constraint of F. This can be done with a monotone projection. Observe that a Γ-formula F is unsatisfiable if, and only if, there exists a constraint of the form $x_{i_1} \vee \cdots \vee x_{i_k}$ in F, such that there exists a path from some x_{i_j} to a constraint $\neg y$ in F. This can be checked in monotone NL by Theorem 51.

The case $\mathrm{S}_{10} \subseteq \mathsf{Pol}(S)$ is analogous. ◀

C Background on Post's Lattice and Clones

In this section, we include the definitions of the various clones that are used in the paper, as well as a figure of Post's lattice, which can be helpful when checking the proofs of Section 5.

Let $\rightarrow$: $(x, y) \mapsto (\neg x \vee y)$. Let also $\leftrightarrow$: $(x, y) \mapsto \neg(x \oplus y)$ and $\mathrm{id} : x \mapsto x$. Let $f : \{0,1\}^k \rightarrow \{0,1\}$ be a Boolean function. We say that f is *linear* if there exists $c \in \{0,1\}^k$ and $b \in \{0,1\}$ such that $f(x) = \langle c, x\rangle + b \pmod 2$. We say that f is self-dual if $f = \mathsf{dual}(f)$. Let $a \in \{0,1\}$. We say that f is *a-reproducing* if $f(a, \ldots, a) = a$. We say that a set $T \subseteq \{0,1\}^k$ is *a-separating* if there exists $i \in [k]$ such that $x_i = a$ for all $x \in T$. We say that f is *a-separating* if $f^{-1}(a)$ is a-separating. We say that f is *a-separating of degree k* if every $T \subseteq f^{-1}(a)$ such that $|T| = k$ is a-separating. The *basis* of a clone B is a set of Boolean functions F such that $B = [F]$.

Name	Definition	Base
BF	All Boolean functions	$\{\vee, \wedge, \neg\}$
R_0	$\{f \in \mathrm{BF} : f \text{ is 0-reproducing}\}$	$\{\wedge, \oplus\}$
R_1	$\{f \in \mathrm{BF} : f \text{ is 1-reproducing}\}$	$\{\vee, \leftrightarrow\}$
R_2	$\mathrm{R}_1 \cap \mathrm{R}_0$	$\{\vee, x \wedge (y \leftrightarrow z)\}$
M	$\{f \in \mathrm{BF} : f \text{ is monotonic}\}$	$\{\vee, \wedge, 0, 1\}$
M_1	$\mathrm{M} \cap \mathrm{R}_1$	$\{\vee, \wedge, 1\}$
M_0	$\mathrm{M} \cap \mathrm{R}_0$	$\{\vee, \wedge, 0\}$
M_2	$\mathrm{M} \cap \mathrm{R}_2$	$\{\vee, \wedge\}$
S_0^n	$\{f \in \mathrm{BF} : f \text{ is 0-separating of degree } n\}$	$\{\rightarrow, \mathsf{dual}(h_n)\}$
S_0	$\{f \in \mathrm{BF} : f \text{ is 0-separating}\}$	$\{\rightarrow\}$
S_1^n	$\{f \in \mathrm{BF} : f \text{ is 1-separating of degree } n\}$	$\{x \wedge \overline{y}, h_n\}$
S_1	$\{f \in \mathrm{BF} : f \text{ is 1-separating}\}$	$\{x \wedge \overline{y}\}$
S_{02}^n	$\mathrm{S}_0^n \cap \mathrm{R}_2$	$\{x \vee (y \wedge \overline{z}), \mathsf{dual}(h_n)\}$
S_{02}	$\mathrm{S}_0 \cap \mathrm{R}_2$	$\{x \vee (y \wedge \overline{z})\}$
S_{01}^n	$\mathrm{S}_0^n \cap \mathrm{M}$	$\{\mathsf{dual}(h_n), 1\}$
S_{01}	$\mathrm{S}_0 \cap \mathrm{M}$	$\{x \vee (y \wedge z), 1\}$
S_{00}^n	$\mathrm{S}_0^n \cap \mathrm{R}_2 \cap \mathrm{M}$	$\{x \vee (y \wedge z), \mathsf{dual}(h_n)\}$
S_{00}	$\mathrm{S}_0 \cap \mathrm{R}_2 \cap \mathrm{M}$	$\{x \vee (y \wedge z)\}$
S_{12}^n	$\mathrm{S}_1^n \cap \mathrm{R}_2$	$\{x \wedge (y \vee \overline{z}), h_n\}$
S_{12}	$\mathrm{S}_1 \cap \mathrm{R}_2$	$\{x \wedge (y \vee \overline{z})\}$
S_{11}^n	$\mathrm{S}_1^n \cap \mathrm{M}$	$\{h_n, 0\}$
S_{11}	$\mathrm{S}_1 \cap \mathrm{M}$	$\{x \wedge (y \vee z), 0\}$
S_{10}^n	$\mathrm{S}_1^n \cap \mathrm{R}_2 \cap \mathrm{M}$	$\{x \wedge (y \vee z), h_n\}$
S_{10}	$\mathrm{S}_1 \cap \mathrm{R}_2 \cap \mathrm{M}$	$\{x \wedge (y \vee z)\}$
D	$\{f \in \mathrm{BF} : f \text{ is self-dual}\}$	$\{(x \wedge \overline{y}) \vee (x \wedge \overline{z}) \vee (\overline{y} \wedge \overline{z})\}$
D_1	$\mathrm{D} \cap \mathrm{R}_2$	$\{(x \wedge y) \vee (x \wedge \overline{z}) \vee (y \wedge \overline{z})\}$
D_2	$\mathrm{D} \cap \mathrm{M}$	$\{(x \wedge y) \vee (y \wedge z) \vee (x \wedge z)\}$
L	$\{f \in \mathrm{BF} : f \text{ is linear}\}$	$\{\oplus, 1\}$
L_0	$\mathrm{L} \cap \mathrm{R}_0$	$\{\oplus\}$
L_1	$\mathrm{L} \cap \mathrm{R}_1$	$\{\leftrightarrow\}$
L_2	$\mathrm{L} \cap \mathrm{R}$	$\{x \oplus y \oplus z\}$
L_3	$\mathrm{L} \cap \mathrm{D}$	$\{x \oplus y \oplus z \oplus 1\}$
V	$\{f \in \mathrm{BF} : f \text{ is constant or an } n\text{-ary OR function}\}$	$\{\vee, 0, 1\}$
V_0	$[\{\vee\}] \cup [\{0\}]$	$\{\vee, 0\}$
V_1	$[\{\vee\}] \cup [\{1\}]$	$\{\vee, 1\}$
V_2	$[\{\vee\}]$	$\{\vee\}$
E	$\{f \in \mathrm{BF} : f \text{ is constant or an } n\text{-ary AND function}\}$	$\{\wedge, 0, 1\}$
E_0	$[\{\wedge\}] \cup [\{0\}]$	$\{\wedge, 0\}$
E_1	$[\{\wedge\}] \cup [\{1\}]$	$\{\wedge, 1\}$
E_2	$[\{\wedge\}]$	$\{\wedge\}$
N	$[\{\neg\}] \cup [\{0\}] \cup [\{1\}]$	$\{\neg, 1\}$
N_2	$[\{\neg\}]$	$\{\neg\}$
I	$[\{\mathrm{id}\}] \cup [\{0\}] \cup [\{1\}]$	$\{\mathrm{id}, 0, 1\}$
I_0	$[\{\mathrm{id}\}] \cup [\{0\}]$	$\{\mathrm{id}, 0\}$
I_1	$[\{\mathrm{id}\}] \cup [\{1\}]$	$\{\mathrm{id}, 1\}$
I_2	$[\{\mathrm{id}\}]$	$\{\mathrm{id}\}$

Figure 4 Table of all closed classes of Boolean functions, and their bases. Here, h_n denotes the function $h_n(x_1, \ldots, x_{n+1}) = \bigvee_{i=1}^{n+1} \bigwedge_{j=1, j\neq i}^{n+1} x_j$. See Definition 48 for the definition of $\mathsf{dual}(\cdot)$. The same table appears in [4, Table 1].

A Degree 4 Sum-Of-Squares Lower Bound for the Clique Number of the Paley Graph

Dmitriy Kunisky
Department of Computer Science, Yale University, New Haven, CT, USA

Xifan Yu
Department of Computer Science, Yale University, New Haven, CT, USA

Abstract

We prove that the degree 4 sum-of-squares (SOS) relaxation of the clique number of the Paley graph on a prime number p of vertices has value at least $\Omega(p^{1/3})$. This is in contrast to the widely believed conjecture that the actual clique number of the Paley graph is $O(\text{polylog}(p))$. Our result may be viewed as a derandomization of that of Deshpande and Montanari (2015), who showed the same lower bound (up to $\text{polylog}(p)$ terms) with high probability for the Erdős-Rényi random graph on p vertices, whose clique number is with high probability $O(\log(p))$. We also show that our lower bound is optimal for the Feige-Krauthgamer construction of pseudomoments, derandomizing an argument of Kelner. Finally, we present numerical experiments indicating that the value of the degree 4 SOS relaxation of the Paley graph may scale as $O(p^{1/2-\varepsilon})$ for some $\varepsilon > 0$, and give a matrix norm calculation indicating that the pseudocalibration construction for SOS lower bounds for random graphs will not immediately transfer to the Paley graph. Taken together, our results suggest that degree 4 SOS may break the "$\sqrt{p}$ barrier" for upper bounds on the clique number of Paley graphs, but prove that it can at best improve the exponent from 1/2 to 1/3.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization; Theory of computation → Semidefinite programming; Mathematics of computing → Combinatorial optimization

Keywords and phrases convex optimization, sum of squares, Paley graph, derandomization

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.30

Related Version *Full Version*: https://arxiv.org/abs/2211.02713

Funding This work was partially supported by ONR Award N00014-20-1-2335 and a Simons Investigator Award from the Simons Foundation to Daniel Spielman.

Acknowledgements We thank Afonso Bandeira, Chris Jones, and Daniel Spielman for helpful discussions, and the anonymous reviewers for their careful reading of the paper.

1 Introduction

1.1 Maximum and Planted Clique Problems in Random Graphs

For a graph G, we denote by $\omega(G)$ the number of vertices in the largest *clique* or complete subgraph in G. Computing $\omega(G)$ is a classical NP-hard problem in combinatorial optimization, which is moreover hard to approximate within any polynomial factor $n^{1-\varepsilon}$ for $\varepsilon > 0$ [31, 24]. Aside from this worst-case hardness, an average-case setting of computing $\omega(G)$ was proposed by Karp [32]. In this setting, the input graph is an Erdős-Rényi (ER) random graph G on n vertices, where each edge is present independently with probability $\frac{1}{2}$. We denote this by distribution by $G \sim \mathcal{G}\left(n, \frac{1}{2}\right)$. It is known that (see, e.g., [5, Section 11.1]), with high probability,

$$\omega(G) \in [(2 - o(1)) \log_2 n, (2 + o(1)) \log_2 n]\,. \tag{1}$$

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 30; pp. 30:1–30:25
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

In [32], Karp showed that a simple greedy algorithm with high probability finds a clique of size roughly $\log_2 n$, and asked whether a polynomial-time algorithm can with high probability find a clique of size $(1+\varepsilon)\log n$ for any constant $\varepsilon > 0$. The problem remains open, but, perhaps surprisingly, evidence has accumulated that such an algorithm does not exist [28, 17].

A natural related problem is that of algorithmically *bounding* the size of the largest clique in G, outputting a number that is always an upper bound on $\omega(G)$. For example, under $G \sim \mathcal{G}(n, \frac{1}{2})$, a simple algorithm based on the maximum degree can produce a $O(\sqrt{n \log n})$ bound [34]. Spectral algorithms operating on the eigenvalues of the adjacency matrix of G can improve this to $O(\sqrt{n})$ (for instance, using Haemers' generalization to irregular graphs of Hoffman's classical spectral bound on the clique number [22]).

The question of algorithmically bounding the clique number is also related to the problem of *hypothesis testing* between $G \sim \mathcal{G}(n, \frac{1}{2})$ and G drawn from another distribution where a typical G contains a *planted* clique of size much larger than $2\log_2 n$, since if we have an algorithm that always produces a valid bound on $\omega(G)$ and this bound is typically small for $G \sim \mathcal{G}(n, \frac{1}{2})$, then we can use its output to detect the planting of a sufficiently large clique. The above then shows that we may detect the presence of a clique of size $C\sqrt{n}$ for sufficiently large C; [2] moreover showed that an efficient spectral algorithm can even *recover* the vertex set of a planted clique of this size.[1]

A long line of work considered whether using convex relaxations of $\omega(G)$ that produce bounds that are in general stronger than spectral bounds can break this "$\sqrt{n}$ barrier" for $G \sim \mathcal{G}(n, \frac{1}{2})$, with a particular focus on semidefinite programming (SDP) relaxations. [30] showed that Lovász's ϑ function [37] also has value $\Omega(\sqrt{n})$; [14] later considered further aspects of using the ϑ function for detecting and recovering planted cliques. [15] showed the same $\Omega(\sqrt{n})$ lower bound for any constant level of the Lovász-Schrijver hierarchy of SDPs, of which the ϑ function is merely the first and weakest. The stronger sum-of-squares (SOS) hierarchy of relaxations proved harder to analyze. The pioneering but flawed analysis of [40] was fixed by [39], albeit at the cost of falling short of an $\Omega(\sqrt{n})$ lower bound. Many subsequent works, first on the degree 4 SOS relaxation [9, 45, 27] and culminating in the development of the *pseudocalibration* technique for larger degrees [4], ultimately established an $\Omega(n^{1/2-o(1)})$ lower bound for any constant degree of the SOS hierarchy.[2]

All of these results apply, as we have mentioned, to the average case of computing the clique number over $G \sim \mathcal{G}(n, \frac{1}{2})$. Some recent literature has revisited other average-case SOS lower bounds and identified *deterministic* instances over which the same quality of lower bound holds (see in particular the work of [11, 26], derandomizing the result of [19] on refuting 3-XORSAT instances).[3] In this paper, we initiate the study of the same question for the clique problem, by derandomizing the SOS lower bound of [9] for the degree 4 SOS relaxation of $\omega(G)$ with $G \sim \mathcal{G}(n, \frac{1}{2})$. The deterministic graphs that achieve this derandomization are the *Paley graphs*, whose clique number is a question of independent interest in number theory. We first review some background on the Paley graphs, and then describe our results.

---

[1] Observe that, while a brute force search can both detect and recover a planted clique of any size $(2+\varepsilon)\log_2 n$, this brute force search does not run in polynomial time.

[2] The SOS hierarchy consists of a sequence of SDPs producing smaller and smaller upper bounds on $\omega(G)$, indexed by an even number called the *degree*. See Section 2.2.1 for a precise definition.

[3] Here we are interested in quantitative lower bounds showing large integrality gaps, rather than arbitrarily small integrality gaps – deterministic explicit examples giving the latter for high degrees of SOS have been shown before for several problems in works such as [19, 35].

1.2 Paley Graphs, Pseudorandomness, and Derandomization

The Paley graphs are an infinite family of graphs that exhibit certain *pseudorandom* properties, behaving in some regards similarly to a typical $G \sim \mathcal{G}(n, \frac{1}{2})$. They are defined on vertex sets identified with finite fields $\mathbb{F}_q$ of order $q \equiv 1 \pmod 4$, where edges connect pairs of elements of $\mathbb{F}_q$ whose differences are quadratic residues. We denote the Paley graph on $\mathbb{F}_q$ by G_q; the reader may see Section 2.2.2 for a more precise definition.

Many quantities that may be computed from Paley graphs are the same as those of typical graphs drawn from $\mathcal{G}(q, \frac{1}{2})$. In the simplest instance, Paley graphs are regular of degree $\frac{q-1}{2}$, roughly the average degree of the corresponding random graph. [7] showed that the same holds for the number of occurrences of any subgraph of constant size, for the first eigenvalue being asymptotically $\frac{q}{2}$, and the second eigenvalue being $o(q^{\frac{1}{2}+\varepsilon})$ for any $\varepsilon > 0$.

How far can we take this analogy? It is natural to ask for subgraph counts of graphs of size growing slowly with q, and the clique number is just such a question: under $G \sim \mathcal{G}(q, \frac{1}{2})$ we have $\mathbb{E}[\omega(G)] \sim 2 \log_2 q$, and we might expect the same for $\omega(G_q)$.

However, the clique number of Paley graphs is not well understood. Let us review what is currently known. Hoffman's spectral bound [25, 22] implies the upper bound

$$\omega(G_q) \leq \sqrt{q}. \tag{2}$$

In fact, this is easy to derive by elementary combinatorial means (see, e.g., [47]) and for this reason is sometimes called the *trivial* upper bound on $\omega(G_q)$. This is tight for $q = p^{2k}$ an even power of a prime, as $\mathbb{F}_{\sqrt{q}}$ may be realized as a subfield of $\mathbb{F}_q$ all of whose elements are quadratic residues in this case [6].

However, for odd prime powers, and even the simplest case $q = p$ a prime, the clique number is believed to be much lower. The upper bound on the diagonal Ramsey number established by [12] implies that

$$\omega(G_p) \geq \left(\frac{1}{2} + o(1)\right) \log_2 p. \tag{3}$$

By a number-theoretic analysis of the least quadratic non-residue modulo p, [18] improved this, showing that for infinitely many primes p,

$$\omega(G_p) \geq \log p \log\log\log p. \tag{4}$$

Moreover, conditional on the Generalized Riemann Hypothesis, the $\log\log\log p$ term may be improved to $\log\log p$ [41, Theorem 13.5].[4]

On the other hand, the best known upper bound [23, 10] improves only by a constant factor on the spectral bound (2),

$$\omega(G_p) \leq \frac{\sqrt{2p-1}+1}{2} \sim \frac{\sqrt{p}}{\sqrt{2}}. \tag{5}$$

In contrast to this state-of-the-art bound, $\omega(G_p)$ is widely believed to actually scale at most polylogarithmically with p based on computations of $\omega(G_p)$ for small p. We express this in the following conjecture; see [46, 3, 47, 33] as well as our Figure 2.

▶ **Conjecture 1.** *For some $C, K > 0$ and all $p \equiv 1 \pmod 4$ prime, $\omega(G_p) \leq C(\log p)^K$.*

Numerical evidence suggests that we might in fact expect to be able to take $K = 2$, as discussed by [3, 33] and illustrated in our Figure 2.

[4] It is still possible to reconcile these results with the proposal that G_p behaves like a random graph, so long as we adopt a more sophisticated random model than $\mathcal{G}(p, \frac{1}{2})$ [42].

Moreover, these graphs are believed to be good constructions for lower bounds on the diagonal Ramsey numbers $R(k,k)$. For example, the Paley graph of order 17 is the unique largest graph that contains neither a clique of size 4 nor an independent set of size 4, which shows that $R(4,4) = 18$ [13]. The current best known bound $R(6,6) \geq 102$ is established by the Paley graph of order 101, which contains neither a clique of size 6 nor an independent set of size 6 [44].

Because of this application among others, it is a long-standing open problem in additive combinatorics and number theory to improve the upper bound for clique numbers of Paley graphs of prime orders, and in particular to break the "$\sqrt{p}$ barrier" and prove an upper bound scaling as $p^{1/2-\varepsilon}$ for some $\varepsilon > 0$.[5] Some recent work has begun to explore whether convex relaxations of the clique number can lead to such improvements. For instance, [21, 33] explored using a hierarchy of SDPs producing bounds between that of the Lovász-Schrijver hierarchy and the SOS hierarchy for this purpose, and [38] empirically found that a modification of the Lovász ϑ function SDP can recover and sometimes slightly improve on the best-known upper bound (5).

1.3 Our Contributions

Our main result contributes to both of the lines of work outlined above. On the one hand, it shows (conditional on Conjecture 1) that the Paley graph gives a derandomization of the SOS lower bound of [9] for ER random graphs. On the other hand, it shows that a powerful convex optimization approach to upper-bounding the clique number cannot be too effective when applied to G_p.

▶ **Theorem 2.** *There is a constant $c > 0$ such that the value of the degree 4 SOS relaxation of the clique number $\mathrm{SOS}_4(G)$, as defined in Section 2.2.1, evaluated with G_p the Paley graph on p vertices for p any prime number with $p \equiv 1 \pmod 4$, as defined in Section 2.2.2, satisfies*

$$\mathrm{SOS}_4(G_p) \geq cp^{1/3}. \tag{6}$$

The main ingredients in our proof are new norm bounds for certain *graph matrices* (as appear in the analysis of SOS relaxations for random graphs; see, e.g., [1]) formed from Paley graphs and certain character sum estimates for the Legendre symbol.

To elaborate on this result, we provide three further pieces of more detailed analysis. Note that Theorem 2 does not exclude the possibility that $\mathrm{SOS}_4(G_p) = o(\sqrt{p})$. In Section 4.1, however, we show that at least the lower bound construction we use to prove Theorem 2, involving the simple class of *Feige-Krauthgamer pseudomoments* (see Definition 6), cannot improve on the $p^{1/3}$ scaling of our lower bound.

On the other hand, in Section 4.2, we present some numerical evidence that $\mathrm{SOS}_4(G_p) \sim p^{\eta}$ for a constant $\eta \in (0, \frac{1}{2})$, with value $\eta \approx 0.4$. As we discuss in Section 4.2, these results are similar to earlier numerical studies of [21], who consider a weaker class of SDPs than the SOS hierarchy, and results of [33], who consider the same weaker SDPs and extract a prediction of the power scaling of their values with p from numerical results. We thus have reason to believe that our lower bound cannot be improved all the way to a scaling of $p^{1/2}$. Unfortunately, we have not found a way to convert these numerical results into a proof of an improved bound on the clique number, but we leave this as a tantalizing open problem for future work.

[5] For instance, this is mentioned as "probably a very hard problem" in the problem list [8].

Finally, to accompany these empirical results, we provide some modest theoretical evidence that the SOS hierarchy may break the $\sqrt{p}$ barrier for upper bounds on $\omega(G_p)$. The tight analysis showing that $\mathbb{E}[\mathrm{SOS}_{2d}(G)] = \Omega(n^{1/2-o(1)})$ for $G \sim \mathcal{G}(n, \frac{1}{2})$ and any constant d uses a construction satisfying a property called *pseudocalibration* [4], whose analysis hinges on norm bounds for the aforementioned graph matrices built from the adjacency matrix of G [1]. In Section 4.3, we show that some of these norm bounds *fail* for the Paley graph. Thus, the analysis of the pseudocalibration construction for random graphs cannot be directly adapted to the case of Paley graphs.[6]

2 Preliminaries and Proof Overview

2.1 Notations

Throughout the paper, p will denote a prime number, and q a prime power $q = p^k$. The finite field of order q (unique up to isomorphism) is denoted by $\mathbb{F}_q$, and its group of units by $\mathbb{F}_q^\times$. A nonzero element y of $\mathbb{F}_q$ is called a *quadratic residue* of $\mathbb{F}_q$ if $y = x^2$ for some $x \in \mathbb{F}_q$, and a *quadratic nonresidue* otherwise. We write $(\mathbb{F}_q^\times)^2$ for the set of quadratic residues. We will also freely identify $\mathbb{F}_p$ with $\mathbb{Z}/p\mathbb{Z}$, with representatives $\{0, 1, \ldots, p-1\}$.

We write $[n] := \{1, 2, \ldots, n\}$. For a finite set X, we write 2^X for the power set, and $\binom{X}{k}$ and $\binom{X}{\leq k}$ to denote the sets of subsets of X with exactly k elements and at most k elements respectively. We also use $X_{(k)}$ to denote the set of tuples of elements of X of length k with all entries distinct.

When the discussion involves variables $\{x_i\}_{i \in \mathcal{I}}$ indexed by $\mathcal{I}$, for a subset $S \subset \mathcal{I}$, we will use x^S to denote the monomial $\prod_{i \in S} x_i$.

We use $\mathbf{1} \in \mathbb{R}^n$ to denote the all-ones vector. We use $I \in \mathbb{R}^{n \times n}$ to denote the identity matrix, $J \in \mathbb{R}^{n \times n}$ to denote the all-ones matrix, and $\mathbf{0} \in \mathbb{R}^{n \times n}$ to denote the all-zeros matrix. The dimensions of these objects will be clear from context. For a real symmetric or Hermitian matrix A, we use $\mathrm{spec}(A)$ to denote its spectrum, which we write in double braces $\{\{\cdots\}\}$ to indicate that the spectrum is a multiset. For matrices $A, B \in \mathbb{C}^{n \times n}$ and $C \in \mathbb{C}^{m \times m}$, we use $A \circ B \in \mathbb{C}^{n \times n}$ to denote the Hadamard product (entrywise product) of A and B, and $A \otimes C \in \mathbb{C}^{nm \times nm}$ to denote the Kronecker product (tensor product) of A and C.

For a graph $G = (V, E)$, we use $V(G)$ to denote its vertex set and $E(G)$ to denote its edge set. We use $\overline{G}$ to denote the complement of G. For vertices $u, v \in V(G)$, we use $u \sim_G v$ to indicate that u and v are adjacent in G and $u \not\sim_G v$ to indicate that they are not adjacent. We will use A_G to denote the $\{0, 1\}$ adjacency matrix of G, and S_G to denote the Seidel or $\{\pm 1\}$ adjacency matrix. We drop the subscript G when the graph is clear from context. Conventionally, the Seidel adjacency matrix is -1 on pairs of adjacent vertices, $+1$ on pairs of nonadjacent vertices, and 0 on the diagonal. In this paper, we abuse this term to mean the matrix that is 1 on pairs of adjacent vertices, -1 on pairs of nonadjacent vertices, and 0 on the diagonal, as this is more conveniently written in terms of the Legendre symbol in the context of Paley graphs (see Section 3.4). It is easy to see that the A_G and S_G are related by $S_G = 2A_G - J + I$. We write $\mathcal{K}(G)$ for the set of subsets of $V(G)$ that form cliques in G.

We will use the standard asymptotic notations $O(\cdot), \Omega(\cdot), \Theta(\cdot)$, and $o(\cdot)$. We will use $\widetilde{O}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ to additionally suppress polylogarithmic factors.

[6] We note that the initial premise of pseudocalibration, which involves comparing a pair of "null" and "alternative" random graph distributions, is not sensible to apply to the deterministic Paley graph. But, ultimately, the pseudocalibration argument yields a function mapping a graph to a matrix that one hopes will be feasible for a high-degree SOS program, and one may simply substitute the Paley graph into this function and consider the result.

2.2 Problem Setup

Let us now specify in full detail the SOS relaxations SOS_{2d} of the clique number, and the Paley graphs G_p.

2.2.1 Sum-Of-Squares Relaxations of the Clique Number

Let G be a graph of order n. The clique number $\omega(G)$ of G is equal to the value of the following polynomial optimization program:

$$\omega(G) = \left\{ \begin{array}{ll} \text{maximize} & \sum_{i \in V(G)} x_i \\ \text{subject to} & x_i^2 = x_i \text{ for all } i \in V(G), \\ & x_i x_j = 0 \text{ for all } i,j \in V(G) \text{ with } i \neq j \text{ and } i \not\sim_G j \end{array} \right\}. \tag{7}$$

It is easy to see that the feasible solutions of the program above are in one-to-one correspondence with the indicator vectors of the cliques in G. Before we introduce the SOS relaxations of the clique number, let us first define the *pseudoexpectation* operators over which the SOS relaxations optimize.

▶ **Definition 3** (Pseudoexpectation). *We say* $\widetilde{\mathbb{E}} : \mathbb{R}[x_1, \ldots, x_n]_{\leq 2d} \to \mathbb{R}$ *is a* degree $2d$ pseudoexpectation *with respect to polynomial constraints* $\{f_i(x) = 0\}_{i=1}^a$, $\{g_j(x) \geq 0\}_{j=1}^b$ *if the following properties hold:*

- $\widetilde{\mathbb{E}}$ *is linear.*
- $\widetilde{\mathbb{E}}[1] = 1$.
- $\widetilde{\mathbb{E}}[f_i(x)p(x)] = 0$, *for all* $p(x) \in \mathbb{R}[x_1, \ldots, x_n]$ *such that* $\deg(f_i p) \leq 2d$, *for all* $1 \leq i \leq a$.
- $\widetilde{\mathbb{E}}[p(x)^2] \geq 0$, *for all* $p(x) \in \mathbb{R}[x_1, \ldots, x_n]_{\leq d}$.
- $\widetilde{\mathbb{E}}[g_j(x)p(x)^2] \geq 0$, *for all* $p(x) \in \mathbb{R}[x_1, \ldots, x_n]$ *such that* $\deg(g_j p^2) \leq 2d$, *for all* $1 \leq j \leq b$.

In the case of the maximum clique program (7), the polynomial constraints are generated by the Boolean constraints $x_i^2 - x_i = 0$ for $i \in V(G)$ and the clique constraints $x_i x_j = 0$ for $i, j \in V(G)$ with $i \neq j$ and $i \not\sim_G j$,. For convenience, let us identify the vertex set $V(G)$ with $[n]$ where $n = |V(G)|$. Then, the degree $2d$ SOS relaxation of the polynomial optimization program (7) written in terms of pseudoexpectations is

$$\mathrm{SOS}_{2d}(G) = \left\{ \begin{array}{ll} \text{maximize} & \sum_{i=1}^n \widetilde{\mathbb{E}}[x_i] \\ \text{subject to} & \widetilde{\mathbb{E}} : \mathbb{R}[x_1, \ldots, x_n]_{\leq 2d} \to \mathbb{R} \text{ linear}, \\ & \widetilde{\mathbb{E}}[1] = 1, \\ & \widetilde{\mathbb{E}}[(x_i^2 - x_i)p(x)] = 0 \text{ for all } i \in [n], \deg(p) \leq 2d-2, \\ & \widetilde{\mathbb{E}}[x_i x_j p(x)] = 0 \text{ for all } i \not\sim_G j, \deg(p) \leq 2d-2, \\ & \widetilde{\mathbb{E}}[p(x)^2] \geq 0 \text{ for all } \deg(p) \leq d. \end{array} \right\}. \tag{8}$$

To see that this is indeed a relaxation of the clique program (7), observe that for any probability measure $\mu : 2^{[n]} \to \mathbb{R}^{\geq 0}$ supported on the cliques of the graph G, the corresponding expectation operator $\mathbb{E}_\mu$ is a pseudoexpectation of any degree.

For every monomial x^S for $S \in \binom{[n]}{\leq 2d}$, $\widetilde{\mathbb{E}}[x^S]$ is called the *pseudomoment* of S of the corresponding pseudoexpectation $\widetilde{\mathbb{E}}$. By linearity, every pseudoexpectation of degree $2d$ is uniquely determined by its pseudomoments of degree at most $2d$, i.e., by the set $\{\widetilde{\mathbb{E}}[x^S] : S \subseteq [n], |S| \leq 2d\}$. We may therefore encode the pseudoexpectation in the *pseudomoment matrix* $M \in \mathbb{R}_{\mathrm{sym}}^{\binom{[n]}{\leq d} \times \binom{[n]}{\leq d}}$ with entries

$$M_{S,T} = \widetilde{\mathbb{E}}[x^S x^T]. \tag{9}$$

This is especially convenient since the positivity of $\widetilde{\mathbb{E}}$ on squared polynomials is equivalent to positive semidefiniteness of M. We can then rewrite the above program (8) in the form of an SDP:

$$\mathrm{SOS}_{2d}(G) = \left\{ \begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} M_{\varnothing,\{i\}} \\ \text{subject to} & M \in \mathbb{R}^{\binom{[n]}{\leq d} \times \binom{[n]}{\leq d}} \\ & M_{\varnothing,\varnothing} = 1, \\ & M_{S,T} \text{ depends only on } S \cup T, \\ & M_{S,T} = 0 \text{ whenever } S \cup T \notin \mathcal{K}(G), \\ & M \succeq 0. \end{array} \right\}. \tag{10}$$

We will not verify in detail the equivalence of (10) and (8); the reader may consult [36] for an overview of this pseudomoment matrix framework, or the papers [9, 45, 27, 4] on SOS relaxations of $\omega(G)$ for further details.

▶ Remark 4 (Pseudomoment matrix compression). We note that the row and column of M indexed by any $S \notin \mathcal{K}(G)$ is forced by the constraints to be identically zero. These entries do not affect the positivity of M and do not play a role in the objective function, so we may just as well take M to be indexed by *cliques* of size at most d rather than arbitrary subsets of vertices.

In the special case $2d = 2$, the SDP in (10) takes the form

$$\mathrm{SOS}_{2}(G) = \left\{ \begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} y_i \\ \text{subject to} & y \in \mathbb{R}^n, Y \in \mathbb{R}^{n \times n}_{\mathrm{sym}}, \\ & Y_{i,i} = y_i \text{ for all } i \in [n], \\ & Y_{i,j} = 0 \text{ for all } i, j \in [n] \text{ with } i \neq j \text{ and } i \not\sim_G j, \\ & M = \begin{bmatrix} 1 & y^\top \\ y & Y \end{bmatrix} \succeq 0. \end{array} \right\}. \tag{11}$$

One can show (see [20, 16]) that the program above is equivalent to the *Lovász ϑ function* of the complement graph $\overline{G}$, a well-known upper bound on $\omega(G)$ due to [37]:

$$\mathrm{SOS}_2(G) = \vartheta(\overline{G}). \tag{12}$$

This SDP enjoys many special properties, some of which we will mention below; the reader may consult the above references for further information.

On the other hand, once the degree increases to $2d = 4$, the resulting SDP is not as well understood. This SDP, which we study in the remainder of the paper, takes the form

$$\mathrm{SOS}_{4}(G) = \left\{ \begin{array}{ll} \text{maximize} & \sum_{i=1}^{n} M^{0,1}_{\varnothing,i} \\ \text{subject to} & M^{r,c} \in \mathbb{R}^{\binom{[n]}{r} \times \binom{[n]}{c}} \text{ for } r, c \in \{0,1,2\}, \\ & M^{r,c}_{S,T} \text{ depends only on } S \cup T, \\ & M^{r,c}_{S,T} = 0 \text{ whenever } S \cup T \notin \mathcal{K}(G), \\ & M = \begin{bmatrix} 1 & M^{0,1} & M^{0,2} \\ M^{1,0} & M^{1,1} & M^{1,2} \\ M^{2,0} & M^{2,1} & M^{2,2} \end{bmatrix} \succeq 0 \end{array} \right\}. \tag{13}$$

2.2.2 Paley Graphs

We now give the definition and some useful basic properties of the Paley graphs.

▶ **Definition 5** (Paley graph). *Let $q = p^k$ be a prime power such that $q \equiv 1 \pmod 4$. The* Paley graph G_q *of order q then has vertex set $V(G_q) := \mathbb{F}_q$ and edge set*

$$E(G_q) := \left\{ \{a, b\} \in \binom{\mathbb{F}_q}{2} : a - b \in (\mathbb{F}_q^\times)^2 \right\}. \tag{14}$$

The condition $q \equiv 1 \pmod 4$ ensures that -1 is a square in $\mathbb{F}_q$. As a result, $a - b \in (\mathbb{F}_q^\times)^2$ if and only if $b - a \in (\mathbb{F}_q^\times)^2$, so the edge set is well-defined.

We will study the SOS relaxations of the clique number of Paley graphs, $\mathrm{SOS}_{2d}(G_q)$. Recall that the degree 2 SOS relaxation of the clique number of the Paley graph G_q is equal to the Lovász theta function of its complement, $\mathrm{SOS}_2(G_q) = \vartheta(\overline{G_q})$. Since G_q is self-complementary (under the automorphism $x \mapsto gx$ for g a multiplicative generator of $\mathbb{F}_q^\times$), $\vartheta(\overline{G_q}) = \vartheta(G_q)$. Since G_q is vertex-transitive, by Lovász's result in [37],

$$\vartheta(\overline{G_q})\vartheta(G_q) = |V(G_q)| = q, \tag{15}$$

whereby combining our observations shows that

$$\mathrm{SOS}_2(G_q) = \sqrt{q}. \tag{16}$$

This is the same as the upper bound of the clique number given by Hoffman's spectral bound. Thus, degree 2 SOS does not improve on the spectral bound, and degree 4 SOS, which we begin to analyze with Theorem 2, is the first more interesting degree.

2.3 Proof Overview

To prove Theorem 2, we will construct a feasible pseudomoment matrix M for the program (13) that has objective value $\Omega(p^{1/3})$. We will consider the following type of pseudomoments, which we call *Feige-Krauthgamer (FK) pseudomoments*, first studied by Feige and Krauthgamer [15] to prove lower bounds on Lovász-Schrijver relaxations for the maximum independent set of random graphs (sometimes these are called *MPW pseudomoments* after their use by the later paper [39]).

▶ **Definition 6** (Feige-Krauthgamer pseudomoments). *Consider the degree $2d$ SOS relaxation of the clique number of a graph G. We say the pseudomoments of a degree $2d$ pseudoexpectation $\widetilde{\mathbb{E}}$ are* Feige-Krauthgamer (FK) pseudomoments *if there exists a sequence $1 = \alpha_0, \alpha_1, \alpha_2, \ldots, \alpha_{2d} \in \mathbb{R}$ such that*

$$\widetilde{\mathbb{E}}[x^S] = \begin{cases} \alpha_{|S|} & \text{if } S \in \mathcal{K}(G) \text{ (i.e., if } S \text{ is a clique in } G) \\ 0 & \text{otherwise.} \end{cases} \tag{17}$$

We note that FK pseudomoments automatically satisfy all conditions on a pseudoexpectation other than positivity.

The line of work beginning with [40] sought to use FK pseudomoments to prove lower bounds on SOS relaxations of $\omega(G)$ for random graphs G.[7] While eventually in [27, 4] it was found that FK pseudomoments could *not* prove optimal $\Omega(\sqrt{n})$ lower bounds, earlier works still proved polynomial $\Omega(n^\eta)$ lower bounds with $\eta < \frac{1}{2}$ using FK pseudomoments,

[7] Some works, wanting to study an SOS relaxation that included the "exact" constraint $\sum_{i=1}^n x_i = k$ for some k, adjusted the FK pseudomoments to satisfy the consequences of this constraint (see, e.g., [27, 43]). We do not take this route here.

which are simpler to define and to work with than the alternatives developed later. In particular, our analysis will closely follow that of [9], who used FK pseudomoments to prove that $\mathrm{SOS}_4(G) = \widetilde{\Omega}(n^{\frac{1}{3}})$ with high probability for $G \sim \mathcal{G}(n, \frac{1}{2})$. [27] later showed that, up to polylogarithmic factors, this is optimal over any choice of FK pseudomoments for the degree 4 relaxation.

▶ Remark 7 (Partial symmetry). By vertex transitivity and edge transitivity of Paley graphs, there always exists an optimal degree 4 pseudoexpectation giving all $\widetilde{\mathbb{E}}[x_i]$ the same value and all $\widetilde{\mathbb{E}}[x_i x_j]$ with $i \sim j$ in G_p the same value, regardless of whether $\widetilde{\mathbb{E}}$ is given by FK pseudomoments or not. This strong symmetry of course fails to hold for ER random graphs.

Recall that in the degree 4 SOS program (13), we write the pseudomoment matrix M in the block form

$$M = \begin{bmatrix} 1 & M^{0,1} & M^{0,2} \\ M_{1,0} & M^{1,1} & M^{1,2} \\ M_{2,0} & M^{2,1} & M^{2,2} \end{bmatrix}. \tag{18}$$

We will follow the strategy of [9] to successively check the Schur complement conditions for positive semidefiniteness of M. Namely, we will rely on the following fact.

▶ **Proposition 8.** *Let*

$$M = \begin{bmatrix} A & B^\top \\ B & C \end{bmatrix} \in \mathbb{R}^{(a+b)\times(a+b)} \tag{19}$$

be a real symmetric matrix written in block form, with $A \in \mathbb{R}^{a\times a}$ and $C \in \mathbb{R}^{b\times b}$. If $A \succ 0$ and $C - BA^{-1}B^T \succeq 0$, then $M \succeq 0$. We call the matrix $C - BA^{-1}B^\top$ the Schur complement *of the block A in M.*

3 Proof of Theorem 2

We restate Theorem 2 in more detailed terms of the FK pseudomoments that we will construct.

▶ **Theorem 9.** *There exists a constant $c > 0$ so that, setting $\alpha_1 := cp^{-2/3}, \alpha_2 := 4\alpha_1^2, \alpha_3 := 8\alpha_1^3$, and $\alpha_4 := 512\alpha_1^4$, the FK pseudomoments defined by these parameters give a feasible solution to the degree 4 SOS relaxation (13) of the clique number of the Paley graphs G_p for all sufficiently large p.*

Theorem 2 follows, since the above gives, for all sufficiently large p,

$$\mathrm{SOS}_4(G_p) \geq p \cdot cp^{-2/3} = cp^{1/3}. \tag{20}$$

To remind the reader of the notations we set in the previous section, the pseudomoment matrix in the degree 4 SOS relaxation (13) is denoted

$$M = \begin{bmatrix} 1 & M^{0,1} & M^{0,2} \\ M^{1,0} & M^{1,1} & M^{1,2} \\ M^{2,0} & M^{2,1} & M^{2,2} \end{bmatrix}, \tag{21}$$

and we take this to be given by the FK pseudomoments proposed in Theorem 9. Recall that $M^{r,c} \in \mathbb{R}^{\binom{\mathbb{F}_p}{r}\times\binom{\mathbb{F}_p}{c}}$ for all $r, c \in \{0, 1, 2\}$. We will use

$$N = \begin{bmatrix} N^{1,1} & N^{1,2} \\ N^{2,1} & N^{2,2} \end{bmatrix} \tag{22}$$

to denote the Schur complement of the top left 1×1 block in M.

3.1 Filling Zero Rows and Columns

As mentioned before, we will fill in the zero rows and columns of N in order to make use of graph matrices. In this section, we define the matrix

$$H = \begin{bmatrix} H^{1,1} & H^{1,2} \\ H^{2,1} & H^{2,2} \end{bmatrix} \tag{23}$$

that will achieve this filling.

▶ **Definition 10.** *We write $\mathbb{1}_k : \binom{\mathbb{F}_p}{k} \to \{0,1\}$ for the function with $\mathbb{1}_k(S) = 1$ if S is a clique in G_p and $\mathbb{1}_k(S) = 0$ otherwise.*

We now expand the $N^{\bullet,\bullet}$ matrices in terms of this indicator function.

▶ **Proposition 11.** *Under the FK pseudomoments proposed in Theorem 9, the matrix N can be written as*

$$N = \begin{bmatrix} N^{1,1} & N^{1,2} \\ N^{2,1} & N^{2,2} \end{bmatrix}, \tag{24}$$

where $N^{1,1} \in \mathbb{R}^{\mathbb{F}_p \times \mathbb{F}_p}, N^{1,2} \in \mathbb{R}^{\mathbb{F}_p \times \binom{\mathbb{F}_p}{2}}, N^{2,1} = {N^{1,2}}^\top \in \mathbb{R}^{\binom{\mathbb{F}_p}{2} \times \mathbb{F}_p}, N^{2,2} \in \mathbb{R}^{\binom{\mathbb{F}_p}{2} \times \binom{\mathbb{F}_p}{2}}$ have entries

$$N^{1,1}_{a,b} = \begin{cases} \alpha_1 - \alpha_1^2 & \text{if } a = b, \\ \alpha_2 \mathbb{1}_2(\{a,b\}) - \alpha_1^2 & \text{if } a \neq b, \end{cases} \tag{25}$$

$$N^{1,2}_{a,\{b,c\}} = \begin{cases} (\alpha_2 - \alpha_1\alpha_2)\,\mathbb{1}_2(\{b,c\}) & \text{if } a \in \{b,c\}, \\ \alpha_3 \mathbb{1}_3(\{a,b,c\}) - \alpha_1\alpha_2 \mathbb{1}_2(\{b,c\}) & \text{if } a \notin \{b,c\}, \end{cases} \tag{26}$$

$$N^{2,2}_{\{a,b\},\{c,d\}} = \begin{cases} (\alpha_2 - \alpha_2^2)\mathbb{1}_2(\{a,b\}) & \text{if } \{a,b\} = \{c,d\}, \\ \alpha_3 \mathbb{1}_3(\{a,b\} \cup \{c,d\}) - \alpha_2^2 \mathbb{1}_2(\{a,b\})\mathbb{1}_2(\{c,d\}) & \text{if } |\{a,b\} \cap \{c,d\}| = 1, \\ \alpha_4 \mathbb{1}_4(\{a,b,c,d\}) - \alpha_2^2 \mathbb{1}_2(\{a,b\})\mathbb{1}_2(\{c,d\}) & \text{if } \{a,b\} \cap \{c,d\} = \varnothing. \end{cases} \tag{27}$$

Per Remark 4, rows and columns indexed by pairs are identically zero in any of these matrices for all pairs that are not edges in G_p.

Next, we define matrices $H^{\bullet,\bullet}$ based on the $N^{\bullet,\bullet}$ by replacing the clique indicator functions with "bipartite" versions of those indicator functions, that only depend on the presence of edges between two subsets of vertices.

▶ **Definition 12.** *We write $\mathbb{1}_{\ell,r} : \binom{\mathbb{F}_p}{\ell} \times \binom{\mathbb{F}_p}{r} \to \{0,1\}$ for the function with*

$$\mathbb{1}_{\ell,r}(L,R) = \begin{cases} 1 & \text{if } v \sim_{G_p} w \text{ for all } v \in L \setminus R, w \in R \setminus L, \\ 0 & \text{otherwise.} \end{cases} \tag{28}$$

In other words, $\mathbb{1}_{\ell,r}(L,R) = 1$ if and only if all pairs of vertices in $\binom{L \cup R}{2}$ that don't belong simultaneously to L or R are connected in G_p.

Now we are ready to state what matrix H is: it is given by blocks $H^{1,1} \in \mathbb{R}^{\mathbb{F}_p \times \mathbb{F}_p}$, $H^{1,2} \in \mathbb{R}^{\mathbb{F}_p \times \binom{\mathbb{F}_p}{2}}$, $H^{2,1} = H^{1,2\top}$, and $H^{2,2} \in \mathbb{R}^{\binom{\mathbb{F}_p}{2} \times \binom{\mathbb{F}_p}{2}}$ having entries

$$H^{1,1}_{a,b} = \begin{cases} \alpha_1 - \alpha_1^2 & \text{if } a = b \\ \alpha_2 \mathbb{1}_{1,1}(\{a\}, \{b\}) - \alpha_1^2 & \text{if } a \neq b \end{cases}, \tag{29}$$

$$H^{1,2}_{a,\{b,c\}} = \begin{cases} \alpha_2 - \alpha_1\alpha_2 & \text{if } a \in \{b,c\} \\ \alpha_3 \mathbb{1}_{1,2}(\{a\}, \{b,c\}) - \alpha_1\alpha_2 & \text{if } a \notin \{b,c\} \end{cases}, \tag{30}$$

$$H^{2,2}_{\{a,b\},\{c,d\}} = \begin{cases} \alpha_2 - \alpha_2^2 & \text{if } \{a,b\} = \{c,d\} \\ \alpha_3 \mathbb{1}_{2,2}(\{a,b\}, \{c,d\}) - \alpha_2^2 & \text{if } |\{a,b\} \cap \{c,d\}| = 1 \\ \alpha_4 \mathbb{1}_{2,2}(\{a,b\}, \{c,d\}) - \alpha_2^2 & \text{if } \{a,b\} \cap \{c,d\} = \varnothing \end{cases}. \tag{31}$$

It is easy to see that proving positive semidefiniteness for H also proves N is positive semidefinite, due to the following observation.

▶ **Proposition 13.** *Up to permutation of rows and columns, N is the direct sum of the principal submatrix of H indexed by singletons and the edges of G_p with a zero matrix.*

The proof is simply that, for $|L|, |R| \leq 2$, we have $\mathbb{1}_{|L \cup R|}(L \cup R) = \mathbb{1}_{|L|,|R|}(L, R)$ so long as L is an edge if $|L| = 2$ and R is an edge if $|R| = 2$.

3.2 Second Schur Complement Bounds

Next, the goal is to prove under the same setting of Theorem 9 that $H \succeq 0$. The argument for this analysis is included in the full version of this paper and is similar to that of [9].

We will use $Q_0 = \frac{1}{p} J \in \mathbb{R}^{\mathbb{F}_p \times \mathbb{F}_p}$ to denote the orthogonal projection matrix to the constant vector, and $Q_1 = I - Q_0$ to denote the projection matrix to the orthogonal complement.

▶ **Proposition 14.** *Under the FK pseudomoments specified by $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ in Theorem 9, for any constant $\varepsilon > 0$, the matrix $H^{1,1}$ satisfies*

$$H^{1,1} \succeq \left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right) Q_0 + (1-\varepsilon)\alpha_1 Q_1 \succ 0 \tag{32}$$

for all sufficiently large primes p.

So, if moreover we can show $H^{2,2} - H^{2,1}(H^{1,1})^{-1}H^{1,2} \succeq 0$, we can conclude the positive semidefiniteness of H. Our last simplification before proceeding to the main technical analysis is to remove the $(H^{1,1})^{-1}$ term above. Fix some constant $\varepsilon > 0$ for all future discussions, say $\varepsilon := \frac{1}{2}$. Then,

$$H^{1,1} \succeq \left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right) Q_0 + (1-\varepsilon)\alpha_1 Q_1 \succ 0 \tag{33}$$

for all sufficiently large primes p, so

$$(H^{1,1})^{-1} \preceq \left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right)^{-1} Q_0 + ((1-\varepsilon)\alpha_1)^{-1} Q_1, \tag{34}$$

and substituting this into the term appearing in the inequality we need to show,

$$H^{2,1}(H^{1,1})^{-1}H^{1,2} \preceq H^{2,1}\left[\left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right)^{-1} Q_0 + ((1-\varepsilon)\alpha_1)^{-1} Q_1\right] H^{1,2}. \tag{35}$$

Note that the column sum (row sum) of $H^{2,1}$ is the same across each column (nonzero row indexed by edges of Paley graphs) due to the partial symmetry of Paley graphs. As a result, $\mathbf{1}$ is an eigenvector of $H^{2,1}H^{1,2}$, and $H^{2,1}Q_0H^{1,2} = P_0H^{2,1}H^{1,2}P_0$, where we use $P_0 = \frac{2}{p(p-1)}J \in \mathbb{R}^{\binom{\mathbb{F}_p}{2}\times\binom{\mathbb{F}_p}{2}}$ to denote the orthogonal projection matrix to the constant vector. Moreover, since $\mathbf{1}$ is an eigenvector of $H^{2,1}H^{1,2}$, $(I-P_0)H^{2,1}H^{1,2}P_0 = 0$. We therefore have

$$\begin{aligned}
&H^{2,1}\left[\left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right)^{-1} Q_0 + ((1-\varepsilon)\alpha_1)^{-1} Q_1\right] H^{1,2} \\
&= H^{2,1}\left[\left(\left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right)^{-1} - ((1-\varepsilon)\alpha_1)^{-1}\right) Q_0 + ((1-\varepsilon)\alpha_1)^{-1} I\right] H^{1,2} \\
&= \left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right)^{-1} P_0H^{2,1}H^{1,2}P_0 + ((1-\varepsilon)\alpha_1)^{-1}(I-P_0)H^{2,1}H^{1,2}(I-P_0).
\end{aligned} \tag{36}$$

Thus, to show $H^{2,2} \succeq H^{2,1}(H^{1,1})^{-1}H^{1,2}$ holds for all sufficiently large primes p, it is sufficient to prove the following proposition:

▶ **Proposition 15.** *Under the FK pseudomoments specified by $\alpha_1, \alpha_2, \alpha_3, \alpha_4$ in Theorem 9, for any constant $\varepsilon > 0$,*

$$\begin{aligned}
H^{2,2} \succeq &\left(\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2\right)^{-1} P_0H^{2,1}H^{1,2}P_0 \\
&+ ((1-\varepsilon)\alpha_1)^{-1}(I-P_0)H^{2,1}H^{1,2}(I-P_0)
\end{aligned} \tag{37}$$

holds for all sufficiently large primes p.

3.3 Ribbons and Graph Matrices

To organize the remaining calculation, now let us review the construction of *graph matrices* that has played a role in many SOS lower bound analyses in previous literature. We will use the following definitions as appeared in the work of [29].

▶ **Definition 16** (Ribbon). *A ribbon on a ground set V is a tuple $R = (V(R), E(R), A_R, B_R)$, where $(V(R), E(R))$ is a graph, and $A_R, B_R \subseteq V(R) \subseteq V$.*

▶ **Definition 17** (Matrix for a Ribbon). *Let $G \in \mathbb{R}^{V\times V}$ be a real symmetric matrix whose off-diagonal entries are ± 1 and whose diagonal entries are zero. For $R = (V(R), E(R), A_R, B_R)$ a on V, the corresponding matrix $M_G(R) \in \mathbb{R}^{\binom{V}{|A_R|}\times\binom{V}{|B_R|}}$ has rows and columns indexed by the subsets of V of sizes $|A_R|$ and $|B_R|$, respectively. The entries of $M_G(R)$ is given by*

$$M_G(R)_{I,J} = \begin{cases} \prod_{\{i,j\}\in E(R)} G_{i,j} & \text{if } I = A_R \text{ and } J = B_R \\ 0 & \text{otherwise} \end{cases}. \tag{38}$$

In other words, there is only one nonzero entry of $M_G(R)$, and it is located at the row and the column corresponding to A_R and B_R.

▶ **Definition 18** (Isomorphisms Between Ribbons). *Two ribbons R, S are isomorphic, or have the same shape, if there is a bijection $f : V(R) \to V(S)$ which is a graph isomorphism between $(V(R), E(R))$ and $(V(S), E(S))$ and also a bijection from A_R to A_S and from B_R to B_S.*

If we ignore the labels on the vertices of a ribbon, what remains is the shape of the ribbon.

▶ **Definition 19** (Shape). *A* shape *is an equivalence class of ribbons of the same shape. Each shape has associated with it a representative* $\beta = (V(\beta), E(\beta), A_\beta, B_\beta)$.

▶ **Definition 20** (Embedding of a Shape). *Given a shape* β *on* V *and an injective function* $f : V(\beta) \to V$*, we let* $f(\beta)$ *be the ribbon by labeling the vertices* $V(\beta)$ *in the natural way.*

▶ **Definition 21** (Graph Matrix). *Let* $G \in \mathbb{R}^{V \times V}$ *be a real symmetric matrix whose off-diagonal entries are* ± 1 *and whose diagonal entries are zero. For a shape* β *on* V*, the graph matrix* $M_G(\beta) \in \mathbb{R}^{\binom{V}{|A_\beta|} \times \binom{V}{|B_\beta|}}$ *is defined as the sum of all ribbon matrices over ribbons with shape* β*:*

$$M_G(\beta) = \sum_{R \text{ ribbon of shape } \beta} M_G(R). \tag{39}$$

▶ **Definition 22** (Automorphism of a Shape). *For a shape* β*,* $\mathrm{Aut}(\beta)$ *is the group of bijection from* $V(\beta)$ *to itself such that* A_β *and* B_β *are fixed as sets and the map is a graph automorphism of* $(V(\beta), E(\beta))$*.*

It is easy to see that if we sum over ribbon matrices of all ribbons obtained from injective labelings of β, we obtain the graph matrix $M_G(\beta)$ multiplied by $|\mathrm{Aut}(\beta)|$. Thus,

$$M_G(\beta) = \sum_{R \text{ ribbon of shape } \beta} M_G(R) = \frac{1}{|\mathrm{Aut}(\beta)|} \sum_{f : V(\beta) \to V \text{ injective}} M_G(f(\beta)). \tag{40}$$

3.4 Graph Matrix Decomposition

▶ **Definition 23** (Legendre Symbol). *Let* $\mathbb{F}_p$ *be the finite field of order* p*. The Legendre symbol is defined as*

$$\chi(a) = \chi_p(a) := \begin{cases} 0 & \text{if } a \equiv 0 \pmod p, \\ 1 & \text{if } a \text{ is a quadratic residue in } \mathbb{F}_p, \\ -1 & \text{if } a \text{ is a quadratic nonresidue in } \mathbb{F}_p. \end{cases} \tag{41}$$

When the underlying finite field $\mathbb{F}_p$ *is fixed and clear from context, we will omit the subscript.*

▶ Remark 24. Recall that all the primes p in our discussion are congruent to 1 modulo 4. This ensures that $\chi(-1) = 1$, and thus $\chi(a) = \chi(-a)$ for any $a \in \mathbb{F}_p$.

▶ **Proposition 25.** *We have* $\mathbb{1}_{\ell,r}(L, R) = \frac{1}{2^{|L \setminus R| \times |R \setminus L|}} \prod_{(a,b) \in (L \setminus R) \times (R \setminus L)} (1 + \chi(a - b))$ *for all* $\ell, r \geq 0$*,* $L \in \binom{\mathbb{F}_p}{\ell}$*, and* $R \in \binom{\mathbb{F}_p}{r}$*.*

Proof. The result follows from observing that, for $a, b \in \mathbb{F}_p$ distinct, $\frac{1}{2}(1 + \chi(a - b))$ is the indicator of the edge $\{a, b\}$ existing in the Paley graph. ◀

In the following few equations, let us write S for the Seidel adjacency matrix of G_p, so that $S_{a,b} := \chi(a - b)$. By substituting the indicator functions $\mathbb{1}_{\ell,r}$ in the definition of H using Proposition 25 and expanding the products, we have

$$H^{2,2}_{\{a,b\},\{c,d\}} = \begin{cases} \alpha_2 - \alpha_2^2 & \text{if } \{a,b\} = \{c,d\}, \\ \left(\frac{\alpha_3}{2} - \alpha_2^2\right) + \frac{\alpha_3}{2} S_{b,d} & \text{if } a = c \text{ and } b \neq d, \\ \left(\frac{\alpha_4}{16} - \alpha_2^2\right) + \frac{\alpha_4}{16}\big(S_{a,c} + S_{a,d} + S_{b,c} + S_{b,d} \\ \quad + S_{a,c}S_{a,d} + S_{b,c}S_{b,d} + S_{a,c}S_{b,c} \\ \quad + S_{a,d}S_{b,d} + S_{a,c}S_{b,d} + S_{a,d}S_{b,c} \\ \quad + S_{a,c}S_{a,d}S_{b,c} + S_{b,d}S_{a,d}S_{b,c} \\ \quad + S_{a,c}S_{a,d}S_{b,d} + S_{a,c}S_{b,c}S_{b,d} \\ \quad + S_{a,c}S_{a,d}S_{b,c}S_{b,d}\big) & \text{if } \{a,b\} \cap \{c,d\} = \varnothing. \end{cases} \tag{42}$$

and

$$\begin{aligned} &(H^{2,1}H^{1,2})_{\{a,b\},\{c,d\}} \\ &= \sum_{i \in \mathbb{F}_p} H^{2,1}_{\{a,b\},i} H^{1,2}_{i,\{c,d\}} \\ &= \begin{cases} 2(\alpha_2 - \alpha_1\alpha_2)^2 + (p-2)((\alpha_1\alpha_2)^2 + \frac{\alpha_3^2}{4} - \frac{\alpha_1\alpha_2\alpha_3}{2}) \\ \quad + (\frac{\alpha_3^2}{4} - \frac{\alpha_1\alpha_2\alpha_3}{2}) \sum_{i \in \mathbb{F}_p \setminus \{a,b\}} (S_{a,i} + S_{b,i} + S_{a,i}S_{b,i}) & \text{if } \{a,b\} = \{c,d\}, \\ (\alpha_2 - \alpha_1\alpha_2)^2 - 2(\alpha_2 - \alpha_1\alpha_2)\alpha_1\alpha_2 + (p-3)(\alpha_1\alpha_2)^2 \\ + \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{2} - (p-3)\frac{\alpha_1\alpha_2\alpha_3}{2} + (p-3)\frac{\alpha_3^2}{8} \\ + \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{4}(S_{a,b}S_{b,d} + S_{a,d}S_{b,d} + S_{a,b} + S_{a,d} + 2S_{b,d}) \\ + (\frac{\alpha_3^2}{8} - \frac{\alpha_1\alpha_2\alpha_3}{2}) \sum_{i \notin \{a,b,d\}} S_{a,i} & \text{if } a = c \text{ and } b \neq d, \\ + (\frac{\alpha_3^2}{8} - \frac{\alpha_1\alpha_2\alpha_3}{4}) \sum_{i \notin \{a,b,d\}} (S_{b,i} + S_{d,i} \\ \quad + S_{a,i}S_{b,i} + S_{a,i}S_{d,i}) \\ + \frac{\alpha_3^2}{8} \sum_{i \in \mathbb{F}_p \setminus \{a,b,d\}} (S_{b,i}S_{d,i} + S_{a,i}S_{b,i}S_{d,i}) \\ (\alpha_2 - \alpha_1\alpha_2)\alpha_3 - 4(\alpha_2 - \alpha_1\alpha_2)\alpha_1\alpha_2 \\ + (p-4)(\alpha_1\alpha_2)^2 - (p-4)\frac{\alpha_1\alpha_2\alpha_3}{2} + (p-4)\frac{\alpha_3^2}{16} \\ + \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{2}(S_{a,c} + S_{a,d} + S_{b,c} + S_{b,d}) \\ + \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{4}(S_{a,c}S_{a,d} + S_{b,c}S_{b,d} + S_{a,c}S_{b,c} + S_{a,d}S_{b,d}) \\ + (\frac{\alpha_3^2}{16} - \frac{\alpha_1\alpha_2\alpha_3}{4}) \sum_{i \notin \{a,b,c,d\}} (S_{a,i} + S_{b,i} + S_{c,i} + S_{d,i} & \text{if } \{a,b\} \cap \{c,d\} = \varnothing. \\ \quad + S_{a,i}S_{b,i} + S_{c,i}S_{d,i}) \\ + \frac{\alpha_3^2}{16} \sum_{i \notin \{a,b,c,d\}} (S_{a,i}S_{c,i} + S_{a,i}S_{d,i} + S_{b,i}S_{c,i} + S_{b,i}S_{d,i} \\ \quad + S_{a,i}S_{b,i}S_{c,i} + S_{a,i}S_{b,i}S_{d,i} \\ \quad + S_{a,i}S_{c,i}S_{d,i} + S_{b,i}S_{c,i}S_{d,i} \\ \quad + S_{a,i}S_{b,i}S_{c,i}S_{d,i}) \end{cases} \end{aligned} \tag{43}$$

We now express this as a sum of graph matrices. We present all the matrices required for this decomposition in Table 1. Using the notations for graph matrices defined above and in the table, we can write the matrix $H^{2,2}$ and the matrix $H^{2,1}H^{1,2}$ as a weighted sum of these matrices, as follows:

$$H^{2,2} = (\alpha_2 - \alpha_2^2)I + \left(\frac{\alpha_3}{2} - \alpha_2^2\right) T^{3,0,1} + \frac{\alpha_3}{2} T^{3,1,1} + \left(\frac{\alpha_4}{16} - \alpha_2^2\right) T^{4,0,1}$$
$$+ \frac{\alpha_4}{16}\left(T^{4,1,1} + T^{4,2,1} + T^{4,2,2} + T^{4,2,3} + T^{4,3,1} + T^{4,4,1}\right), \tag{44}$$

$$H^{2,1}H^{1,2} = \left[2(\alpha_2 - \alpha_1\alpha_2)^2 + (p-2)\left((\alpha_1\alpha_2)^2 + \frac{\alpha_3^2}{4} - \frac{\alpha_1\alpha_2\alpha_3}{2}\right)\right] I$$
$$+ \left(\frac{\alpha_3^2}{4} - \frac{\alpha_1\alpha_2\alpha_3}{2}\right)\left(U^{3,1,1} + U^{3,2,1}\right)$$
$$+ \left[(\alpha_2 - \alpha_1\alpha_2)\left(\alpha_2 - 3\alpha_1\alpha_2 + \frac{\alpha_3}{2}\right) + (p-3)\left((\alpha_1\alpha_2)^2 - \frac{\alpha_1\alpha_2\alpha_3}{2} + \frac{\alpha_3^2}{8}\right)\right] T^{3,0,1}$$
$$+ \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{4}(T^{3,2,1} + T^{3,2,2} + 2T^{3,1,1} + T^{3,1,2} + T^{3,1,3})$$
$$+ \left(\frac{\alpha_3^2}{8} - \frac{\alpha_1\alpha_2\alpha_3}{2}\right) U^{4,1,1}$$
$$+ \left(\frac{\alpha_3^2}{8} - \frac{\alpha_1\alpha_2\alpha_3}{4}\right)\left(U^{4,1,2} + U^{4,1,3} + U^{4,2,1} + U^{4,2,2}\right) + \frac{\alpha_3^2}{8}\left(U^{4,2,3} + U^{4,3,1}\right)$$
$$+ \left[(\alpha_2 - \alpha_1\alpha_2)(\alpha_3 - 4\alpha_1\alpha_2) + (p-4)\left(\alpha_1\alpha_2 - \frac{\alpha_3}{4}\right)^2\right] T^{4,0,1}$$
$$+ \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{2} T^{4,1,1} + \frac{(\alpha_2 - \alpha_1\alpha_2)\alpha_3}{4}(T^{4,2,1} + T^{4,2,2})$$
$$+ \left(\frac{\alpha_3^2}{16} - \frac{\alpha_1\alpha_2\alpha_3}{4}\right)\left(U^{5,1,1} + U^{5,1,2} + U^{5,2,1} + U^{5,2,2}\right)$$
$$+ \frac{\alpha_3^2}{16}\left(U^{5,2,3} + U^{5,3,1} + U^{5,3,2} + U^{5,4,1}\right). \tag{45}$$

3.5 Graph Matrix Norm Bounds

Now we analyze the norms of the graph matrices defined above in order to prove Proposition 15.

▶ Remark 26. Previous work of [1] established the typical norm of graph matrices when the underlying matrix G is the Seidel adjacency matrix of an ER random graph, where the quantities that characterize the norm bounds are the sizes of the minimum vertex separators of the shapes. In this work, using different techniques, we prove graph matrix norm bounds when the underlying matrix is the Seidel adjacency matrix of the Paley graph G_p.

Recall that we defined $P_0 = \frac{1}{p(p-1)} J \in \mathbb{R}^{\binom{\mathbb{F}_p}{2} \times \binom{\mathbb{F}_p}{2}}$ to denote the orthogonal projection matrix to the constant vector. Following the strategies in [9], we define the following subspaces of $\mathbb{R}^{\binom{\mathbb{F}_p}{2}}$:

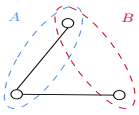
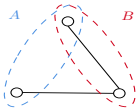
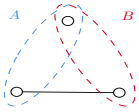
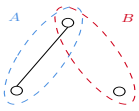
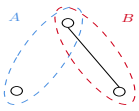
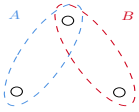
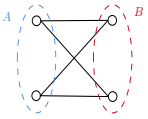
$$\mathbb{V}_0 = \left\{ v \in \mathbb{R}^{\binom{\mathbb{F}_p}{2}} : v_{i,j} = v_{i',j'}, \quad \forall \{i,j\}, \{i',j'\} \in \binom{\mathbb{F}_p}{2} \right\} \tag{46}$$

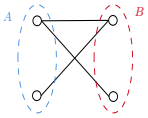
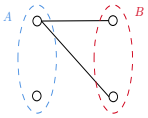
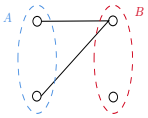
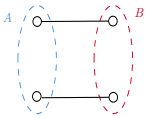
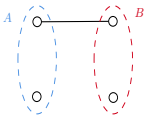
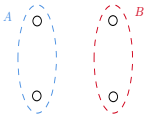
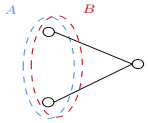
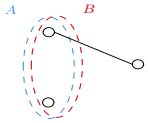
$$\mathbb{V}_1 = \left\{ v \in \mathbb{R}^{\binom{\mathbb{F}_p}{2}} : \exists u \in \mathbb{R}^{\mathbb{F}_p}, \text{s.t. } \langle \mathbf{1}, u \rangle = 0 \text{ and } v_{\{i,j\}} = u_i + u_j, \quad \forall \{i,j\} \in \binom{\mathbb{F}_p}{2} \right\} \tag{47}$$

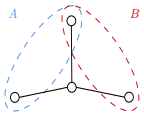
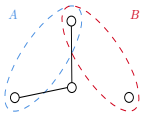
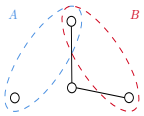
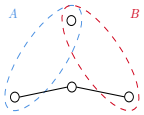
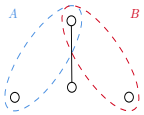
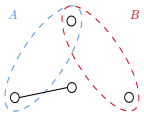
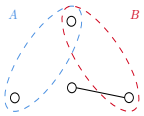
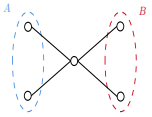
$$\mathbb{V}_2 = (\mathbb{V}_0 \oplus \mathbb{V}_1)^{\perp}. \tag{48}$$

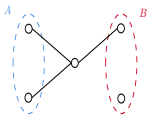
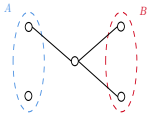
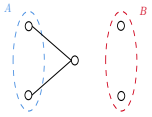
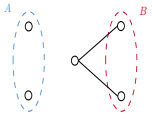
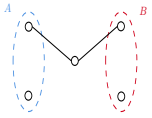
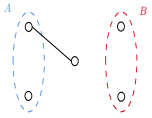
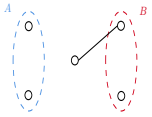
In words, $\mathbb{V}_0$ is the span of constant vectors, $\mathbb{V}_0 \oplus \mathbb{V}_1$ is the span of vectors v whose entries $v_{\{i,j\}}$ can be decomposed to a sum of $u_i + u_j$ for some $u \in \mathbb{R}^{\mathbb{F}_p}$, and $\mathbb{V}_2$ is the orthogonal

Table 1 We present the graph matrices that we consider in Section 3.5 for the proof of Theorem 2, all defined on the Seidel adjacency matrix S of G_p. For each matrix, we give its name, the associated shape (see Definition 19), and the formula for the entries of the matrix. Some matrices are only non-zero on index sets satisfying certain equalities; in this case, for the sake of brevity, we indicate this "pattern" in the first column, and do not include the requisite indicator function in the third column. We also give the norm bound we prove in Section 3.5 and the norm bound for the same graph matrix evaluated on an ER random graph that follows from [1]. In these bounds we give only the order of growth; our bounds should be viewed as having an implicit $O(\cdot)$, and the bounds of [1] as having an implicit $\widetilde{O}(\cdot)$.

Matrix	Shape	Entry Formula	G_p	$\mathcal{G}(p, \frac{1}{2})$
$T^{3,2,1}_{\{a,b\},\{a,c\}}$		$S_{a,b}S_{b,c}$	$p^{1/2}$	$p^{1/2}$
$T^{3,2,2}_{\{a,b\},\{a,c\}}$		$S_{a,c}S_{b,c}$	$p^{1/2}$	$p^{1/2}$
$T^{3,1,1}_{\{a,b\},\{a,c\}}$		$S_{b,c}$	$p^{1/2}$	$p^{1/2}$
$T^{3,1,2}_{\{a,b\},\{a,c\}}$		$S_{a,b}$	p	p
$T^{3,1,3}_{\{a,b\},\{a,c\}}$		$S_{a,c}$	p	p
$T^{3,0,1}_{\{a,b\},\{a,c\}}$		1	p	p
$T^{4,4,1}_{\{a,b\},\{c,d\}}$		$S_{a,c}S_{a,d}S_{b,c}S_{b,d}$	$p^{5/4}$	p

Matrix	Shape	Entry Formula	G_p	$\mathcal{G}(p, \frac{1}{2})$
$T^{4,3,1}_{\{a,b\},\{c,d\}}$		$S_{a,c}S_{a,d}S_{b,c} + S_{a,c}S_{a,d}S_{b,d} + S_{a,c}S_{b,c}S_{b,d} + S_{a,d}S_{b,c}S_{b,d}$	p	p
$T^{4,2,1}_{\{a,b\},\{c,d\}}$		$S_{a,c}S_{a,d} + S_{b,c}S_{b,d}$	$p^{3/2}$	$p^{3/2}$
$T^{4,2,2}_{\{a,b\},\{c,d\}}$		$S_{a,c}S_{b,c} + S_{a,d}S_{b,d}$	$p^{3/2}$	$p^{3/2}$
$T^{4,2,3}_{\{a,b\},\{c,d\}}$		$S_{a,c}S_{b,d} + S_{a,d}S_{b,c}$	p	p
$T^{4,1,1}_{\{a,b\},\{c,d\}}$		$S_{a,c} + S_{a,d} + S_{b,c} + S_{b,d}$	$p^{3/2}$	$p^{3/2}$
$T^{4,0,1}_{\{a,b\},\{c,d\}}$		1	p^2	p^2
$U^{3,2,1}_{\{a,b\},\{a,b\}}$		$\sum_{i \notin \{a,b\}} S_{a,i}S_{b,i}$	1	$p^{1/2}$
$U^{3,1,1}_{\{a,b\},\{a,b\}}$		$\sum_{i \notin \{a,b\}} S_{a,i} + S_{b,i}$	1	$p^{1/2}$

Matrix	Shape	Entry Formula	G_p	$\mathcal{G}(p, \frac{1}{2})$
$U^{4,3,1}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{a,i} S_{b,i} S_{c,i}$	$p^{3/2}$	p
$U^{4,2,1}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{a,i} S_{b,i}$	p	$p^{3/2}$
$U^{4,2,2}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{a,i} S_{c,i}$	p	$p^{3/2}$
$U^{4,2,3}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{b,i} S_{c,i}$	p	p
$U^{4,1,1}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{a,i}$	p	$p^{3/2}$
$U^{4,1,2}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{b,i}$	p	$p^{3/2}$
$U^{4,1,3}_{\{a,b\},\{a,c\}}$		$\sum_{i \notin \{a,b,c\}} S_{c,i}$	p	$p^{3/2}$
$U^{5,4,1}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{a,i} S_{b,i} S_{c,i} S_{d,i}$	p^2	p^2

Matrix	Shape	Entry Formula	G_p	$\mathcal{G}(p, \frac{1}{2})$
$U^{5,3,1}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{a,i}S_{b,i}S_{c,i} + S_{a,i}S_{b,i}S_{d,i}$	p^2	p^2
$U^{5,3,2}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{a,i}S_{c,i}S_{d,i} + S_{b,i}S_{c,i}S_{d,i}$	p^2	p^2
$U^{5,2,1}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{a,i}S_{b,i}$	p^2	$p^{5/2}$
$U^{5,2,2}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{c,i}S_{d,i}$	p^2	$p^{5/2}$
$U^{5,2,3}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{a,i}S_{c,i} + S_{a,i}S_{d,i} + S_{b,i}S_{c,i} + S_{b,i}S_{d,i}$	p^2	p^2
$U^{5,1,1}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{a,i} + S_{b,i}$	p^2	$p^{5/2}$
$U^{5,1,2}_{\{a,b\},\{c,d\}}$		$\sum_{i \notin \{a,b,c,d\}} S_{c,i} + S_{d,i}$	p^2	$p^{5/2}$

complement of $\mathbb{V}_0 \oplus \mathbb{V}_1$. Furthermore, let P_1 and P_2 be the orthogonal projection matrices to the subspaces $\mathbb{V}_1$, and $\mathbb{V}_2$ respectively. Note that this is consistent with the previously defined P_0, which is the orthogonal projection matrix to the span of constant vectors $\mathbb{V}_0$.

In the analysis of ER graphs, these subspaces appear because they are the decomposition of $\mathbb{R}^{\binom{\mathbb{F}_p}{2}}$ into irreducible subrepresentations under the action of S_p, with respect to which the expectation of an FK pseudomoment matrix is invariant. This invariance does not hold for our deterministic FK pseudomoment matrix, but we will see that the same decomposition is still useful.

We will use the following norm bounds for the graph matrices defined earlier. The proofs may be found in the full version of the paper.

▶ **Proposition 27.** $\|T^{3,2,i}\| = O(\sqrt{p})$ *for* $i \in \{1,2\}$.

▶ **Proposition 28.** $\|T^{3,1,1}\| = O(\sqrt{p})$.

▶ **Proposition 29.** $\|T^{3,1,i}\| = O(p)$ *for* $i \in \{2,3\}$.

▶ **Proposition 30.** $T^{3,0,1} = 2(p-2)P_0 + (p-4)P_1 - 2P_2$.

▶ **Proposition 31.** $\|T^{4,3,1}\| = O(p)$.

▶ **Proposition 32.** $\|T^{4,i,j}\| = O(p^{3/2})$ *for* $(i,j) \in \{(2,1),(2,2),(1,1)\}$. *Moreover, all of* $\|T^{4,2,1}P_2\|$, $\|P_2T^{4,2,2}\|$, $\|P_2T^{4,1,1}\|$, *and* $\|T^{4,1,1}P_2\|$ *are* $O(\sqrt{p})$.

▶ **Proposition 33.** $\|T^{4,2,3}\| = O(p)$.

▶ **Proposition 34.** $T^{4,0,1} = \frac{(p-2)(p-3)}{2}P_0 - (p-3)P_1 + P_2$.

▶ **Proposition 35.** $\|U^{3,i,1}\| = O(1)$ *for* $i \in \{1,2\}$.

▶ **Proposition 36.** $\|U^{4,3,1}\| = O(p^{3/2})$.

▶ **Proposition 37.** $\|U^{4,i,j}\| = O(p)$ *for* $i \in \{1,2\}$ *and* $j \in \{1,2,3\}$.

▶ **Proposition 38.** $\|U^{5,4,1}\| = O(p^2)$.

▶ **Proposition 39.** $\|U^{5,3,i}\| = O(p^2)$ *for* $i \in \{1,2\}$.

▶ **Proposition 40.** $\|U^{5,i,j}\| = O(p^2)$ *for* $i \in \{1,2\}$ *and* $j \in \{1,2,3\}$, *where* $j \neq 3$ *if* $i = 1$.

▶ **Theorem 41.** $\|T^{4,4,1}\| = O(p^{5/4})$.

Of these statements, Theorem 41 is by far the subtlest – unlike the other terms, where fairly straightforward arguments work, for $T^{4,4,1}$ it turns out that a naive bound is insufficient, and we must more carefully account for character sum cancellations. The bounds we prove are generally incomparable to those for random graphs following from [1]: for some graph matrices we expect a comparable norm bound but cannot prove one due to technical obstacles, while for other graph matrices the Paley graph exhibits stronger cancellations than a random graph and we can show a stronger norm bound. We compare the respective bounds in Table 1. Moreover, as we show in Section 4.3, there is an example of a graph matrix for which the norm when evaluated on the Paley graph is actually asymptotically larger than the norm when evaluated on a random graph; however, this example does not figure in our analysis.

3.6 Final Steps

Finally, putting all the graph matrix norm bounds together, we prove Proposition 15, which will conclude the proof of Theorem 9, as we have discussed earlier.

Proof of Proposition 15. The statements in this proof will hold for all sufficiently large primes p.

To show $H^{2,2} \succeq (\alpha_1 + \frac{p-1}{2}\alpha_2 - p\alpha_1^2)^{-1} P_0 H^{2,1} H^{1,2} P_0 + ((1-\varepsilon)\alpha_1)^{-1}(I - P_0) H^{2,1} H^{1,2} (I - P_0)$, we have to show that $M_1 \succeq M_2$, where M_1 is the sum of all multiples of the identity, $T^{3,0,1}$, and $T^{4,0,1}$ (possibly conjugated by P_0 or $I - P_0$) appearing in the expressions (44) and (45), and M_2 is the sum of the remaining graph matrices of shapes having at least one edge. Note that $\mathbb{V}_0, \mathbb{V}_1, \mathbb{V}_2$ are eigenspaces of M_1, with eigenvalues scaling as $(1 - o(1))6p^2\alpha_1^4, (1 - o(1))4p\alpha_1^3$, and $(1 - o(1))4\alpha_1^2$, respectively.

It is then sufficient to show

$$\begin{bmatrix} 3p^2\alpha_1^4 & 0 & 0 \\ 0 & 2p\alpha_1^3 & 0 \\ 0 & 0 & 2\alpha_1^2 \end{bmatrix} \overset{?}{\succeq} \begin{bmatrix} \|P_0 M_2 P_0\| & \|P_0 M_2 P_1\| & \|P_0 M_2 P_2\| \\ \|P_1 M_2 P_0\| & \|P_1 M_2 P_1\| & \|P_1 M_2 P_2\| \\ \|P_2 M_2 P_0\| & \|P_2 M_2 P_1\| & \|P_2 M_2 P_2\| \end{bmatrix}. \tag{49}$$

Using the graph matrix norm bounds above, we have for any $i \in \{0,1,2\}$ and $j \in \{0,1,2\}$ with $(i,j) \neq (2,2)$ that

$$\|P_i M_2 P_j\| = O(p^{3/2}\alpha_1^4), \tag{50}$$

and for the remaining case

$$\|P_2 M_2 P_2\| = O(p^2\alpha_1^5), \tag{51}$$

so we only need to prove that the following matrix is positive semidefinite:

$$\begin{bmatrix} 3p^2\alpha_1^4 - O(p^{3/2}\alpha_1^4) & -O(p^{3/2}\alpha_1^4) & -O(p^{3/2}\alpha_1^4) \\ -O(p^{3/2}\alpha_1^4) & 2p\alpha_1^3 - O(p^{3/2}\alpha_1^4) & -O(p^{3/2}\alpha_1^4) \\ -O(p^{3/2}\alpha_1^4) & -O(p^{3/2}\alpha_1^4) & 2\alpha_1^2 - O(p^2\alpha_1^5) \end{bmatrix}, \tag{52}$$

which is verified by taking the Schur complement and using diagonal dominance when $\alpha_1 = c \cdot p^{-2/3}$ for a sufficiently small constant c. ◀

With Proposition 15 proved, we have finished proving Theorem 9.

4 Ancillary Results

4.1 Optimality Over Feige-Krauthgamer Pseudomoments

In this section, we show that our lower bound is optimal over those achievable by FK pseudomoments. To be precise, let us define a new SDP corresponding to this restricted type of pseudomoment, a variant of (13):

$$\mathrm{FK}_4(G) := \left\{ \begin{array}{ll} \text{maximize} & \sum_{i=1}^n M_{\varnothing,i}^{0,1} \\ \text{subject to} & M^{r,c} \in \mathbb{R}^{\binom{[n]}{r} \times \binom{[n]}{c}} \text{ for } r,c \in \{0,1,2\}, \\ & M_{S,T}^{r,c} \text{ depends only on } S \cup T, \\ & M_{S,T}^{r,c} = 0 \text{ whenever } S \cup T \notin \mathcal{K}(G), \\ & M_{S,T}^{r,c} \text{ depends only on } |S \cup T| \text{ when } S \cup T \in \mathcal{K}(G), \\ & M = \begin{bmatrix} 1 & M^{0,1} & M^{0,2} \\ M^{1,0} & M^{1,1} & M^{1,2} \\ M^{2,0} & M^{2,1} & M^{2,2} \end{bmatrix} \succeq 0 \end{array} \right\}. \tag{53}$$

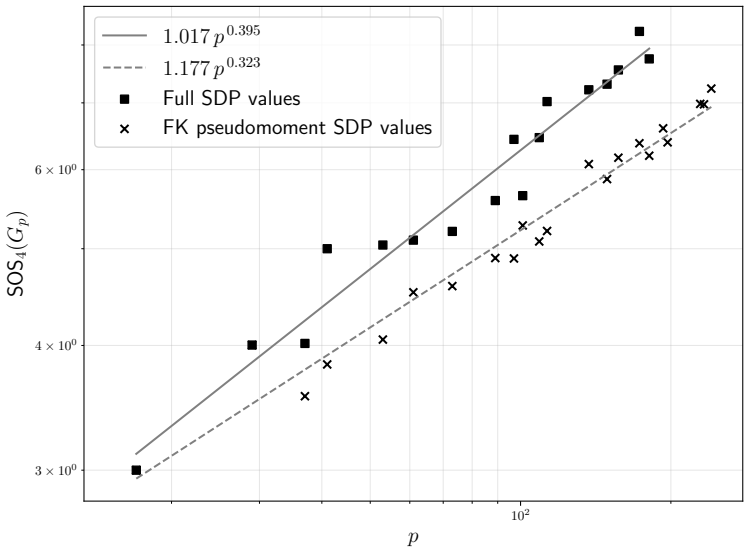


Figure 1 For primes $5 \leq p \leq 250$, we present the value of $\mathrm{SOS}_4(G_p)$ and the value of $\mathrm{FK}_4(G_p)$ (where the semidefinite program is restricted to optimize over only FK pseudomoments). We fit power models ap^b to the data and plot the results as well.

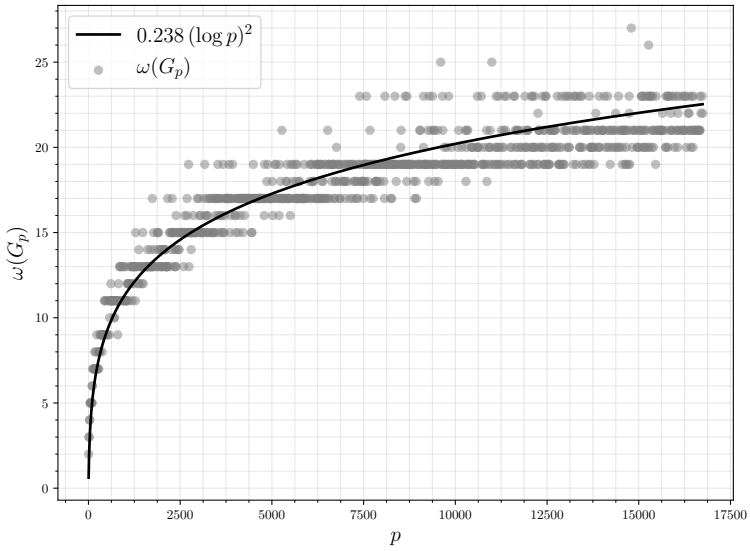


Figure 2 For primes $5 \leq p \leq 16741$, we present computations of the true clique number $\omega(G_p)$ (taken from [46] and its online supplementary materials). We fit a model $a(\log p)^2$ to the data and plot the results as well.

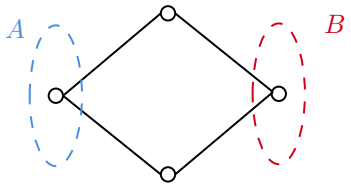


Figure 3 We illustrate the graph matrix used as an example in Section 4.3.

Since the conditions of this SDP are more restrictive than those of $\mathrm{SOS}_4(G)$, we have $\mathrm{SOS}_4(G) \geq \mathrm{FK}_4(G)$. Our strategy has been to show that $\mathrm{FK}_4(G)$ is large; the following shows a limitation to this approach. The proof is given in the full version of the paper.

▶ **Theorem 42.** *Over primes* $p \equiv 1 \pmod 4$, $\mathrm{FK}_4(G_p) = \Theta(p^{1/3})$.

4.2 Numerical Experiments

Given our results in Theorems 2 and 42, it is natural to ask whether a better lower bound technique than working with FK pseudomoments might prove an optimal lower bound of the form $\mathrm{SOS}_4(G_p) = \Omega(p^{1/2})$. In Figure 1, we present some surprising numerical results suggesting that this is *not* the case. Namely, in addition to the true values of $\omega(G_p)$, we plot the values of $\mathrm{SOS}_4(G_p)$ (the "full SDP") and of $\mathrm{FK}_4(G_p)$ (the "FK pseudomoment SDP") on a log-log plot, and fit lines to these results.[8]

These results for $\mathrm{FK}_4(G_p)$ confirm the statement of Theorem 42, with an estimated scaling of $\mathrm{FK}_4(G_p) \sim p^{0.323}$, close to our result showing that $\mathrm{FK}_4(G_p) \sim p^{1/3}$. For $\mathrm{SOS}_4(G_p)$, the results still indicate a scaling below $p^{1/2}$, estimated at $\mathrm{SOS}_4(G_p) \sim p^{0.395}$. Based on these results, it seems reasonable to conjecture that $\mathrm{SOS}_4(G_p) = O(p^{1/2-\varepsilon})$ for some $\varepsilon > 0$. This prediction is compatible with that of [33], who, based experiments solving a weaker SDP than degree 4 SOS as proposed by [21], experimentally found that $\mathrm{SOS}_4(G_p) \lesssim p^{0.456}$.

4.3 General Graph Matrix Norm Bounds Do Not Derandomize

In this section, we give a simple example of a graph matrix for which the norm bound of [1] for ER graphs fails to hold for Paley graphs. Since the bound of [1] is a crucial ingredient in the proof of the $\Omega(p^{1/2})$ SOS lower bound of [4], we take this as some evidence that a sufficiently high degree of SOS can prove a bound of the form $\omega(G_p) \leq O(p^{1/2-\varepsilon})$. In particular, this gives theoretical evidence for the numerical observations above.

Let $S \in \mathbb{R}^{n \times n}$ be the Seidel adjacency matrix of a graph. We consider the graph matrix $M = M(S)$ formed from S and the shape in Figure 3, with entries $M_{xy} = \mathbb{1}\{x \neq y\} \sum_{\substack{a,b \in [n] \\ a \neq b}} S_{a,x} S_{a,y} S_{b,x} S_{b,y}$, where we do not need to include the constraints $a, b \notin \{x, y\}$ since these are automatically enacted by having $S_{a,a} = 0$ for all a.

For any such S and $x \neq y$, we have $M_{xy} = (S^2)^2_{x,y} - (p-2)$. When S is the Seidel adjacency matrix of the Paley graph, we have $S^2 = pI - \mathbf{1}\mathbf{1}^\top$. Thus in this case we have $M(S) = (p-3)I - (p-3)\mathbf{1}\mathbf{1}^\top$, and $\|M\| = (p-1)(p-3) \sim p^2$. On the other hand, when S is the Seidel adjacency matrix of a random ER graph, then the results of [1] show that, since the shape of M has minimum vertex separator of size 1, with high probability, $\|M\| \leq \widetilde{O}(p^{3/2})$. Thus, the Paley graph adjacency matrix fails to satisfy this basic graph matrix bound.

[8] These SDPs are solved using the `Mosek` solver through the `CVXPY` interface for `Python`.

References

1 Kwangjun Ahn, Dhruv Medarametla, and Aaron Potechin. Graph matrices: Norm bounds and applications. *arXiv preprint*, 2016. arXiv:1604.03423.

2 Noga Alon, Michael Krivelevich, and Benny Sudakov. Finding a large hidden clique in a random graph. *Random Structures & Algorithms*, 13(3-4):457–466, 1998.

3 Christine Bachoc, Máté Matolcsi, and Imre Z Ruzsa. Squares and difference sets in finite fields. *Integers*, 13:A77, 2013.

4 Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.

5 Béla Bollobás. *Random graphs*. Cambridge University Press, second edition, 2001.

6 I Broere, D Döman, and JN Ridley. The clique numbers and chromatic numbers of certain Paley graphs. *Quaestiones Mathematicae*, 11(1):91–93, 1988.

7 Fan R. K. Chung, Ronald L. Graham, and Richard M. Wilson. Quasi-random graphs. *Combinatorica*, 9(4):345–362, 1989.

8 Ernie Croot and Vsevolod F Lev. Open problems in additive combinatorics. *Additive Combinatorics*, 43:207–233, 2007.

9 Yash Deshpande and Andrea Montanari. Improved sum-of-squares lower bounds for hidden clique and hidden submatrix problems. In *28th Annual Conference on Learning Theory (COLT 2015)*, pages 523–562, 2015.

10 Daniel Di Benedetto, József Solymosi, and Ethan P White. On the directions determined by a Cartesian product in an affine Galois plane. *Combinatorica*, 41(6):755–763, 2021.

11 Irit Dinur, Yuval Filmus, Prahladh Harsha, and Madhur Tulsiani. Explicit SoS lower bounds from high-dimensional expanders. *arXiv preprint*, 2020. arXiv:2009.05218.

12 Paul Erdös and George Szekeres. A combinatorial problem in geometry. *Compositio Mathematica*, 2:463–470, 1935.

13 RJ Evans, JR Pulham, and J Sheehan. On the number of complete subgraphs contained in certain graphs. *Journal of Combinatorial Theory, Series B*, 30(3):364–371, 1981.

14 Uriel Feige and Robert Krauthgamer. Finding and certifying a large hidden clique in a semirandom graph. *Random Structures & Algorithms*, 16(2):195–208, 2000.

15 Uriel Feige and Robert Krauthgamer. The probable value of the Lovász-Schrijver relaxations for maximum independent set. *SIAM Journal on Computing*, 32(2):345–370, 2003.

16 Laura Galli and Adam N Letchford. On the Lovász theta function and some variants. *Discrete Optimization*, 25:159–174, 2017.

17 David Gamarnik and Ilias Zadik. The landscape of the planted clique problem: Dense subgraphs and the overlap gap property. *arXiv preprint*, 2019. arXiv:1904.07174.

18 Sidney West Graham and CJ Ringrose. Lower bounds for least quadratic non-residues. In *Analytic number theory*, pages 269–309. Springer, 1990.

19 Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1-2):613–622, 2001.

20 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

21 Nebojša Gvozdenović, Monique Laurent, and Frank Vallentin. Block-diagonal semidefinite programming hierarchies for 0/1 programming. *Operations Research Letters*, 37(1):27–31, 2009.

22 Willem H Haemers. Interlacing eigenvalues and graphs. *Linear Algebra and its Applications*, 226:593–616, 1995.

23 Brandon Hanson and Giorgis Petridis. Refined estimates concerning sumsets contained in the roots of unity. *Proceedings of the London Mathematical Society*, 122(3):353–358, 2021.

24 Johan Hastad. Clique is hard to approximate within $n^{1-\varepsilon}$. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 627–636. IEEE, 1996.

25 Alan J Hoffman. On eigenvalues and colorings of graphs. In Bernard Harris, editor, *Graph Theory and its Applications*. Academic Press, 1970.

26 Max Hopkins and Ting-Chun Lin. Explicit lower bounds against $\omega(n)$-rounds of sum-of-squares. *arXiv preprint*, 2022. `arXiv:2204.11469`.

27 Samuel B Hopkins, Pravesh K Kothari, and Aaron Potechin. SOS and planted clique: Tight analysis of MPW moments at all degrees and an optimal lower bound at degree four. *arXiv preprint*, 2015. `arXiv:1507.05230`.

28 Mark Jerrum. Large cliques elude the Metropolis process. *Random Structures & Algorithms*, 3(4):347–359, 1992.

29 Chris Jones, Aaron Potechin, Goutham Rajendran, Madhur Tulsiani, and Jeff Xu. Sum-of-squares lower bounds for sparse independent set. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 406–416. IEEE, 2022.

30 Ferenc Juhász. The asymptotic behaviour of Lovász' theta function for random graphs. *Combinatorica*, 2(2):153–155, 1982.

31 Richard M Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972.

32 Richard M. Karp. The probabilistic analysis of some combinatorial search algorithms. In *Algorithms and complexity: New Directions and Recent Results*, 1976.

33 Vladimir A Kobzar and Krishnan Mody. Revisiting block-diagonal sdp relaxations for the clique number of the paley graphs. *arXiv preprint*, 2023. `arXiv:2304.08615`.

34 Luděk Kučera. Expected complexity of graph partitioning problems. *Discrete Applied Mathematics*, 57(2-3):193–212, 1995.

35 Monique Laurent. Lower bound for the number of iterations in semidefinite hierarchies for the cut polytope. *Mathematics of Operations Research*, 28(4):871–883, 2003.

36 Monique Laurent. Sums of squares, moment matrices and optimization over polynomials. In *Emerging Applications of Algebraic Geometry*, pages 157–270. Springer, 2009.

37 László Lovász. On the Shannon capacity of a graph. *IEEE Transactions on Information theory*, 25(1):1–7, 1979.

38 Mark Magsino, Dustin G Mixon, and Hans Parshall. Linear programming bounds for cliques in Paley graphs. In *Wavelets and Sparsity XVIII*, volume 11138, page 111381H. International Society for Optics and Photonics, 2019.

39 Raghu Meka, Aaron Potechin, and Avi Wigderson. Sum-of-squares lower bounds for planted clique. In *47th Annual ACM Symposium on Theory of Computing (STOC 2015)*, pages 87–96. ACM, 2015.

40 Raghu Meka and Avi Wigderson. Association schemes, non-commutative polynomial concentration, and sum-of-squares lower bounds for planted clique. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 20, page 10, 2013.

41 Hugh L Montgomery. *Topics in multiplicative number theory*, volume 227. Springer, 1971.

42 Rudi Mrazović. A random model for the Paley graph. *The Quarterly Journal of Mathematics*, 68(1):193–206, 2017.

43 Shuo Pang. SOS lower bound for exact planted clique. In *36th Computational Complexity Conference (CCC 2021)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2021.

44 Stanislaw Radziszowski. Small Ramsey numbers. *The Electronic Journal of Combinatorics*, 1000:DS1–Aug, 2011.

45 Prasad Raghavendra and Tselil Schramm. Tight lower bounds for planted clique in the degree-4 SOS program. *arXiv preprint*, 2015. `arXiv:1507.05136`.

46 James B Shearer. Lower bounds for small diagonal Ramsey numbers. *Journal of Combinatorial Theory, Series A*, 42(2):302–304, 1986.

47 Chi Hoi Yip. On the clique number of Paley graphs of prime power order. *Finite Fields and Their Applications*, 77:101930, 2022.

Sum-Of-Squares Lower Bounds for the Minimum Circuit Size Problem

Per Austrin
KTH Royal Institute of Technology, Stockholm, Sweden

Kilian Risse
EPFL, Lausanne, Switzerland

Abstract

We prove lower bounds for the Minimum Circuit Size Problem (MCSP) in the Sum-of-Squares (SoS) proof system. Our main result is that for every Boolean function $f : \{0,1\}^n \to \{0,1\}$, SoS requires degree $\Omega(s^{1-\epsilon})$ to prove that f does not have circuits of size s (for any $s > \mathsf{poly}(n)$). As a corollary we obtain that there are no low degree SoS proofs of the statement $\mathbf{NP} \not\subseteq \mathbf{P}/\mathsf{poly}$.

We also show that for any $0 < \alpha < 1$ there are Boolean functions with circuit complexity larger than 2^{n^α} but SoS requires size $2^{2^{\Omega(n^\alpha)}}$ to prove this. In addition we prove analogous results on the minimum *monotone* circuit size for monotone Boolean slice functions.

Our approach is quite general. Namely, we show that if a proof system Q has strong enough constraint satisfaction problem lower bounds that only depend on good expansion of the constraint-variable incidence graph and, furthermore, Q is expressive enough that variables can be substituted by local Boolean functions, then the MCSP problem is hard for Q.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases Proof Complexity, Sum of Squares, Minimum Circuit Size Problem

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.31

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2023/010/

Funding Supported by the Approximability and Proof Complexity project funded by the Knut and Alice Wallenberg Foundation.
Kilian Risse: Supported by Swiss National Science Foundation project 200021-184656 "Randomness in Problem Instances and Randomized Algorithms".

1 Introduction

Even before the dawn of complexity theory, there was an interest in the minimum circuit size problem (MCSP): given the truth table of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ and a parameter s, the MCSP problem asks whether there is a Boolean circuit of size at most s computing f. Despite many years of research, we do not know whether this problem is **NP**-hard. It clearly is in **NP**: given a circuit of size at most s (described by $O(s \log s)$ bits) we can easily check in time $O(s \cdot 2^n)$ whether this circuit indeed computes f.

Determining the hardness of MCSP itself turns out to be a difficult problem. Kabanets and Cai [14] showed that **NP**-hardness of the MCSP problem implies breakthrough circuit lower bounds. These lower bounds are not implausible but well out of reach of current techniques. In a similar vein Murray and Williams [19] showed that **NP**-hardness of MCSP implies that $\mathbf{EXP} \neq \mathbf{ZPP}$ and more recently Hirahara [13] proved that **NP**-hardness of MCSP implies a worst-case to average-case reduction for problems in **NP** (for an appropriate MCSP version).

On the other hand if one could show that MCSP is in $\mathbf{P}/\mathsf{poly}$, this would imply even stronger (though less realistic) results: Kabanets and Cai [14] also showed that if MCSP is in $\mathbf{P}/\mathsf{poly}$, then crypto-secure one way functions can be inverted on a considerable fraction of their range.

To summarize it seems unlikely that MCSP is in **P**, but showing that it is **NP**-hard implies very strong consequences. As these results seem out of reach for current techniques, it might be a more fruitful avenue to try to at least rule out that certain (families of) algorithms solve the MCSP problem efficiently.

This can be achieved very elegantly in proof complexity: show that some proof system capturing your algorithm requires long proofs to refute the claim that a complex function has a small circuit. This will then rule out that the algorithm in question can efficiently solve the MCSP problem. This will not only show that this specific algorithm requires long running time but would also show that any algorithm captured by this proof system requires long running time to solve the MCSP problem. Hence by this line of reasoning we manage to rule out entire classes of algorithms to solve the MCSP problem efficiently.

This paper focuses on the Sum of Squares proof system (SoS). This proof system provides certificates of unsatisfiability of systems of polynomial equations $\mathcal{P} = \{p_1 = 0, \ldots, p_m = 0\}$ over $\mathbb{R}$. In this paper we are only interested in Boolean systems of equations, meaning that $\mathcal{P}$ contains the equation $x^2 - x = 0$ for every variable x appearing in the system. A key complexity measure is the degree of a refutation, which is the maximum degree of a monomial occurring in the refutation of $\mathcal{P}$. All Boolean systems $\mathcal{P}$ over n variables have an SoS refutation of degree n and we are interested in the minimum degree that SoS requires to refute $\mathcal{P}$. An SoS refutation of degree d has size $O(n^d)$ and can be found in $n^{O(d)}$ time using semidefinite programming and this is often a useful heuristic bound for the complexity of an SoS refutation. The actual size complexity of SoS can sometimes be significantly smaller than n^d [21] and it would be surprising if the shortest refutation can be found efficiently. Hence it is in general of interest to understand both the degree and the size needed to refute any given system.

SoS is a very powerful proof system and captures many state of the art algorithms that are based on spectral methods. A classic algorithm captured by SoS is Goemans and Williamson's Max-Cut algorithm [8], but also approximate graph coloring algorithms [15], and algorithms solving constraint satisfaction problems [2, 22] are captured by SoS. On the other hand SoS has real difficulty arguing about integers and in particular parities. Indeed, Grigoriev [10] showed that SoS requires degree $\Omega(n)$ to refute a random *xor* constraint satisfaction problem of the appropriate (constant) density. After this initial lower bound it took a few years to develop good lower bounds methods for SoS, but in recent years there has been a flurry of strong SoS degree lower bounds [18, 4, 16].

In order to rule out that algorithms captured by SoS can solve MCSP efficiently, we need to encode the claim that a given function has a small circuit as a propositional formula. We work with the encoding suggested by Razborov [26], which encodes this claim that the function $f : \{0,1\}^n \to \{0,1\}$ has a circuit of size s by a propositional formula $\text{Circuit}_s(f)$ over $O(s^2 + s \cdot 2^n) = O(s \cdot 2^n)$ variables as follows. We have $\Theta(s^2)$ *structure variables* to encode all possible size s circuits, and for every assignment $\alpha \in \{0,1\}^n$ we then have an additional $\Theta(s)$ *evaluation variables* that simulate the evaluation of the circuit on each input, and constraints forcing the circuit to output the correct value on each input α.

A closely related question to the MCSP problem is the question of how hard it is to actually prove strong circuit lower bounds. For example, are there efficient refutations of the statement $\mathbf{NP} \subseteq \mathbf{P}/\mathsf{poly}$, assuming the statement is false? This question, as proposed by Razborov [26], can also be investigated by studying above formula: consider $\text{Circuit}_{n^{O(1)}}(\mathsf{SAT})$, where SAT is the function that outputs 1 if and only if the input is an encoding of a satisfiable CNF. This is, essentially, a propositional encoding of the claim that SAT has a circuit in $\mathbf{P}/\mathsf{poly}$. Hence proving lower bounds for $\text{Circuit}_{n^{O(1)}}(\mathsf{SAT})$ rules out efficient proofs of $\mathbf{NP} \not\subseteq \mathbf{P}/\mathsf{poly}$ in the proof system under consideration.

Experience suggests that studying such meta-mathematical questions is difficult. This problem is no exception to this rule and, even though the formula has been conjectured to be hard for strong proof systems such as extended Frege, progress has been slow. The only proof systems for which we have unconditional, superpolynomial lower bounds on proofs of the $\text{Circuit}_s(f)$ formula are Resolution [23, 27], small width DNF-Resolution [28] and Polynomial Calculus [26, 28]. The resolution size and Polynomial Calculus degree lower bounds follow from a reduction of the pigeonhole principle to $\text{Circuit}_s(f)$. In fact, this reduction was a main motivation for a long line of work [29, 20, 23, 27] eventually establishing strong resolution lower bounds for the weak pigeonhole principle. The other size lower bounds follow from a general connection between pseudo-random generator lower bounds and MCSP lower bounds as outlined in [1, 28], building on Krajíček's iterability trick [17].

As the pigeonhole principle is easy for the SoS proof system [11], we cannot hope to borrow the hardness from that formula. Neither do we have strong enough pseudorandom generator lower bounds for SoS to employ that connection. In fact, to date, we have no unconditional (degree) lower bounds for any semi-algebraic proof system, that is, proof systems that manipulate polynomial inequalities such as SoS or Cutting Planes. Furthermore it has been stated [24, 25] as an explicit open problem to prove SoS degree lower bounds for the formula $\text{Circuit}_s(f)$.

1.1 Our Results

Our first result gives a lower bound on the degree needed to refute $\text{Circuit}_s(f)$ in SoS. This lower bound is very general and in fact applies to *every* Boolean function $f : \{0,1\}^n \to \{0,1\}$.

▶ **Theorem 1.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For $n \in \mathbb{N}$, all $s \geq n^d$ and any Boolean function $f : \{0,1\}^n \to \{0,1\}$ on n bits, SoS requires degree $\Omega_\varepsilon(s^{1-\varepsilon})$ to refute $\text{Circuit}_s(f)$.*

It is worthwhile to point out that the proof of Theorem 1 is not specific to the SoS proof system. In fact we outline a general reduction that shows that if one has a CSP lower bound of the form of Theorem 13 that only requires good expansion of the underlying constraint-variable incidence graph and the proof system is expressive enough so that one can replace variables by local Boolean functions, then one obtains strong lower bounds for the $\text{Circuit}_s(f)$ formula.

The lower bound of $\Omega_\varepsilon(s^{1-\varepsilon})$ on the degree is essentially tight: if f does not have a circuit of size s then there exists an SoS refutation of this statement in degree $O(s)$.

▶ **Proposition 2.** *Let $s \in N$ and $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function on n bits that requires circuits of size larger than s to be computed. Then there is a degree $O(s)$ SoS refutation of $\text{Circuit}_s(f)$.*

For a proof of Proposition 2 we refer to the full version.

We also prove a result about the minimum size (number of monomials) required for SoS to refute $\text{Circuit}_s(f)$. This result holds for all functions that "almost" have a circuit of size s, in the sense that they have an errorless heuristic circuit (see the survey [7]) of size $s/2$ and extremely small error probability with respect to the uniform distribution. Formally, we let $\mathcal{F}_n(s,t)$ denote the class of Boolean functions that consists of all functions $f : \{0,1\}^n \to \{0,1\}$ for which there is a Boolean circuit $C_f : \{0,1\}^n \to \{0,1,\perp\}$ of size at most s such that

1. if $C_f(\alpha) \neq \perp$, then $C_f(\alpha) = f(\alpha)$, and
2. $C_f(\alpha) = \perp$ on at most t inputs.

In other words the circuit C_f computes f correctly on all except t inputs. Note that technically the output of the circuit C_f is two bits with the first one indicating whether the output is $\perp$ or the value of the second bit. We believe that above presentation is more intuitive and hope that the slight abuse of notation causes no confusion. With the class of functions $\mathcal{F}_n(s,t)$ at hand we can state our main SoS size lower bound.

▶ **Theorem 3.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. Let $n \in N$ and $s \in \mathbb{N}$ such that $s \geq n^d$. If $t \geq s$ and $f \in \mathcal{F}_n(s/2, t)$, then it holds that SoS requires size $\exp\left(\Omega_\varepsilon(s^{2-\varepsilon}/t)\right)$ to refute $\mathrm{Circuit}_s(f)$.*

This yields non-trivial size lower bounds for t as large as $s^{2-\varepsilon}/\omega(1)$. Furthermore, note that once $t \gg s \log s$ there are functions that require such large circuits. For example setting $s = 2^{n^{0.99}}$ and $t = s^{1.5}$, the theorem shows that there are functions f that do not have circuits of size s, but SoS requires size $2^{2^{\Omega(n^{0.99})}}$ to prove this.

It is natural to wonder whether SoS fares better in the monotone setting. In other words, whether SoS can refute the claim that a complex monotone function has a small monotone circuit. The following two theorems show that this is not the case for the set $\mathcal{M}_n(\ell)$ of monotone ℓ-slice functions. Recall that $\mathcal{M}_n(\ell)$ consist of all Boolean functions f on n bits such that $f(\alpha) = 0$ for all α with Hamming weight less than ℓ, and $f(\alpha) = 1$ for all α with Hamming weight greater than ℓ (note that any such f is monotone).

We define a variant $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$ of the $\mathrm{Circuit}_s(f)$ formula, which instead encodes the claim that f has a monotone circuit of size s, and prove the following theorem.

▶ **Theorem 4.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For all $n, \ell \in \mathbb{N}$, all $s \geq n^d$ and any monotone slice function $f \in \mathcal{M}_n(\ell)$ SoS requires degree $\Omega_\varepsilon(s^{1-\varepsilon})$ to refute $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$.*

As in the non-monotone case, we can also obtain size lower bounds for the monotone-MCSP. Akin to the general size lower bound we consider monotone Boolean slice functions that have good monotone errorless heuristic circuits. Let $\mathcal{M}_n(\ell, s, t) \subseteq \mathcal{M}_n(\ell)$ be the class of monotone Boolean ℓ-slice functions $f : \{0,1\}^n \to \{0,1\}$ for which there is a (not necessarily monotone) Boolean circuit $C_f^{\mathsf{mon}} : \{0,1\}^n \to \{0,1,\perp\}$ of size s such that

1. for all ℓ-slice inputs $\alpha \in \binom{[n]}{\ell}$ it holds that if $C_f^{\mathsf{mon}}(\alpha) \neq \perp$, then $C_f^{\mathsf{mon}}(\alpha) = f(\alpha)$, and
2. $C_f^{\mathsf{mon}}(\alpha) = \perp$ on at most t inputs $\alpha \in \binom{[n]}{\ell}$.

▶ **Theorem 5.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For $n, \ell \in \mathbb{N}$, all $s \geq n^d$ and $t \geq s$ and monotone function $f \in \mathcal{M}_n(\ell, s/10, t)$ SoS requires size $\exp\left(\Omega_\varepsilon(s^{2-\varepsilon}/t)\right)$ to refute $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$.*

1.2 Overview of Proof Techniques

Degree Lower Bound

The main idea that drives our result is a reduction from an expanding *xor* constraint satisfaction problem to the $\mathrm{Circuit}_s(f)$ formula. The reduction is achieved through a careful restriction of the $\mathrm{Circuit}_s(f)$ formula, such that each input $\alpha \in \{0,1\}^n$ to the circuit specifies an *xor* constraint over some new set of variables Y. These Y variables are a subset of roughly $\Theta(s^{1-\epsilon})$ out of the $\Theta(s^2)$ many structure variables of the $\mathrm{Circuit}_s(f)$ formula. All other structure variables apart from the Y variables are fixed to constant values in this step. This will then result in an XOR-CSP instance with 2^n constraints over the variables Y. All that SoS has to prove is that there is no satisfying assignment to this XOR-CSP instance. By

ensuring that the constraint-variable incidence graph is sufficiently expanding, SoS requires large degree to refute the restricted formula (see Theorem 13). At the same time, we need the constraint graph to be very explicit so that it can be encoded into a small circuit. For this we utilize a construction of unbalanced expanders by Guruswami et al. [12] (see Theorem 8). This reduction then immediately yields Theorem 1.

This lower bound may also be viewed as implementing the general program sketched by Razborov [28] relating pseudorandom generators in proof complexity to the MCSP problem. However, we prefer to describe it as a direct reduction to the MCSP problem.

Size Lower Bound

In order to obtain size lower bounds, we would like to apply the degree-size tradeoff due to Atserias and Hakoniemi [3] to Theorem 1. Unfortunately the formula is over too many variables to be able to conclude a meaningful size lower bound: it is defined over roughly $\Omega(2^n \cdot s)$ variables.

Instead of applying Theorem 1, we restrict our attention to functions with all except the at most t $\perp$-outputs computed by the corresponding errorrless heuristic circuit. If we choose t small enough, then we are able to heavily restrict $\mathrm{Circuit}_s(f)$ and significantly reduce the number of variables to the point where the Atserias-Hakoniemi degree-size tradeoff is applicable.

Monotone Circuits

We prove these theorems by adapting the proofs for the non-monotone setting. The idea is to work over the ℓth slice and disregard all other inputs. The key feature that makes this work is the fact that the monotone circuit complexity of a slice function is essentially the same as the (ordinary) circuit complexity (see Lemma 10). This lets us convert all subcircuits used in the reduction to small monotone circuits (if we only work on the slice).

The size lower bound goes along the same lines as the proof of Theorem 3.

1.3 Organization

In Section 2, we provide the necessary background material. In Section 3 we set up the general framework for our lower bounds with some preliminary definitions and lemmas. Then in Section 4 we prove the main degree Theorem 1 and size Theorem 3 lower bounds. We prove the monotone lower bounds Theorem 4 and Theorem 5 in Section 5. Finally in Section 6 we give some concluding remarks.

2 Preliminaries

All logarithms are in base 2. For integers $n \geq 1$ we write $[n] = \{1, 2, \ldots, n\}$ and for a set U we denote the power set of U by 2^U. Further, for a set $V \subseteq U$ we let $\overline{V}$ be the complement of V with respect to U, that is, $\overline{V} = U \setminus V$. We write $\binom{[n]}{\ell} \subseteq \{0,1\}^n$ for the set of binary strings with Hamming weight ℓ. For a string $\alpha \in \{0,1\}^n$ we let $|\alpha| = \sum_{i \in [n]} \alpha_i$.

We sometimes want to supress dependencies on constants and write $f(n,\varepsilon) \in O_\varepsilon\big(g(n,\varepsilon)\big)$, respectively $f(n,\varepsilon) \in \Omega_\varepsilon\big(g(n,\varepsilon)\big)$, to mean that there exists a function $c(\varepsilon) > 0$ such that there is an n_0 and for all $n \geq n_0$ it holds that $f(n,\varepsilon) \leq c(\varepsilon) \cdot g(n,\varepsilon)$, respectively $f(n,\varepsilon) \geq c(\varepsilon) \cdot g(n,\varepsilon)$.

▶ **Definition 6.** *A sequence of bipartite graphs* $\{G_n = (U_n, V_n, E_n)\}_{n\in\mathbb{N}}$ *with* $\deg(u) = d$ *for all* $u \in U_n$ *is* explicit *if there is an algorithm that given* (n, u, j)*, where* $n \in \mathbb{N}, u \in U_n$ *and* $j \in [d]$*, computes the* j*th neighbor of vertex* u *in the graph* G_n *in time* $\mathsf{poly}(\log n + \log|U| + \log d)$.

From now on it is understood that whenever we talk about an explicit graph we actually mean to say that there is a sequence of explicit graphs with above properties.

For a graph $G = (V, E)$ and $W \subseteq V$ we denote by

$$N(W) = \{\, u \in V \setminus W \mid (w, u) \in E \text{ for some } w \in W \,\}$$

the set of neighbors of W.

▶ **Definition 7.** *A bipartite graph* $G = (U, V, E)$ *is an* (r, d, c)*-expander if every vertex* $u \in U$ *has degree* $\deg(u) = d$ *and every set* $W \subseteq U$ *of size* $|W| \leq r$ *satisfies* $|N(W)| \geq c \cdot |W|$.

A key ingredient in our proofs is the following result on the existence of strong explicit expanders.

▶ **Theorem 8** ([12])**.** *For all constants* $\gamma > 0$*, every* $M \in \mathbb{N}$*,* $r \leq M$*, and* $\varepsilon > 0$*, there is an* $N \leq d^2 \cdot r^{1+\gamma}$ *and an explicit* $(r, d, (1-\varepsilon)d)$*-expander* $G = (U, V, E)$*, with* $|U| = M$*,* $|V| = N$*, and* $d = O\big(((\log M)(\log r)/\varepsilon)^{1+1/\gamma}\big)$.

For our purposes it is more relevant to compute the neighbor relation $\mathbf{Neigh}(u, v)$ indicating whether $(u, v) \in E$ rather than the neighbor function as in Definition 6, but this is an immediate consequence of being able to compute the neighbor function.

▷ Claim 9. If $G = (U, V, E)$ is explicit then the neighbor relation $\mathbf{Neigh} : U \times V \to \{0, 1\}$ is computable by a circuit of size $d \cdot \big(\mathsf{poly}(\log n + \log|U| + \log d) + 2\log|V| + 1\big)$.

A slice function is a Boolean function f such that there is a $\ell \in [n]$ with $f(\alpha) = 0$ whenever $|\alpha| < \ell$, and $f(\alpha) = 1$ whenever $|\alpha| > \ell$. Note that all slice functions are monotone.

The circuit complexity $\mathcal{C}(f)$ of a Boolean function f is the size of the smallest circuit over the basis $\vee, \wedge$, and $\neg$ (with fan-in 2). Similarly the monotone circuit complexity $\mathcal{C}_{\mathsf{mon}}(f)$ of a monotone Boolean function f is the size of the smallest circuit over the basis $\vee$, and $\wedge$. We have the following useful inequality between these measures.

▶ **Lemma 10** ([6])**.** *If* g *is any slice function on* n *bits, then* $\mathcal{C}_{\mathsf{mon}}(g) \leq 2\,\mathcal{C}(g) + O(n^2 \log n)$.

Finally we also rely on the following simple claim.

▷ Claim 11. Let $p : \mathbb{R}^n \to \mathbb{R}$ be a degree d polynomial such that $p(x) = 0$ for all $x \in \{0, 1\}^n$. Then p can be written as

$$p(x) = \sum_{i\in[n]} q_i(x) \cdot (x_i^2 - x_i)$$

where each term in the sum has degree at most d.

Proof sketch. We take the polynomial p and multilinearize it, using the appropriate polynomial $x_i^2 - x_i$. Eventually we are left with a sum of polynomials of the form $q_i(x) \cdot (x_i^2 - x_i)$ and a multilinear polynomial $\tilde{p}(x)$ which is 0 on all Boolean inputs. As multilinear polynomials are a basis for Boolean functions this implies that $\tilde{p}(x)$ is equal to the 0 polynomial and hence the claim follows. ◁

2.1 Sum of Squares

Let $\mathcal{P} = \{p_1 = 0, \dots, p_m = 0\}$ be a system of polynomial equations over the set of variables $X = \{x_1, \dots, x_n, \bar{x}_1, \dots, \bar{x}_n\}$. Each p_i is called an *axiom*, and throughout the paper we always assume that $\mathcal{P}$ includes all axioms $x_i^2 - x_i$ and $\bar{x}_i^2 - \bar{x}_i$, ensuring that the variables are Boolean, as well as the axioms $1 - x_i - \bar{x}_i$, making sure that the "bar" variables are in fact the negation of the "non-bar" variables.

▶ **Definition 12** (Sum-of-Squares)**.** Sum-of-Squares (SoS) *is a static, semi-algebraic proof system. An SoS proof of $f \geq 0$ from $\mathcal{P}$ is a sequence of polynomials $\pi = (t_1, \dots, t_m; s_1, \dots, s_a)$ such that*

$$\sum_{i \in [m]} t_i p_i + \sum_{i \in [a]} s_i^2 = f \ .$$

The degree *of a proof π is $\mathsf{Deg}(\pi) = \max\{\max_{i\in[m]} \deg(t_i) + \deg(p_i), \max_{i\in[a]} 2\deg(s_i)\}$, an* SoS refutation of $\mathcal{P}$ *is an SoS proof of $-1 \geq 0$ from $\mathcal{P}$, and the* SoS degree to refute $\mathcal{P}$ *is the minimum degree of any SoS refutation of $\mathcal{P}$: if we let π range over all SoS refutations of $\mathcal{P}$, we can write $\mathsf{Deg}(\mathcal{P} \vdash_{SoS} \bot) = \min_\pi \mathsf{Deg}(\pi)$. The* size of an SoS refutation π, $\mathsf{Size}(\pi)$, *is the sum of the number of monomials in each polynomial in π and the* size of refuting $\mathcal{P}$ *is the minimum size over all refutations $\mathsf{Size}(\mathcal{P} \vdash_{SoS} \bot) = \min_\pi \mathsf{Size}(\pi)$.*

Let us recall some well-known results about SoS. Given a bipartite graph $G = (U, V, E)$, and $b \in \{0,1\}^{|U|}$ we denote by $\Phi(G, b)$ the following XOR-CSP instance defined over G: for each $v \in V$ there is a Boolean variable x_v, and for every vertex $u \in U$ there is a constraint $\oplus_{v \in N(u)} x_v = b_u$. We encode this in the obvious way as a system of polynomial equations:

$$\Big\{ \prod_{v \in N(u)} (1 - 2 \cdot x_v) = 1 - 2 \cdot b_u \mid u \in U \Big\} \ ,$$

along with the Boolean axioms and the negation axioms for the x variables. The first theorem we need to recall is the classic lower bounds for XOR-CSPs by Grigoriev.

▶ **Theorem 13** ([10])**.** *For $n \in \mathbb{N}$, all $k = k(n)$ and $r = r(n)$ the following holds. Let $G = (U, V, E)$ be an $(r, k, 2)$-expander with $|V| = n$. Then for every $b \in \{0,1\}^{|U|}$ SoS requires degree $\Omega(r)$ to refute the claim that there is a satisfying assignment to $\Phi(G, b)$.*

We also need to recall the size-degree tradeoff by Atserias and Hakoniemi.

▶ **Theorem 14** ([3])**.** *Let $\mathcal{P}$ be a system of polynomial equations over n Boolean variables and degree at most k. If d is the minimum degree SoS requires to refute $\mathcal{P}$, then the minimum size of an SoS refutation of $\mathcal{P}$ is at least $\exp(\Omega((d-k)^2/n))$.*

2.2 Restrictions

Let $\mathcal{P} = \{p_1 = 0, \dots, p_m = 0\}$ be a system of polynomial equations over the set of Boolean variables $X = \{x_1, \dots, x_n, \bar{x}_1, \dots, \bar{x}_n\}$. For a map $\rho : \{x_1, \dots, x_n\} \to \{0, 1, x_1, \dots, x_n, \bar{x}_1, \dots, \bar{x}_n\}$ denote by $\mathcal{P}\big|_\rho$ the system of polynomial equations $\mathcal{P}$ restricted by ρ, i.e.,

$$\begin{aligned} \mathcal{P}\big|_\rho = \{ & p_1(\rho(x_1), \dots, \rho(x_n)) = 0, \\ & p_2(\rho(x_1), \dots, \rho(x_n)) = 0, \\ & \vdots \\ & p_m(\rho(x_1), \dots, \rho(x_n)) = 0\} \ , \end{aligned}$$

where it is understood that $\rho(\bar{x}_i) = \overline{\rho(x_i)}$, with the convention $\bar{\bar{x}}_i = x_i$, $\bar{0} = 1$ and vice versa. Throughout the paper all our restrictions set the bar variables to the negation of the non-bar variables. As such it makes sense to treat the pair of variables $(x_i, \bar{x}_i)$ as one variable and we say that $\mathcal{P}$ has n *unset* variables.

▶ **Definition 15** (Variable Substitution). *We say that a system of polynomial equations $\mathcal{P}'$ is a* variable substitution of $\mathcal{P}$ *if there is a map $\rho : \{x_1, \ldots, x_n\} \to \{0, 1, x_1, \ldots, x_n, \bar{x}_1, \ldots, \bar{x}_n\}$ such that $\mathcal{P}' = \mathcal{P}\big|_\rho$, where we ignore polynomial equations of the form $0 = 0$.*

The following well-known lemma states that a system of polynomial equations $\mathcal{P}$ is at least as hard as any of its variable substitutions.

▶ **Lemma 16.** *Let $\mathcal{P}, \mathcal{P}'$ be systems of polynomial equations such that $\mathcal{P}'$ is a variable substitution of $\mathcal{P}$. Then,*
1. $\mathsf{Deg}(\mathcal{P} \vdash_{SoS} \bot) \geq \mathsf{Deg}(\mathcal{P}' \vdash_{SoS} \bot)$*, and*
2. $\mathsf{Size}(\mathcal{P} \vdash_{SoS} \bot) \geq \mathsf{Size}(\mathcal{P}' \vdash_{SoS} \bot)$.

The lemma is easy to verify by considering an SoS refutation of $\mathcal{P}$ and hitting it with the appropriate variable substitution. The restricted proof is now a refutation of $\mathcal{P}'$ and it can be seen that the degree/size of the restricted refutation is at most the degree/size of the original refutation.

We also consider more general substitutions.

▶ **Definition 17** (Polynomial Substitution). *Functions $\rho : \{x_1, \ldots, x_n\} \to \mathbb{R}[x]_{\leq k}$ that map variables to polynomials of degree at most k are called* polynomial substitutions.

For polynomial substitutions we have the following well-known lemma.

▶ **Lemma 18.** *Let $\mathcal{P}$ be a system of polynomial equations and let ρ be a polynomial substitution mapping variables to polynomials of degree at most k. Then,* $\mathsf{Deg}(\mathcal{P} \vdash_{SoS} \bot) \geq \mathsf{Deg}(\mathcal{P}\big|_\rho \vdash_{SoS} \bot)/k$.

This lemma can again be verified by considering a refutation of $\mathcal{P}$. Substitute each variable x_i in the proof by $\rho(x_i)$. This results in a refutation of $\mathcal{P}\big|_\rho$, whose degree is at most a factor k larger than the degree of the refutation of $\mathcal{P}$.

2.3 The Circuit Size Formula

The formula $\mathrm{Circuit}_s(f)$ encodes the claim that the function f, given as a truthtable $f \in \{0,1\}^{2^n}$, can be computed by a circuit of size s over n Boolean inputs $x_1, \ldots, x_n$. The encoding is not essential but for concreteness let us fix one encoding of this claim. We deviate from the encoding used by Razborov [26, 27] and do not present the formula as a propositional formula but rather as a system of polynomial equations. In order to encode below constraints as a constant width CNF formula, as done by Razborov, one needs to introduce extension variables. Despite this difference it is not difficult to see that our lower bound also works against the CNF encoding (see the full version of this paper for more details).

We also need to define the monotone version of $\mathrm{Circuit}_s(f)$ denoted by $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$. The later is a restriction of the former with the $\mathsf{IsNeg}(v)$ (see below) variable, for all $v \in [s]$, set to 0. This forces the circuit to only contain $\wedge$ and $\vee$ gates, i.e., the circuit is monotone.

All variables introduced in the following are Boolean variables and we implicitly add the Boolean axiom $y(1-y) = 0$ for each variable y and further implicitly introduce the "bar variable" $\bar{y}$ along with the negation axiom $y = 1 - \bar{y}$ (and the corresponding Boolean axiom) ensuring that $\bar{y}$ is always the negation of y.

Let us first describe the *structure variables* which are used to describe the circuit that supposedly computes the function f.

We view the s gates as being indexed from 1 to s in topological order with gate s being the output. For each gate $v \in [s]$ there are three variables $\mathsf{IsNeg}(v), \mathsf{IsOr}(v), \mathsf{IsAnd}(v)$ indicating the operation computed at v. Similarly for a gate $v \in [s]$ and a wire $a \in \{1,2\}$ we have variables $\mathsf{IsFromConst}(v,a), \mathsf{IsFromInput}(v,a), \mathsf{IsFromGate}(v,a)$ indicating whether the input wire a of v is connected to a constant, a variable or a gate.

Further, we have the variables $\mathsf{ConstantValue}(v,a)$, $\mathsf{IsInput}(v,a,i)$ and $\mathsf{IsGate}(v,a,u)$, for $a \in \{1,2\}$, $i \in [n]$ and $u < v$, specifying the constant value, input x_i or gate u, the input wire a of v is connected to (assuming a is connected to the corresponding kind).

The second set of variables are the *evaluation variables*, which describe what value is computed at each v on input $\alpha = \alpha_1, \ldots, \alpha_n$ (i.e., we have $x_i = \alpha_i$).

For each gate $v \in [s]$ and assignment $\alpha \in \{0,1\}^n$ we have a Boolean variable $\mathsf{Out}_\alpha(v)$ indicating the value computed at gate v on input α. The Boolean variable $\mathsf{In}_\alpha(v,a)$ indicates the value brought to the vertex $v \in [s]$ on wire $a \in \{1,2\}$ on input α.

Note that there is a total of $3s + 6s + 2s + 2sn + 2\binom{s}{2} = \Theta(s^2 + sn)$ structure variables, and a total of $3s2^n$ evaluation variables, for a total of $\Theta(s^2 + s2^n)$ variables in $\mathrm{Circuit}_s(f)$.

The formula consists of the following axioms. For the sake of readability we omit some universal quantifiers: the variable $a \in \{1,2\}$ in Axioms 1 and 3–9 as well as the variable $\alpha \in \{0,1\}^n$ in Axioms 7–13 are implicitly universally quantified.

Let us first describe the axioms on the structure of the circuit. In the following section we refer to this set of axioms as the *structure axioms*. The first axioms ensure that every wire is connected to a single kind

$$\mathsf{IsFromConst}(v,a) + \mathsf{IsFromInput}(v,a) + \mathsf{IsFromGate}(v,a) = 1 \quad \forall\, v \in [s] \ , \tag{1}$$

and similarly the next axioms make sure that each gate is of precisely one kind

$$\mathsf{IsNeg}(v) + \mathsf{IsOr}(v) + \mathsf{IsAnd}(v) = 1 \quad \forall\, v \in [s] \ . \tag{2}$$

The final structure axioms ensure that the variables, which indicate to what input or gate a fixed wire is connected to, always sum to one (except for gate 1 which cannot have any inputs from other gates)

$$\sum_{i=1}^{n} \mathsf{IsInput}(v,a,i) = 1 \quad \forall v \in [s], \text{ and} \tag{3}$$

$$\sum_{u=1}^{v-1} \mathsf{IsGate}(v,a,u) = 1 \quad \forall v \in [s] \setminus \{1\} \ . \tag{4}$$

We further strengthen our encoding by adding the axioms

$$\mathsf{IsInput}(v,a,i)\,\mathsf{IsInput}(v,a,j) = 0 \quad \forall v \in [s],\ i < j \in [n], \text{ and} \tag{5}$$

$$\mathsf{IsGate}(v,a,u)\,\mathsf{IsGate}(v,a,u') = 0 \quad \forall u < u' < v \in [s] \ . \tag{6}$$

Note that Axioms 5 and 6 are implied by Axioms 3 and 4. We add these axioms in order to argue that a short refutation of the CNF encoding of this principle leads to a short refutation of the present encoding.

The second group of axioms are the *evaluation axioms* and they ensure that the evaluation variables indeed compute the intended values. We start by making sure that the wires carry the value intended by the structure axioms. If a wire is connected to a constant, then the evaluation variable associated with that wire should always be equal to the constant

$$\mathsf{IsFromConst}(v,a)\cdot\big(\mathsf{In}_\alpha(v,a)-\mathsf{ConstantValue}(v,a)\big)=0\ , \tag{7}$$

and similarly in case if a wire is connected to an input or a gate

$$\mathsf{IsFromInput}(v,a)\cdot\mathsf{IsInput}(v,a,i)\cdot\big(\mathsf{In}_\alpha(v,a)-\alpha_i\big)=0\ , \tag{8}$$

$$\mathsf{IsFromGate}(v,a)\cdot\mathsf{IsGate}(v,a,u)\cdot\big(\mathsf{In}_\alpha(v,a)-\mathsf{Out}_\alpha(u)\big)=0\ . \tag{9}$$

The final set of evaluation axioms makes sure that the output evaluation variable of a gate is correctly related to the input evaluation variables:

$$\mathsf{IsNeg}(v)\cdot\mathsf{Out}_\alpha(v)=\mathsf{IsNeg}(v)\cdot\overline{\mathsf{In}_\alpha(v,1)}\ , \tag{10}$$

$$\mathsf{IsOr}(v)\cdot\mathsf{Out}_\alpha(v)=\mathsf{IsOr}(v)\cdot\big(1-\overline{\mathsf{In}_\alpha(v,1)}\cdot\overline{\mathsf{In}_\alpha(v,2)}\big)\ , \tag{11}$$

$$\mathsf{IsAnd}(v)\cdot\mathsf{Out}_\alpha(v)=\mathsf{IsAnd}(v)\cdot\mathsf{In}_\alpha(v,1)\cdot\mathsf{In}_\alpha(v,2)\ . \tag{12}$$

Last but not least we have the axioms that ensure that the circuit outputs the function specified by the truthtable

$$\mathsf{Out}_\alpha(s)=f(\alpha)\ . \tag{13}$$

3 On Circuits and Restrictions

Let $G=(U,V,E)$ be a bipartite graph with $U=\{0,1\}^n$ and $V=[m]$. As in the XOR-CSP setup (Section 2.1) we think of vertices in U as constraints and vertices in V as variables. More specifically, we think of each vertex $\alpha\in U$ as an *xor* constraint over the variables in the neighborhood $\oplus_{i\in N(\alpha)}v_i=b_\alpha$, for a constraint vector $b\in\{0,1\}^U$. Given an assignment $\beta\in\{0,1\}^m$ to the variables V, we let $f_{G,\beta}:U\to\{0,1\}$ be the function defined by $f_{G,\beta}(\alpha)=\oplus_{i\in N(\alpha)}v_i$. In other words, viewing $f_{G,\beta}$ as a vector in $\{0,1\}^U$, it is the unique constraint vector such that the XOR-CSP instance, defined over G, is satisfied by the assignment β. Let us denote the set of all such constraint vectors that give rise to a satisfiable XOR-CSP instance by

$$\mathcal{F}_G=\{f_{G,\beta}\,|\,\beta\in\{0,1\}^m\}\ .$$

In order for SoS to refute an XOR-CSP instance defined over G, it must prove that the given constraint vector is not in the set $\mathcal{F}_G$.

On the other hand in order for SoS to refute the formula $\mathrm{Circuit}_s(f)$ it needs to show that there is no circuit of size at most s computing f. That is, SoS needs to show that f is not in the set

$$\mathcal{C}_\emptyset=\{T:\{0,1\}^n\to\{0,1\}\text{ such that }\mathrm{Circuit}_s(T)\text{ is satisfiable}\}\ .$$

More generally, if we restrict $\mathrm{Circuit}_s(f)$ by a restriction ρ, then the proof system must prove that f is not a member of the family of truthtables

$$\mathcal{C}_\rho=\{T:\{0,1\}^n\to\{0,1\}\text{ such that }\mathrm{Circuit}_s(T)\big|_\rho\text{ is satisfiable}\}\ .$$

In the following we show that there is a well-behaved restriction ρ such that $\mathcal{C}_\rho=\mathcal{F}_G$ for some explicit graphs G. In other words, once we consider the restricted formula $\mathrm{Circuit}_s(f)\big|_\rho$, SoS needs to rule out that f is a valid right hand side of an XOR-CSP instance. But we know that if G is a moderate expander, then low degree SoS cannot determine whether the XOR-CSP instance is satisfiable and hence we obtain our lower bound.

Let us first formalize the properties we require from ρ. We start off by restricting our attention to a certain natural class of variable substitutions. Namely, we do not want that the structure of the circuit depends on evaluation variables.

▶ **Definition 19** (natural variable substitutions). *A variable substitution ρ to the variables of* $\mathrm{Circuit}_s(f)$ *is* natural *if there is* no *structure variable y such that $\rho(y)$ is an evaluation variable.*

In order to motivate the next definition, let us informally describe the natural restriction ρ and explain the properties of ρ we require.

For now we can think of ρ as a restriction to the structure variables (though for the size lower bounds we also need to restrict some of the evaluation variables). Some set of m structure variables remains undetermined. Let us denote these variables by $y_1, \dots, y_m$. We intend to choose ρ such that on a given input $\alpha \in \{0,1\}^n$ to the circuit, it is forced to compute $\oplus_{i \in N(\alpha)} y_i$. In other words, given such a restriction ρ, we are *essentially* left with an XOR-CSP problem over G, with right hand side f. There is however a difference in that the encoding is non-standard: the evaluation variables act like extension variables that correspond to the functions computed at each gate of the circuit. In order to argue that the known degree lower bound for the XOR-CSP problem implies a degree lower bound for the problem at hand, we need to get rid of these extension variables. This can be done if the functions computed at the gates are of low degree in the y variables.

Recall from Section 2.2 that a system of polynomial equations $\mathcal{P}$ has n unset variables if there are n tuples of variables $(x, \bar{x})$ such that at least one variable of each tuple occurs in $\mathcal{P}$ and all variables in these tuples are unset, i.e., they are not fixed to a constant.

▶ **Definition 20** (k-determined). *Let ρ be a variable substitution to the variables of* $\mathrm{Circuit}_s(f)$ *and suppose that ρ leaves m structural variables $Y = \{y_1, \dots, y_m\}$ unset. Then ρ is* k-determined *if for every $v \in [s]$ and $\alpha \in \{0,1\}^n$ there are multilinear polynomials*

$$g^{\mathrm{out}}_{v,\alpha}, g^{\mathrm{in}_1}_{v,\alpha}, g^{\mathrm{in}_2}_{v,\alpha} : \{0,1\}^m \to \{0,1\}$$

depending on at most k variables such that the following holds. For all $T \in \mathcal{C}_\rho$ and all total assignments σ that satisfy $\mathrm{Circuit}_s(T)\big|_\rho$ *it holds that*

$$\mathsf{Out}_\alpha(v)\big|_{\rho\cup\sigma} = g^{\mathrm{out}}_{v,\alpha}(\beta) \;\;,\quad \mathsf{In}_\alpha(v,1)\big|_{\rho\cup\sigma} = g^{\mathrm{in}_1}_{v,\alpha}(\beta) \;\;,\text{ and }\quad \mathsf{In}_\alpha(v,2)\big|_{\rho\cup\sigma} = g^{\mathrm{in}_2}_{v,\alpha}(\beta) \;\;, \tag{14}$$

where $\beta \subseteq \sigma$ is the assignment to Y.

However, Definition 20 is not quite sufficient. For example, there is no guarantee that $\mathcal{C}_\rho$ is non-empty, i.e., that the restriction ρ describes a valid (partial) circuit. More generally, we need the additional guarantee that there are still many viable circuits that the restricted formula can describe: if there is just a single setting of the Y variables such that all structural axioms are satisfied, then the formula may be refuted in constant degree. Hence we need to ensure that there are many viable assignments to the Y variables that satisfy all structure axioms. This leads us to the following definition.

▶ **Definition 21** (m-independent). *A variable substitution ρ to the variables of the formula* $\mathrm{Circuit}_s(f)$ *is* m-independent *if ρ leaves exactly m structural variables $Y = \{y_1, \dots, y_m\}$ unset, and for every assignment $\beta \in \{0,1\}^Y$ it holds that $|\mathcal{C}_{\rho\cup\beta}| = 1$.*

With these definitions at hand we can state the lemma that drives all our lower bounds.

▶ **Lemma 22.** *Let ρ be a natural m-independent k-determined variable substitution of* $\mathrm{Circuit}_s(f)$*, and let Y and $g^{\mathrm{out}}_{u,\alpha}$ be as in Definition 20. If there is an SoS refutation of* $\mathrm{Circuit}_s(f)\big|_\rho$ *of degree d, then there is a degree $d \cdot k$ SoS refutation of the system of polynomial equations*

$$\{g^{\mathrm{out}}_{s,\alpha}(Y) = f(\alpha) \mid \alpha \in \{0,1\}^n\} \cup \{y_i^2 = y_i \mid i \in [m]\} \;. \tag{15}$$

For proving this lemma, we consider the natural extension of ρ which substitutes all evaluation variables by appropriate degree-k polynomials as indicated by Definition 20.

▶ **Definition 23.** *For a k-determined restriction ρ (with associated polynomials $g^{\text{out}}_{v,\alpha}, g^{\text{in}_1}_{v,\alpha}, g^{\text{in}_2}_{v,\alpha}$) of* $\text{Circuit}_s(f)$*, we denote by $\hat{\rho}$ the polynomial substitution that extends ρ by first substituting any bar variable $\bar{x}$ by $1 - x$ and then substituting all evaluation variables as follows:*

$$\hat{\rho}(\mathsf{Out}_\alpha(v)) = g^{\text{out}}_{v,\alpha}(Y) \ , \qquad \hat{\rho}(\mathsf{In}_\alpha(v,1)) = g^{\text{in}_1}_{v,\alpha}(Y) \ , \text{ and } \qquad \hat{\rho}(\mathsf{In}_\alpha(v,2)) = g^{\text{in}_2}_{v,\alpha}(Y) \ .$$

Note that the formula $\text{Circuit}_s(f)\big|_{\hat{\rho}}$ is defined only over Y. Let us stress that there are no "bar" variables left in the formula. The main observation used to prove Lemma 22 is the following claim, which establishes that the formula (15) is in fact essentially the same as $\text{Circuit}_s(f)\big|_{\hat{\rho}}$.

▷ Claim 24. Let ρ be a natural m-independent k-determined variable substitution of $\text{Circuit}_s(f)$. Then $\text{Circuit}_s(f)\big|_{\hat{\rho}}$ can be written as

$$\text{Circuit}_s(f)\big|_{\hat{\rho}} = \mathcal{P} \cup \mathcal{Q} \ ,$$

where $\mathcal{P}$ is the formula (15) and $\mathcal{Q}$ only consists of axioms that are satisfied for all assignments $\beta \in \{0,1\}^Y$.

Proof. Note that the set of output axioms (13) of $\text{Circuit}_s(f)$ under $\hat{\rho}$ equals the first part of (15), and that the Boolean axioms on the Y variables in $\text{Circuit}_s(f)\big|_{\hat{\rho}}$ are exactly the second part of (15).

The remaining axioms of $\text{Circuit}_s(f)\big|_{\hat{\rho}}$, which are not present in (15), are the Boolean axioms on the variables outside Y, the negation axioms, as well as Axioms 1–12.

The Boolean axioms may turn into polynomials of degree at most $2k$. Because the polynomials we substitute the variables with are Boolean valued, we see that these substituted axioms are satisfied for all assignments $\beta \in \{0,1\}^Y$ and we can thus put them into the set $\mathcal{Q}$.

The negation axioms all become "$0 = 0$" under $\hat{\rho}$ since $\hat{\rho}(\bar{x}) = 1 - \hat{\rho}(x)$.

Finally we need to argue that the Axioms 1–12 are also of the form $p(Y) = 0$ for a polynomial p which is identically 0 on all of $\{0,1\}^m$. This in turn follows immediately from the assumption that ρ is m-independent: for every $\beta \in \{0,1\}^Y$, there exists some T such that the complete assignment $\hat{\rho}(\beta) \cup \beta$ satisfies $\text{Circuit}_s(T)$. But since none of the remaining Axioms 1–12 depends on T, they must then all be satisfied for every $\beta \in \{0,1\}^Y$. ◁

Using this claim we can easily prove Lemma 22.

Proof of Lemma 22. Suppose $\text{Circuit}_s(f)\big|_{\rho}$ has a refutation in degree d. By Lemma 18, there then exists a degree $d \cdot k$ refutation of $\text{Circuit}_s(f)\big|_{\hat{\rho}}$.

By Claim 24, this new refutation is *almost* a refutation of (15), except that $\text{Circuit}_s(f)\big|_{\hat{\rho}}$ has an additional set $\mathcal{Q}$ of axioms that the refutation may use. However, each of these additional axioms is of the form $p(Y) = 0$ for a polynomial which is identically 0 on the entire Boolean cube. By Claim 11, such an axiom can be rewritten as a linear combination of the Boolean axioms. Since the Boolean axioms are present in (15), this yields a refutation of that formula in degree $d \cdot k$. ◀

4 Lower Bounds for General Circuits

We state the following lemma general enough so that we can apply it for the degree as well as the size lower bound. As explained previously, for the size lower bounds we rely on functions that almost have circuits of size s. Recall that we consider the class of functions $\mathcal{F}_n(s,t)$ that consists of all Boolean functions $f : \{0,1\}^n \to \{0,1\}$ for which there is a Boolean circuit $C_f : \{0,1\}^n \to \{0,1,\perp\}$ of size at most s such that

1. if $C_f(\alpha) \neq \perp$, then $C_f(\alpha) = f(\alpha)$, and
2. $C_f(\alpha) = \perp$ on at most t inputs.

The following lemma establishes the existence of m-independent k-determined variable substitutions that result in XOR-CSP instances over explicit graphs.

▶ **Lemma 25.** *For all $k, m, n, t \in \mathbb{N}$ satisfying $m \leq 2^n$, and any explicit bipartite graph $G = (U, V, E)$ such that $|U| = 2^n$, $|V| = m$ and all $u \in U$ are of degree $\deg(u) \leq k$, the following holds. There is a constant $C > 0$, depending on the explicitness of G, such that for all $s \geq C \cdot m \cdot n^C \cdot k^C$ and any Boolean function $f \in \mathcal{F}_n(s/2, t)$ there is a natural m-independent k-determined variable substitutions ρ for the formula $\mathrm{Circuit}_s(f)$ such that*

$$g^{\mathrm{out}}_{s,\alpha}(Y) = \begin{cases} f(\alpha), & \text{if } C_f(\alpha) \neq \perp, \\ \oplus_{i \in N(\alpha)} y_i, & \text{otherwise} \end{cases}$$

for all $\alpha \in \{0,1\}^n$ and $g^{\mathrm{out}}_{s,\alpha}$ and Y as in Definition 20. Furthermore, the formula $\mathrm{Circuit}_s(f)|_\rho$ is over $O(t \cdot k + m)$ variables.

For the degree lower bound (Theorem 1) we will set $t = 2^n$ and use the trivial C_f which always outputs $\perp$, so the reader who wishes a simplified version of the lemma can focus on this special case.

Proof. We consider the formula $\mathrm{Circuit}_s(f)$ and let the first m gates of the formula be denoted by Y. We restrict the formula such that each gate in Y computes an *or* of two constants. The first wire to the gate is fixed to the constant 0, whereas the second wire is only restricted to carry either the constant 0 or 1. In the end these will be the only structural variables that are not restricted to a constant. In the following we think of the gates Y as Boolean variables; as m additional input bits to our circuit.

Further, we restrict another part of the formula such that one part of the circuit described by the formula computes the circuit C_f. Recall that we pretend that the output of C_f is in $\{0,1,\perp\}$, but it actually outputs two bits C^1_f and C^2_f, where $C^1_f(\alpha) = 1$ if and only if $C^2_f(\alpha) = f(\alpha)$.

Finally we also want to hard code the bipartite graph $G(\{0,1\}^n, Y, E)$ into our circuit. Since G is very large this requires G to be explicit. That is, we require small circuits $\mathsf{Sel}_1, \ldots, \mathsf{Sel}_m$, where given any $\alpha \in \{0,1\}^n$, $\mathsf{Sel}_i(\alpha)$ is 1 if and only if the vertex $y_i \in Y$ is a neighbor of the vertex α. By Claim 9 these circuits Sel_i are each of size

$$k \cdot (\mathsf{poly}(n + \log k) + 2 \log m + 1) \leq \mathsf{poly}(n, k) \ .$$

The restriction ρ restricts some structural variables such that a part of the circuit computes $\mathsf{Sel}_1, \ldots, \mathsf{Sel}_m$. We connect each output of the Sel_i circuit by an *and* gate to the negation of C^1_f. Denote the resulting circuits by $\mathsf{Sel}'_1, \ldots, \mathsf{Sel}'_m$. Observe that the circuits Sel'_i output 0 whenever $C^2_f(\alpha) = f(\alpha)$ and otherwise output Sel_i. We think of these circuits as "selector circuits" which indicate whether on input $\alpha \in \{0,1\}^n$ (to the original variables $x_1, \ldots, x_n$ over which the circuit is defined) the variable $y_i \in Y$ appears in the constraint for α.

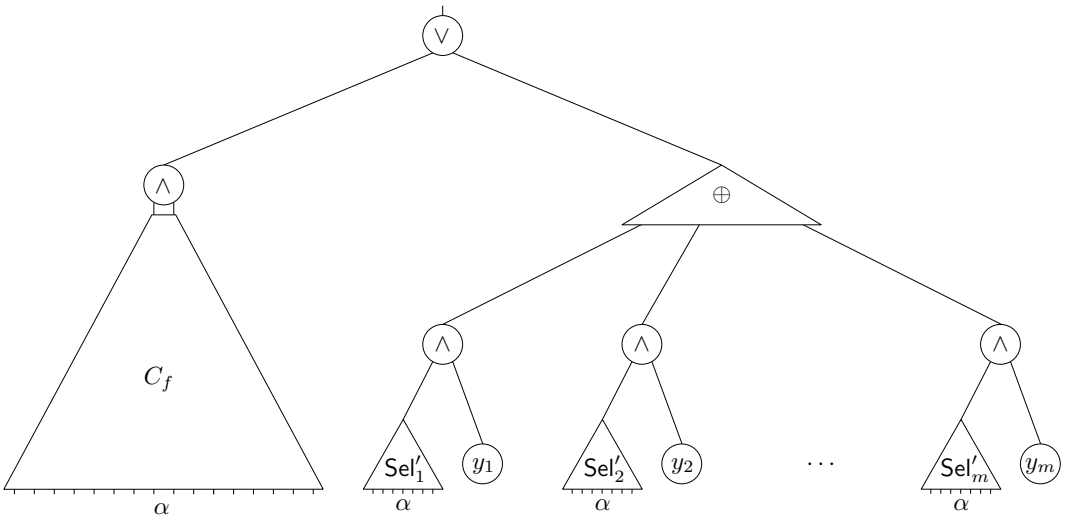


Figure 1 A schematic depiction of the formula after hitting it with the described restriction.

The output of these selector circuits Sel'_i is connected to the gate y_i by an *and* gate. All these m *and* gates are in turn connected to a circuit computing the *xor* of these gates. Finally, to ensure that the circuit computes $f(\alpha)$ on inputs α such that $C_f(\alpha) \neq \bot$, we connect C^1_f with C^2_f by an *and* gate which is then connceted by a *or* gate to the output of the *xor* circuit. This completes the description of the restriction on the structure variables. A depiction of the resulting circuit can be found in Figure 1.

Note that this implements the intended semantics: for each input $\alpha \in \{0,1\}^n$ the selector circuits output 1 on some variables y_i which are then *xor*'ed, and the restricted circuit outputs

$$\bigoplus_{i \in N(\alpha)} y_i \ , \tag{16}$$

unless $C_f(\alpha) \neq \bot$, in which case the output of the circuit is $f(\alpha)$ and all selector circuits output 0. We require that s is larger than the size of the described circuit which is of size $O\big(m \cdot \mathsf{poly}(n,k)\big) + s/2$.

We have the intended semantics of the circuit and need to ensure the furthermore property: that the restricted formula is over few variables. First, since the selector circuits Sel'_i are fixed, all evaluation variables for these subcircuits can be fixed to constants. The same holds for the circuit C_f. Similarly, since the y_i gate always carries the value of the y_i variable, all $2^n \cdot m$ wire variables corresponding to the Y variables can be substituted by the corresponding y_i variable and are thus restricted away.

After these restrictions the only evaluation variables left are those for the evaluation of the $\oplus$ circuit. For α such that $C_f(\alpha) \neq \bot$, the selector circuits are hard-wired to 0 and in particular the inputs to the $\oplus$ circuit is hard-wired to 0, meaning that these evalation variables can be restricted away.

There remains then only the $O(t \cdot m)$ evaluation variables corresponding to the evaluation of the $\oplus$ circuit for inputs α such that $C_f(\alpha) = \bot$. Let us, without loss of generality, use an *xor*-circuit which iteratively *xor*s each variable. Concretely, let it have subcircuits χ_i where $\chi_1 = \mathsf{Sel}'_1 \wedge y_1$ and $\chi_i = \chi_{i-1} \oplus (\mathsf{Sel}'_i \wedge y_i)$ for $i > 1$, and χ_m is the overall output of the $\oplus$ circuit.

The only observation required is that if the circuit $\mathsf{Sel}'_i(\alpha) = 0$, then χ_i gets a 0 as input from index i, independent of the value of y_i. Hence the output wire variable of the circuit χ_i indexed by the input α can be substituted by the output of the circuit χ_{i-1}. Hence for each α such that $C_f(\alpha) = \perp$, we can reduce the number of free wire variables indexed by α to $O(k)$, as each $\oplus$-constraint is over at most k variables. As C_f outputs $\perp$ on at most t inputs, we end up with a restriction leaving only a total of $O(t \cdot k + m)$ remaining variables in the restricted formula.

This completes the description of the restriction ρ. The only part that remains is to verify that ρ is natural, k-determined, and m-independent. That ρ is natural is immediate – it does not substitute any structural variable by an evaluation variable. For k-determinedness, note that for a fixed input α at most k selector circuits output 1, and thus for every gate u the value of $\mathsf{Out}_\alpha(u)$ as a function of Y can be computed by a function over those k variables. Finally, each assignment to the remaining structure variables Y gives a valid circuit and thus ρ is m-independent. ◀

We are ready to prove the degree lower bound, restated here for convenience.

▶ **Theorem 1.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For $n \in \mathbb{N}$, all $s \geq n^d$ and any Boolean function $f : \{0,1\}^n \to \{0,1\}$ on n bits, SoS requires degree $\Omega_\varepsilon(s^{1-\varepsilon})$ to refute* $\mathrm{Circuit}_s(f)$.

Proof. Let $G = (U, V, E)$ be an explicit bipartite graph as in Theorem 8, with $U = \{0,1\}^n$, $k = O_\gamma\big((n \log r)^{1+1/\gamma}\big)$, and $|V| \leq k^2 r^{1+\gamma}$ for parameters $\gamma > 0$ and $r \leq 2^n$ to be fixed later. Apply Lemma 25 with $t = 2^n$ along with $C_f = \perp$ to obtain, for $s \geq m \cdot \mathsf{poly}(n, k)$, a natural m-independent k-determined variable substitution ρ for $\mathrm{Circuit}_s(f)$ such that $g^{\mathrm{out}}_{s,\alpha}(Y) = \oplus_{i \in N(\alpha)} y_i$. In words, the circuit of the restricted formula on input α computes an *xor* of the neighborhood of the vertex α of G.

Apply Lemma 22 to ρ to conclude that if there is an SoS refutation of $\mathrm{Circuit}_s(f)\big|_\rho$ of degree d, then there is a degree $d \cdot k$ SoS refutation of the system of polynomial equations computing

$$\mathcal{P}_G = \Big\{ \bigoplus_{i \in N(\alpha)} y_i = f(\alpha) : \alpha \in \{0,1\}^n \Big\} \cup \{y_i^2 = y_i \mid i \in [m]\} \ .$$

As the graph G is a strong expander, we can apply Theorem 13 to get an SoS degree lower bound of $\Omega(r)$ for the XOR-CSP instance $\mathcal{P}_G$ defined over G, which in turn gives us an $\Omega(r/k)$ degree lower bound for the $\mathrm{Circuit}_s(f)\big|_\rho$ formula and hence also for the unrestricted formula.

Let us fix the parameters. We want to choose r as large as possible. However, the larger we choose r, the larger m may become, since Theorem 8 only guarantees that $m \leq k^2 r^{1+\gamma}$. Let us analyze how large r can be chosen in terms of n and s.

Note that $k = \mathsf{poly}_\gamma(n)$, where we use that $r \leq 2^n$, and we write $\mathsf{poly}_\gamma(n)$ to denote some polynomial in n whose degree and coefficients may depend on γ. Hence we may choose

$$m = \frac{s}{\mathsf{poly}_\gamma(n)} \ , \tag{17}$$

according to the requirement on s in Lemma 25. From the guarantees of Theorem 8 we know that $r \geq (m/k^2)^{1/(1+\gamma)}$. Substituting m according to the previous equation we get that

$$r \geq \left(\frac{s}{k^2 \mathsf{poly}_\gamma(n)}\right)^{\frac{1}{1+\gamma}} = \frac{s^{1/(1+\gamma)}}{\mathsf{poly}_\gamma(n)} \ . \tag{18}$$

Hence if we choose γ small enough so that $\frac{1}{1+\gamma} > 1 - \varepsilon/2$ and then require s to be large enough such that the final $\mathsf{poly}_\gamma(n)$ is at most $s^{\varepsilon/2}$, we obtain the claimed lower bound. ◀

In the following we prove the claimed size lower bound.

▶ **Theorem 3.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. Let $n \in N$ and $s \in \mathbb{N}$ such that $s \geq n^d$. If $t \geq s$ and $f \in \mathcal{F}_n(s/2, t)$, then it holds that SoS requires size $\exp\left(\Omega_\varepsilon(s^{2-\varepsilon}/t)\right)$ to refute $\mathrm{Circuit}_s(f)$.*

Proof. Apply Lemma 25 with the graphs from Theorem 8 as in the proof of Theorem 1. We get a natural m-independent k-determined variable substitution ρ and the formula $\mathrm{Circuit}_s(f)\big|_\rho$ over $O(t \cdot k + m)$ variables. To this formula we then apply Lemma 22 to obtain a degree lower bound of $\Omega(r/k)$, akin to the proof of Theorem 1. By setting the parameters as in the aforementioned proof we get the same degree lower bound of $\Omega_\varepsilon(s^{1-\varepsilon/3})$ for the formula $\mathrm{Circuit}_s(f)\big|_\rho$. As this formula is over few variables we can apply Theorem 14 to obtain an SoS size lower bound of $\exp\left(\Omega_\varepsilon\left((s^{1-\varepsilon/3} - 3k)^2/(t \cdot k + m)\right)\right)$ for the restricted formula. As variable substitutions may only decrease the size of a refutation, the same lower bound also holds for the unrestricted formula. We obtain the desired lower bound by choosing s large enough such that $s^{\varepsilon/3} \geq k = \mathsf{poly}_\varepsilon(n)$ and by recalling that $t \geq s \geq m$. ◀

5 Lower Bounds for Monotone Circuits

Recall that $\mathcal{M}_n(\ell)$ denotes all Boolean monotone ℓ-slice functions on n bits: all Boolean functions $f : \{0,1\}^n \to \{0,1\}$ that output 0 on all inputs of Hamming weight less than ℓ and 1 on all inputs of Hamming weight larger than ℓ. There is no restriction on the output for inputs of Hamming weight ℓ and we have $|\mathcal{M}_n(\ell)| = 2^{\binom{n}{\ell}}$. Further, recall that $\mathcal{M}_n(\ell, s, t) \subseteq \mathcal{M}_n(\ell)$ is the class of monotone Boolean ℓ-slice functions $f : \{0,1\}^n \to \{0,1\}$ for which there is a (not necessarily monotone) Boolean circuit $C_f^{\mathsf{mon}} : \{0,1\}^n \to \{0,1,\bot\}$ of size s such that

1. for all ℓ-slice inputs $\alpha \in \binom{[n]}{\ell}$ it holds that if $C_f^{\mathsf{mon}}(\alpha) \neq \bot$, then $C_f^{\mathsf{mon}}(\alpha) = f(\alpha)$, and
2. $C_f^{\mathsf{mon}}(\alpha) = \bot$ on at most t inputs $\alpha \in \binom{[n]}{\ell}$.

It is very convenient to work with slice functions as we have a handle on their monotone circuit complexity: by Lemma 10 their monotone circuit size is the same as their ordinary circuit size up to a polynomial size increase. Hence we do not need to worry whether the functions needed for the reduction have small monotone circuits, as long as we are working on a slice only.

The proof of the monotone lower bound is an adaption of the argument used to prove Lemma 25. The idea is to work over the ℓth slice and disregard all other inputs. By Lemma 10 we can implement our selector circuits by small monotone circuits. We then also need to take care of the negations in the $\oplus$-circuit. We push the negations down until they either hit a gate in Y or a selector circuit. We create a set $\overline{Y}$ gates, which we can think of as the negation of the gates in Y and also create negated selector circuits (on the ℓth slice). By doing so we can now get rid of the last negations by appropriately connecting the appropriate circuits. The following corollary of Lemma 10 will be useful to us.

▷ Claim 26. Let C be a Boolean circuit on n input bits of size s. Then, for $\ell \in [n]$, there is a monotone Boolean circuit C^{mon} of size $2s + \mathsf{poly}(n)$ computing the ℓ-slice function that is equal to C on the ℓ-slice.

Proof. Let $\mathcal{T}_{\geq \ell}$ be the threshold function that outputs 1 if and only if the Hamming weight of an input $\alpha \in \{0,1\}^n$ is at least ℓ. Connect the output of C by an *and* gate to a circuit computing $\mathcal{T}_{\geq \ell}$. The output of this circuit is then connected by an *or* gate to the output of a circuit computing $\mathcal{T}_{>\ell}$. Let us denote this new circuit by C'.

The circuit C' clearly outputs 1 whenever the input is of Hamming weight larger than ℓ. Furthermore, on the ℓ-slice it is equal to C because $\mathcal{T}_{\geq \ell}$ outputs 1 while $\mathcal{T}_{>\ell}$ outputs 0. Finally the output is 0 if the Hamming weight is less than ℓ because the output of both threshold functions is 0.

Clearly the size of the circuits computing the threshold functions is $\mathsf{poly}(n)$. We apply Lemma 10 to conclude that there is a monotone circuit C^{mon} computing the same function as C' of size $2s + \mathsf{poly}(n)$. ◁

Before stating the following lemma we need to adapt some terminology to the monotone setting. Observe that $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$ is a restriction of $\mathrm{Circuit}_s(f)$. Let τ be such that $\mathrm{Circuit}_s(f)\big|_\tau = \mathrm{Circuit}_s^{\mathsf{mon}}(f)$. This allows us to naturally extend k-determined restrictions to the monotone setting: a restriction ρ is a k-determined restriction for $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$ if the restriction $\rho\tau$ is a k-determined restriction for $\mathrm{Circuit}_s(f)$. Similarly we can extend m-independence to the monotone setting. This will later allow us to use Lemma 22 even though we are working with the monotone formula.

▶ **Lemma 27.** *For all $k, \ell, m, n, t \in \mathbb{N}$ satisfying $m \leq 2^n$, and any explicit bipartite graph $G = (U, V, E)$ such that $|U| = 2^n$, $|V| = m$ and all $u \in U$ are of degree $\deg(u) \leq k$, the following holds. There is a constant $C > 0$, depending on the explicitness of G, such that for all $s \geq C \cdot m \cdot n^C \cdot k^C$ and any $f \in \mathcal{M}_n(\ell, s/10, t)$ there is a natural m-independent k-determined variable substitution ρ for the formula $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$ such that*

$$g_{s,\alpha}^{\mathrm{out}}(Y) = \begin{cases} 1, & \text{if } |\alpha| > \ell, \\ 0, & \text{if } |\alpha| < \ell, \\ f(\alpha), & \text{if } |\alpha| = \ell \text{ and } C_f^{\mathsf{mon}}(\alpha) \neq \bot, \\ \oplus_{i \in N(\alpha)} y_i, & \text{otherwise,} \end{cases}$$

for $g_{s,\alpha}^{\mathrm{out}}$ and Y as in Definition 20.

Furthermore, the formula $\mathrm{Circuit}_s^{\mathsf{mon}}(f)\big|_\rho$ is over $O(t \cdot k + m)$ variables.

Proof. This proof is an adaptation of the argument of the proof Lemma 25. Let us describe the natural m-independent k-determined restriction ρ for the formula $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$.

As in the proof of Lemma 25 we have gates that act as Boolean variables. But instead of having a single set Y of variables we now have two sets Y and $\overline{Y}$, each of size m. We think of the variables in $\overline{Y}$ as the negations of the variables in Y and ensure this by applying the appropriate variable substitution for all $\alpha \in \{0,1\}^n$ and $i \in [m]$.

According to Claim 26 we may assume that the circuit C_f^{mon} computes a monotone ℓ-slice function in both outputs $C_{f,1}^{\mathsf{mon}}, C_{f,2}^{\mathsf{mon}}$ for a mild increase in size; $|C_f^{\mathsf{mon}}| \leq s/5 + \mathsf{poly}(n) \leq s/4$ for s large enough. Recall that the first output of C_f^{mon} indicates whether the second output bit is equal to f on the ℓ-slice. Let $\overline{C}_{f,1}^{\mathsf{mon}}$ be the negation of $C_{f,1}^{\mathsf{mon}}$ on the ℓ-slice. In other words, $\overline{C}_{f,1}^{\mathsf{mon}}(\alpha) = \neg C_{f,1}^{\mathsf{mon}}(\alpha)$ if α has Hamming weight ℓ, and $\overline{C}_{f,1}^{\mathsf{mon}}(\alpha) = C_{f,1}^{\mathsf{mon}}(\alpha)$ otherwise.

The monotone circuit C_f^{mon} is of size at most $s/4$ and hence according to Lemma 10 there is a monotone circuit of size $s/2 + \mathsf{poly}(n) \leq 5s/8$ computing $\overline{C}_{f,1}^{\mathsf{mon}}(\alpha)$.

We restrict the formula such that a part of the circuit is equivalent to C_f^{mon} and another part is equal to $\overline{C}_{f,1}^{\mathsf{mon}}$. Note that the size of these two circuits is at most $7s/8$ by above discussion.

Recall that because $G(\{0,1\}^n, Y, E)$ is explicit, there are circuits $\mathsf{Sel}_1, \mathsf{Sel}_2, \ldots, \mathsf{Sel}_m$, each of size $\mathsf{poly}(n, k)$, where each Sel_i computes, given an input $\alpha \in \{0,1\}^n$, whether the vertex $y_i \in Y$ is a neighbor of the vertex α. Let $\overline{\mathsf{Sel}}_i = \neg\, \mathsf{Sel}_i$ and denote by $\mathsf{Sel}_i^{\mathsf{mon}}$ (respectively $\overline{\mathsf{Sel}}_i^{\mathsf{mon}}$) the circuit obtained by applying Claim 26 to Sel_i (to $\overline{\mathsf{Sel}}_i$ respectively). By the guarantees of Claim 26 all these $2m$ circuits are of size $\mathsf{poly}(n, k)$.

We restrict the formula such that a part of the circuit computes the functions

$$\mathsf{Sel}_1^{\mathsf{mon}}, \dots, \mathsf{Sel}_m^{\mathsf{mon}}, \overline{\mathsf{Sel}}_1^{\mathsf{mon}}, \dots, \overline{\mathsf{Sel}}_m^{\mathsf{mon}} \quad . \tag{19}$$

From these ℓ-slice selector circuits we can then define selector circuits that take C_f^{mon} into account. Namely, we connect $\mathsf{Sel}_i^{\mathsf{mon}}$ by an *and* gate to the output of $\overline{C}_{f,1}^{\mathsf{mon}}$ to obtain the circuit $\mathsf{Sel}_i'^{\mathsf{mon}}$ and similarly connect $\overline{\mathsf{Sel}}_i^{\mathsf{mon}}$ by an *or* gate to $C_{f,1}^{\mathsf{mon}}$ to obtain the circuit $\overline{\mathsf{Sel}}_i'^{\mathsf{mon}}$.

Finally, we also put each variable y_i and $\bar{y}_i$ onto the slice by the same construction used in the proof of Claim 26: connect the variable y_i (respectively $\bar{y}_i$) by an *and* to the threshold circuit $\mathcal{T}_{\geq \ell}$ and connect this circuit in turn by an *or* gate to a $\mathcal{T}_{>\ell}$ threshold circuit to obtain y_i^{mon} (respectively $\bar{y}_i^{\mathsf{mon}}$). It is well-known [30, 5, 9] that threshold circuits have montone circuits of size $\mathsf{poly}(n)$ and we can thus restrict the formula such that a part of the circuit computes y_i^{mon} and $\bar{y}_i^{\mathsf{mon}}$.

Finally we connect y_i^{mon} by an *and* gate to the selector circuit $\mathsf{Sel}_i'^{\mathsf{mon}}$. Note that this circuit is equal to an ℓ-slice function. As we will see later this ensures that the whole circuit outputs an ℓ-slice function. We connect the circuits $\bar{y}_i^{\mathsf{mon}}$ similarly: connect $\bar{y}_i^{\mathsf{mon}}$ by an *or* gate to the negated selector circuit $\overline{\mathsf{Sel}}_i'^{\mathsf{mon}}$. Again, the output of this circuit is equal to an ℓ-slice function.

Equally inportant is that these circuits behave well on the ℓ-slice. Indeed it can be checked that the positive circuit, on input $\alpha \in \{0,1\}^n$, outputs $\mathsf{Sel}_i'^{\mathsf{mon}}(\alpha) \wedge y_i$ while the negative circuit outputs $\overline{\mathsf{Sel}}_i'^{\mathsf{mon}}(\alpha) \vee \bar{y}_i$. On the ℓ-slice these functions are the negation of eachother, which we are going to use in the following.

We need to construct a monotone circuit for the *xor* of $\mathsf{Sel}_i'^{\mathsf{mon}}(\alpha) \wedge y_i$ for i from 1 to m, on ℓ-slice inputs α. We take a standard $O(m)$-size $\oplus$-circuit and monotonize it by pushing all negations in it down using De Morgan's law until they reach one of the $\oplus$-circuit's inputs $\mathsf{Sel}_i'^{\mathsf{mon}} \wedge y_i$. Whenever the negation of $\mathsf{Sel}_i'^{\mathsf{mon}}(\alpha) \wedge y_i$ is needed, we do one last step of De Morgan and replace it by $\overline{\mathsf{Sel}}_i'^{\mathsf{mon}}(\alpha) \vee \bar{y}_i$.

To ensure that the circuit outputs $f(\alpha)$ whenever $C_f^{\mathsf{mon}}(\alpha) \neq \perp$, we connect the two outputs of C_f^{mon} by an *and* gate and connect this gate by an *or* gate to the output of the *xor* circuit. This completes the description of the restriction on the structure variables. A depiction of the resulting circuit can be found in Figure 2. We ensure that s is large enough so that above circuit can be described by the formula.

Note that the constructed circuit always outputs a monotone ℓ-slice function: as the monotonized $\oplus$-circuit is non-constant, we see that if all inputs to the circuit are 0, it outputs 0 and if all inputs are 1, it outputs 1. This, in particular, implies that the circuit outputs 0 (respectively 1) if the input is below (respectively, above) the ℓ-slice and hence the entire circuit computes a monotone ℓ-slice function.

It can be easily checked that the described restriction is m-independent and k-determined. In order to prove the furthermore part, we need to reduce the number of evaluation variables. This can be achieved analogous to the proof of Lemma 25 and we thus omit it here. ◀

Let us prove our degree lower bound for monotone circuits, restated here for convenience.

▶ **Theorem 4.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For all $n, \ell \in \mathbb{N}$, all $s \geq n^d$ and any monotone slice function $f \in \mathcal{M}_n(\ell)$ SoS requires degree $\Omega_\varepsilon(s^{1-\varepsilon})$ to refute* $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$.

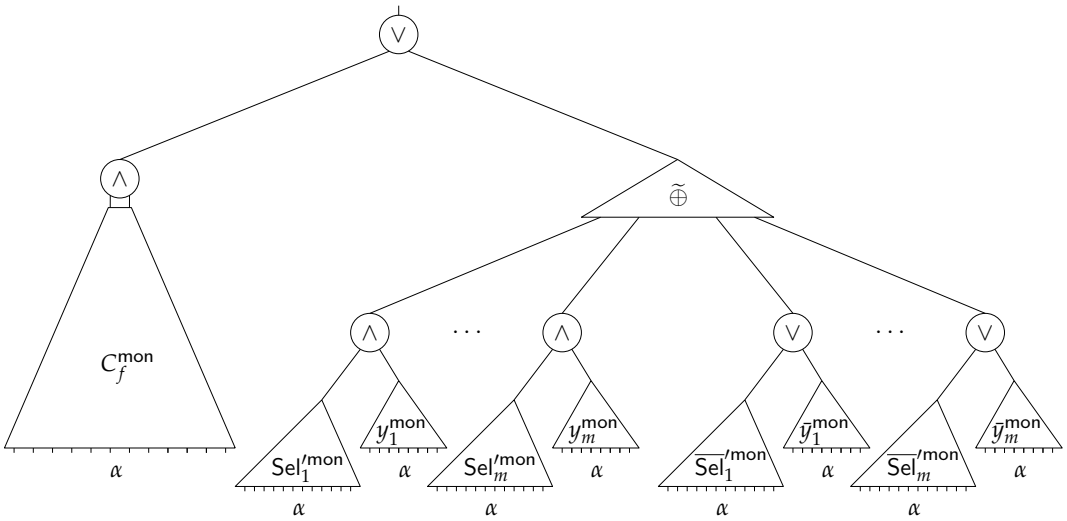


Figure 2 A depiction of the monotone circuit, where $\widetilde{\oplus}$ is the $\oplus$ circuit with the negations pushed down.

Proof of Theorem 4. As in the proof of Theorem 1, we use the graphs from Theorem 8, with $U = \{0,1\}^n$, $k = O_\gamma\big((n \log r)^{1+1/\gamma}\big)$, and $|V| \leq k^2 r^{1+\gamma}$ for parameters $\gamma > 0$ and $r \leq 2^n$. We apply Lemma 27 with above graph and $t = 2^n$ along with $C_f^{\mathsf{mon}} = \bot$ to obtain, for $s \geq m \cdot \mathsf{poly}(n,k)$, an appropriate natural m-independent k-determined variable substitution ρ for $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$. In particular ρ satisfies

$$g_{s,\alpha}^{\mathrm{out}}(Y) = \begin{cases} 1, & \text{if } |\alpha| > \ell, \\ 0, & \text{if } |\alpha| < \ell, \\ \oplus_{i \in N(\alpha)} y_i, & \text{otherwise,} \end{cases}$$

for $g_{s,\alpha}^{\mathrm{out}}$ and Y as in definition Definition 20.

Recall that there is a restriction τ such that $\mathrm{Circuit}_s^{\mathsf{mon}}(f) = \mathrm{Circuit}_s(f)\big|_\tau$ and we can thus apply Lemma 22 with $\tau\rho$ to conclude that if there is an SoS refutation of $\mathrm{Circuit}_s^{\mathsf{mon}}(f)\big|_\rho$ in degree d, then there is a degree $d \cdot k$ SoS refutation of the system of polynomial equations computing

$$\{ \bigoplus_{i \in N(\alpha)} y_i = f(\alpha) \mid \alpha \in \binom{[n]}{\ell} \} \ . \tag{20}$$

As the graph G is a strong expander, we can apply Theorem 13 to get an SoS degree lower bound of $\Omega(r)$ for above system of equations. By above connection this gives an $\Omega(r/k)$ degree lower bound for the $\mathrm{Circuit}_s^{\mathsf{mon}}(f)\big|_\rho$ formula and hence also for the unrestricted formula.

Regarding the parameters, as in the proof of Theorem 1 we choose $m = s/\mathsf{poly}_\gamma(n)$. Repeating the calculations from the aforementioned proof we obtain that $r \geq s^{1/(1+\gamma)}/\mathsf{poly}_\gamma(n)$. Thus by choosing γ small enough such that $\frac{1}{1+\gamma} > 1 - \varepsilon/2$ and s large enough such that the final $\mathsf{poly}_\gamma(n) \leq s^{\varepsilon/2}$ we obtain the claimed degree lower bound of $\Omega_\varepsilon(s^{1-\varepsilon})$. ◀

As in the non-monotone case, we can also obtain size lower bounds for functions that almost have a circuit of size s.

▶ **Theorem 5.** *For all $\varepsilon > 0$ there is a $d = d(\varepsilon)$ such that the following holds. For $n, \ell \in \mathbb{N}$, all $s \geq n^d$ and $t \geq s$ and monotone function $f \in \mathcal{M}_n(\ell, s/10, t)$ SoS requires size* $\exp\left(\Omega_\varepsilon(s^{2-\varepsilon}/t)\right)$ *to refute* $\mathrm{Circuit}_s^{\mathsf{mon}}(f)$.

Proof. Analogous to the proof of Theorem 3. ◀

6 Concluding Remarks

We have shown degree and size lower bounds in the Sum-of-Squares proof system for the minimum circuit size problem. There are a number of interesting questions left open for further study. Let us name a few.

Better Size Lower Bounds

Whereas our degree lower bounds apply for all Boolean functions f, the corresponding size lower bounds only apply to an albeit rich but still restricted class of functions.

Monotone Circuit Lower Bounds

For monotone circuits, we were only able to obtain lower bounds for slice functions (essentially because they behave in many ways like non-monotone functions). An intriguing question is whether this limitation can be overcome, or whether it is inherent and there exist some monotone circuit lower bounds that SoS *is* able to prove.

References

1. Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Pseudorandom generators in propositional proof complexity. *SIAM Journal on Computing*, 34(1):67–88, 2004. Preliminary version in *FOCS '00*.
2. S. R. Allen, R. ODonnell, and D. Witmer. How to refute a random csp. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 689–708, Los Alamitos, CA, USA, October 2015. IEEE Computer Society. doi:10.1109/FOCS.2015.48.
3. Albert Atserias and Tuomas Hakoniemi. Size-Degree Trade-Offs for Sums-of-Squares and Positivstellensatz Proofs. In Amir Shpilka, editor, *34th Computational Complexity Conference (CCC 2019)*, volume 137 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 24:1–24:20, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2019.24.
4. B. Barak, S. B. Hopkins, J. Kelner, P. Kothari, A. Moitra, and A. Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 428–437, 2016.
5. Amos Beimel and Enav Weinreb. Monotone circuits for monotone weighted threshold functions. *Inf. Process. Lett.*, 97(1):12–18, January 2006.
6. S. J. Berkowitz. On some relationships between monotone and non-monotone circuit complexity. Technical report, Technical Report, University of Toronto, 1982.
7. Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1):1–106, 2006. doi:10.1561/0400000004.
8. Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.
9. Oded Goldreich. *On (Valiant's) Polynomial-Size Monotone Formula for Majority*, pages 17–23. Springer International Publishing, Cham, 2020. doi:10.1007/978-3-030-43662-9_3.
10. Dima Grigoriev. Linear lower bound on degrees of positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1):613–622, 2001. doi:10.1016/S0304-3975(00)00157-2.
11. Dima Grigoriev, Edward A. Hirsch, and Dmitrii V. Pasechnik. Complexity of semi-algebraic proofs. In *STACS 2002*, pages 419–430, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

12 Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced expanders and randomness extractors from Parvaresh–Vardy codes. *Journal of the ACM*, 56(4):20:1–20:34, July 2009. Preliminary version in *CCC '07*.

13 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within np. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 247–258, 2018. doi:10.1109/FOCS.2018.00032.

14 Valentine Kabanets and Jin-Yi Cai. Circuit minimization problem. In *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing*, STOC '00, pages 73–79, New York, NY, USA, 2000. Association for Computing Machinery. doi:10.1145/335305.335314.

15 David Karger, Rajeev Motwani, and Madhu Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, March 1998. doi:10.1145/274787.274791.

16 Pravesh K. Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any csp. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 132–145, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055485.

17 Jan Krajíček. On the weak pigeonhole principle. *Fundamenta Mathematicae*, 170(1-2):123–140, 2001. URL: http://eudml.org/doc/282141.

18 Raghu Meka, Aaron Potechin, and Avi Wigderson. Sum-of-squares lower bounds for planted clique. In *Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC '15)*, pages 87–96, June 2015.

19 Cody D. Murrayand and R. Ryan Williams. On the (Non) NP-Hardness of Computing Circuit Complexity. In David Zuckerman, editor, *30th Conference on Computational Complexity (CCC 2015)*, volume 33 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 365–380, Dagstuhl, Germany, 2015. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik. doi:10.4230/LIPIcs.CCC.2015.365.

20 Toniann Pitassi and Ran Raz. Regular resolution lower bounds for the weak pigeonhole principle. *Combinatorica*, 24(3):503–524, 2004. Preliminary version in *STOC '01*. doi:10.1007/s00493-004-0030-y.

21 Aaron Potechin. Sum of Squares Bounds for the Ordering Principle. In Shubhangi Saraf, editor, *35th Computational Complexity Conference (CCC 2020)*, volume 169 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 38:1–38:37, Dagstuhl, Germany, 2020. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.CCC.2020.38.

22 Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly refuting random csps below the spectral threshold. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 121–131, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055417.

23 Ran Raz. Resolution lower bounds for the weak pigeonhole principle. *J. ACM*, 51(2):115–138, 2004. doi:10.1145/972639.972640.

24 Alexander Razborov. P, NP and Proof Complexity. https://youtu.be/ZVL_HsPC4xE?t=2646, 2021. Accessed April 2022.

25 Alexander Razborov. Open problems. https://people.cs.uchicago.edu/~razborov/teaching/index.html, 2022. Accessed April 2022.

26 Alexander A. Razborov. Lower bounds for the polynomial calculus. *Computational Complexity*, 7(4):291–324, December 1998.

27 Alexander A. Razborov. Resolution lower bounds for perfect matching principles. *Journal of Computer and System Sciences*, 69(1):3–27, August 2004. Preliminary version in *CCC '02*.

28 Alexander A. Razborov. Pseudorandom generators hard for k-DNF resolution and polynomial calculus resolution. *Annals of Mathematics*, 181(2):415–472, March 2015.

29 Alexander A. Razborov, Avi Wigderson, and Andrew Chi-Chih Yao. Read-once branching programs, rectangular proofs of the pigeonhole principle and the transversal calculus. *Combinatorica*, 22(4):555–574, 2002. Preliminary version in *STOC '97*. doi:10.1007/s00493-002-0007-7.

30 Leslie G. Valiant. Short monotone formulae for the majority function. *J. Algorithms*, 5:363–366, 1984.

Leakage-Resilient Hardness vs Randomness

Yanyi Liu ✉
Cornell Tech, New York, NY, USA

Rafael Pass ✉
Tel-Aviv University, Israel
Cornell Tech, New York, NY, USA

Abstract

A central open problem in complexity theory concerns the question of whether all efficient randomized algorithms can be simulated by efficient deterministic algorithms. The celebrated "hardness v.s. randomness" paradigm pioneered by Blum-Micali (SIAM JoC'84), Yao (FOCS'84) and Nisan-Wigderson (JCSS'94) presents hardness assumptions under which e.g., $\mathsf{prBPP} = \mathsf{prP}$ (so-called "high-end derandomization), or $\mathsf{prBPP} \subseteq \mathsf{prSUBEXP}$ (so-called "low-end derandomization), and more generally, under which $\mathsf{prBPP} \subseteq \mathsf{prDTIME}(\mathcal{C})$ where $\mathcal{C}$ is a "nice" class (closed under composition with a polynomial), but these hardness assumptions are not known to also be necessary for such derandomization.

In this work, following the recent work by Chen and Tell (FOCS'21) that considers "almost-all-input" hardness of a function f (i.e., hardness of computing f on more than a finite number of inputs), we consider "almost-all-input" *leakage-resilient hardness* of a function f – that is, hardness of computing $f(x)$ even given, say, $\sqrt{|x|}$ bits of leakage of $f(x)$. We show that leakage-resilient hardness characterizes derandomization of prBPP (i.e., gives a both *necessary* and *sufficient* condition for derandomization), both in the high-end and in the low-end setting.

In more detail, we show that there exists a constant c such that for every function T, the following are equivalent:

- $\mathsf{prBPP} \subseteq \mathsf{prDTIME}(\mathsf{poly}(T(\mathsf{poly}(n))))$;
- Existence of a $\mathsf{poly}(T(\mathsf{poly}(n)))$-time computable function $f : \{0,1\}^n \to \{0,1\}^n$ that is almost-all-input leakage-resilient hard with respect to n^c-time probabilistic algorithms.

As far as we know, this is the first assumption that characterizes derandomization in both the low-end and the high-end regime.

Additionally, our characterization naturally extends also to derandomization of prMA, and also to average-case derandomization, by appropriately weakening the requirements on the function f. In particular, for the case of average-case (a.k.a. "effective") derandomization, we no longer require the function to be almost-all-input hard, but simply satisfy the more standard notion of average-case leakage-resilient hardness (w.r.t., every samplable distribution), whereas for derandomization of prMA, we instead consider leakage-resilience for relations.

2012 ACM Subject Classification Theory of computation → Pseudorandomness and derandomization

Keywords and phrases Derandomization, Leakage-Resilient Hardness

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.32

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2022/113/

Funding *Yanyi Liu*: Work done while visiting Tel-Aviv University.
Rafael Pass: Supported in part by NSF Award CNS 2149305, NSF Award SATC-1704788, NSF Award RI-1703846, AFOSR Award FA9550-18-1-0267, a JP Morgan Faculty Award, and an Algorand Foundation grant (MEGA-ACE). This material is based upon work supported by DARPA under Agreement No. HR00110C0086. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 32; pp. 32:1–32:20

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

1 Introduction

Randomness is an ubiquitous tool in algorithm design. A central open problem in complexity theory concerns the question of whether all randomized algorithms can be derandomized; that is, can every randomized polynomial-time algorithm be simulated by a deterministic polynomial-time, or perhaps even just sub-exponential one? In this work, we consider this question with respect to promise problems; as usual, we refer to prBPP as the class of promise problems (as opposed to languages) that can be solved in probabilistic polynomial time (with 2-sided error), and prP to the class of promise problems than can be solved in deterministic polynomial time.

We here focus on the questions of whether $\mathsf{prBPP} = \mathsf{prP}$ (the, so-called, "high-end regime"), $\mathsf{prBPP} \subseteq \mathsf{prSUBEXP}$ (the, so-called, "low-end regime") and more generally whether $\mathsf{prBPP} \subseteq \mathsf{prDTIME}(\mathcal{C})$, where $\mathcal{C}$ is a "nice" class of time bounds (where by "nice" we here mean that $\mathcal{C}$ is closed under composition with a polynomial).

A long sequence of works originating with the works of Blum-Micali [3], Yao [30], Nisan [23], Nisan-Wigderson [24], Babai-Fortnow-Nisan-Wigderson [2], Impagliazzo-Wigderson [14] have presented beautiful connections between this problem and the problem of proving computational-complexity lower bounds – the so-called *hardness v.s. randomness* paradigm. For instance, the results of [24, 14] show that $\mathsf{prBPP} = \mathsf{prP}$ under the assumption that $\mathsf{E} = \mathsf{DTIME}(2^{O(n)})$ contains a language that requires Boolean circuits of size $2^{\Omega(n)}$ for almost all input lengths (i.e., E is not contained in $\mathsf{ioSIZE}(2^{\Omega(n)})$). Additionally, results by Impagliazzo, Kabanets and Wigderson [13] show a *partial* converse: if $\mathsf{prBPP} = \mathsf{prP}$, then some non-trivial circuit lower bound must also hold. In more detail, if $\mathsf{prBPP} = \mathsf{prP}$ (or even just $\mathsf{MA} = \mathsf{NP}$), then $\mathsf{NEXP} \not\subseteq \mathsf{P/poly}$; very recent works [28, 22] managed to strengthen the conclusion to e.g., $\mathsf{NTIME}[n^{\mathsf{poly}\log n}] \not\subseteq \mathsf{P/poly}$.

But despite over 40 years of research on the topic of derandomization, there has still been a large "gap" between the hardness assumptions required for derandomizing prBPP, and the ones that are known to be necessary for derandomization, leaving open the following question:

> *For "nice" classes $\mathcal{C}$, does there exist some (natural) hardness assumption that is* **equivalent** *to $\mathsf{prBPP} \subseteq \mathsf{prDTIME}[\mathcal{C}]$?*

Most notably, known derandomization results for prBPP require complexity lower-bounds on functions in EXP, whereas it is only known that derandomization of prBPP implies complexity lower bounds for functions in non-deterministic classes.

There as been some recent progress on the above problem:

- An elegant work by Chen, Rothblum, Tell and Yogev show an equivalence between derandomization and circuit lower bound under a conjecture (a weaker version of the non-deterministic exponential-time hypothesis) [5]. There work applies for both the high-end and the low-end regime, but is only conditional (i.e., relies on a conjecture).
- Chen and Tell [6], relying on the work by Goldreich [9], show that the existence of a multi-output function $f : \{0,1\}^n \to \{0,1\}^n$ computable by polynomial size logspace-uniform circuits with depth bounded by n^2 that cannot be computed in some (a-priori bounded) probabilistic polynomial time on *any* sufficiently large input – this is referred to as *"almost-all-input hardness"* – implies that $\mathsf{prBPP} = \mathsf{prP}$. They also show a partial converse: That a relaxed version of this conjecture, where the depth requirement is dropped, also is necessary.
- Liu and Pass [18], following the work of Hirahara [11], demonstrates a *class* of promise problems (related to conditional time-bounded Kolmogorov complexity) such that (worst-case) hardness with respect to a-priori polynomially bounded probabilistic algorithms of

all problems in the class is equivalent to $\mathsf{prBPP} = \mathsf{prP}$. (The result of Liu and Pass also shows that a single problem can be used to characterize $\mathsf{prBPP} = \mathsf{prP}$ but this problem is very artificial.) Similar to the results of [6], this characterization can be extended slightly beyond the "high-end regime", but fails to capture the "low-end regime": it only works to handle derandomization in time $\mathcal{C}$, where $\mathcal{C}$ is a class closed under composition (i.e., if $T \in \mathcal{C}$, then $T(T(\cdot) \in T)$, and thus already does not apply to e.g., SUBEXP.

- Finally, a very recent elegant work by Korten [17] demonstrates a natural search problem – the R-Lossy Code problem – that is complete for prBPP. As such, the assumption that this problem can be solved in deterministic time $\mathcal{C}$ characterizes when $\mathsf{prBPP} \subseteq \mathsf{DTIME}(\mathcal{C})$. We note, however that this assumption is not a hardness assumption, but rather an "easiness" assumption.

Thus summing up, it is known how to characterize both the high-end and low-end derandomization through a hardness assumption under a conjecture [5]; unconditionally, however, it is only known how to characterize the high-end regime (i.e., $\mathsf{prBPP} = \mathsf{prP}$) and even there it is only known under either (a) a class of hardness assumptions (as opposed to one), or (b) a very specific and artificial single hardness assumption.

In this work, we follow the work by Chen and Tell [6] and also consider the notion of "almost-all-input" hardness of a multi-output function. In contrast to them, however, we consider such "almost-all-input" hardness in the context of *leakage resilience*, a notion first considered in the cryptographic literature in the 1980s. As we shall see, our main result shows that "almost-all-input" leakage resilient hardness can be used to fully characterize derandomization, both in the high-end and in the low-end setting; additionally, our characterization will extend also to derandomization of prMA, and also to average-case (a.k.a. "effective") derandomization. In particular, for the case of average-case derandomization, we no longer require the function to be almost-all-input hard, but simply satisfy the standard notion of average-case leakage-resilient hardness (w.r.t., every samplable distribution). And to characterize derandomization of prMA, we instead consider the notion of a *leakage-resilient relation*, which again is a notion considered in the cryptographic literature. Taken together, we believe that our results demonstrate an intriguing connection between derandomization and notions from cryptography.

1.1 Leakage-resilient Hardness

Consider some multi-output function $f : \{0,1\}^n \to \{0,1\}^n$. Roughly speaking, we say that f is T-*hard* if no $T(|x|)$-time algorithm/attacker A can compute $f(x)$ given any input x. Perhaps the most widely known example of a candidate construction of a hard function is *integer factorization*: given a product of two-primes x, compute the factorization $f(x)$ of x. In 1985, Rivest and Shamir [26] asked the question of what happens to this candidate hard function if the attacker gets some additional "side-information" about the factorization of x. Namely, the attacker gets not only x, but also some T-time computable side-information (a.k.a. "leakage") $\mathsf{leak}(x, f(x))$. Of course, if $|\mathsf{leak}(x, f(x))| \geq |f(x)|$, then the problem becomes trivial since the side-information can simply reveal the whole factorization; in fact, for the particular factorization problem, it trivially suffices to leak $n/2$ bits to reveal just one of the primes. Rivest and Shamir [26] show that, in fact, it suffices to get $n/3$ bits of leakage for the function to becomes easy; this result was improved by Coppersmith [7] to $n/4$ bits, and a heuristic (with a conjectured polynomial running-time bound) by Maurer [20] shows an attack given just ϵn bits of leakage for any constant $\epsilon > 0$. As far as we know, it is unknown if this problem can be solved using just n^ϵ bits of leakage, for any $\epsilon < 1$.

The notion of leakage-resilient hardness captures the notion that a function is hard even given some *bounded-length* leakage [26, 20, 1]: We say that a function f is (T, ℓ)-*leakage resilient hard* if no $T(|x|)$-time attacker A can compute $f(x)$ given x and $\mathsf{leak}(x, f(x))$ for any $T(|x|)$-time computable leakage function leak that outputs at most $\ell(|x|)$ bits. In recent years, *leakage-resilient cryptography* [15, 21, 8, 1] – the design of cryptographic protocols resilient to some forms of leakage of honest players' secrets – has received significant attention and has become a subfield in cryptography; the bounded-length leakage model is the most common way of formalizing the class of leakage functions.

In this paper, we will consider this notion of leakage-resilient hardness, except that following Chen and Tell, we will also consider it in the context of almost-all-input hardness: that is, for any pair (A, leak) there can be at most finitely many x for which A can compute $f(x)$ given x and $\mathsf{leak}(x, f(x))$.

1.2 Characterizing Derandomization

We are now ready to state our main theorem, which shows that almost-all-input leakage-resilient hardness characterizes both high-end and low-end derandomization. We say that a class of time-bounds $\mathcal{C}$ is *nice* if for all polynomials p, q it holds that if $T \in \mathcal{C}$, then $p(n)T(q(n)) \in \mathcal{C}$. For instance, the sets $\mathsf{poly}(n)$ and $2^{n^{o(1)}}$ are nice and recall that $\mathsf{P} = \mathsf{DTIME}(\mathsf{poly})$ and $\mathsf{SUBEXP} = \cap_{\epsilon>0}\mathsf{DTIME}(2^{n^\epsilon})$.

Derandomizing prBPP. Our main theorem gives a characterization of derandomization of prBPP:

▶ **Theorem 1.1.** *There exists a constant c such that for every "nice" class of running-time bounds $\mathcal{C}$, and for every $0 < \epsilon < 1$, the following are equivalent:*

- $\mathsf{prBPP} \subseteq \cup_{T\in\mathcal{C}}\mathsf{prDTIME}[T(n)]$.
- *The existence of a multi-output function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic time $T \in \mathcal{C}$ such that f is almost-all-input (n^c, n^ϵ)-leakage-resilient hard.*

In particular, $\mathsf{prBPP} = \mathsf{prP}$ (resp. $\mathsf{prBPP} \subseteq \mathsf{prSUBEXP}$) iff there exists a polynomial-time (resp. subexponential-time) computable multi-output function f that is almost-all-input $(n^c, \sqrt{n})$-leakage-resilient hard.

Some Corollaries to Leakage-resilient Hardness. For the proof of Theorem 1.1, in one direction, we only require leakage-resilience with a "small" amount of leakage, whereas in the other direction, we obtain a function that is leakage-resilient hard with a "large" amount of leakage. Consequently, our proof actually also yields an amplification theorem for leakage-resilient hardness: For any $\epsilon > 0, d \geq 1$, the existence of an efficient (n^c, n^ϵ)-leakage-resilient hard function implies an efficient $(n^d, n - \Omega(\log n))$-leakage-resilient hard function.

▶ **Theorem 1.2.** *There exists a constant c such that for every nice class $\mathcal{C}$, all constants $\epsilon > 0, d \geq 1$, the following holds. If there exists a $\mathcal{C}$-computable almost-all-input (n^c, n^ϵ)-leakage-resilient hard function. Then there exists a $\mathcal{C}$-computable almost-all-input $(n^d, n - 3\log n)$-leakage-resilient hard function.*

Let us highlight that as far as we know, this is the first type of leakage-resilience amplification result in the literature that we are aware of. (Brakerski and Kalai [4] demonstrate a parallel repetition theorem for leakage-resilience but it does not show how to amplify the amount of leakage a function is resilient against, but rather only how to maintain the (relative) amount of leakage.)

Additionally, by combing the result of Chen and Tell [6] with our Theorem 1.1, we get an interesting (one-sided) connection between low-depth computable hard functions and leakage-resilience:

▶ **Theorem 1.3.** *There exists some c such that the following holds. If there exists a function f computable by polynomial-size logspace-uniform circuits with depth bounded by n^2 that is almost-all-input n^c-hard, then for any constant $d \geq 1$, there exists a polynomial-time computable almost-all-input $(n^d, n - 3\log n)$-leakage-resilient hard function.*

Comparison with IW: leakage-resilient *local* hardness. Recall that Impagliazzo and Wigderson [14] shows that $\mathsf{prBPP} = \mathsf{prP}$ under the assumption that $\mathsf{E} \not\subseteq \mathsf{ioSIZE}(2^{\Omega(n)})$ (i.e., that there exists some exponential-time computable function that does not have $2^{\Omega(n)}$-size circuits.) Since Theorem 1.1 shows that leakage-resilient hardness is both a sufficient and necessary condition for derandomizing prBPP, it directly follows that $\mathsf{E} \not\subseteq \mathsf{ioSIZE}(2^{\Omega(n)})$ implies leakage-resilient hardness (by combining [14] with Theorem 1.1), but it gives little insight into whether the type of assumption used by IW is inherent, or to what extent it "overshoots". Indeed, understanding to what extent the NW/IW framework is inherent for derandomization is a long standing open problem.

We now show how to use our framework to provide an (in our eyes) crisp answer to this question. We start by noting that by a slight adjustment to the proof of Theorem 1.1, an (a-priori) weaker notion of leakage-resilient *local* hardness actually suffices to derandomize prBPP: Given a function $f : \{0,1\}^n \to \{0,1\}^n$, we say that A *t-locally computes $f(\cdot)$* on input x if for every $i \in [|x|]$, $A(x,i) = f(x)_i$ (i.e., the ith bit of $f(x)$), and $A(x,i)$ runs in time bounded by $t(|x|)$; we analogously say that A *t-locally computes f on input x given (T,ℓ)-leakage* leak if for every $i \in [|x|]$, $A(x, \mathsf{leak}(x, f(x)), i) = f(x)_i$, $A(x, z, i)$ runs in time bounded by $t(|x|)$, $\mathsf{leak}(x, f(x))$ runs in time bounded by $T(|x|)$, and $|\mathsf{leak}(x, f(x))| \leq \ell(|x|)$. We finally say that f is *almost-all-input (T,ℓ)-leakage resilient t-local hard* if there does not exist (A, leak) such that A t-locally computes f on infinitely many x given (T,ℓ)-leakage leak. Note that this notion of leakage-resilient hardness differs from the standard one in two ways: (a) we are decoupling the running time T of the leakage function, and the running time t of the computing machine A, and (b) we require the computing machine A to be able to locally compute each bit of the output of $f(x)$ – this will allow us to consider sublinear running times t. Indeed, we will focus our attention on the regime where A is required to reconstruct each bit of $f(x)$ in sublinear time: We say that f is simply *almost-all-input (T,ℓ)-leakage resilient locally hard* if there exists some $0 < \epsilon < 1$ such that f is almost-all-input (T,ℓ)-leakage resilient t-locally hard where $t(n) = n^\epsilon$. Note that (T,ℓ)-leakage resilient hardness is trivially an (a-priori) stronger condition than (T,ℓ)-leakage resilient local hardness when $T \in \Omega(n^2)$. We now have the following generalization of Theorem 1.1:

▶ **Theorem 1.4.** *There exists a constant $c \geq 2$ such that for every "nice" class of running-time bounds $\mathcal{C}$, and for every $0 < \epsilon < 1$, the following are equivalent:*

- $\mathsf{prBPP} \subseteq \cup_{T \in \mathcal{C}} \mathsf{prDTIME}[T(n)]$.
- *The existence of a multi-output function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic time $T \in \mathcal{C}$ such that f is almost-all-input (n^c, n^ϵ)-leakage-resilient* locally *hard.*
- *The existence of a multi-output function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic time $T \in \mathcal{C}$ such that f is almost-all-input (n^c, n^ϵ)-leakage-resilient hard.*

Leakage-resilient local hardness is useful as it allows to capture the assumption that $\mathsf{E} \not\subseteq \mathsf{ioSIZE}(2^{\Omega(n)})$. In fact, we observe that $\mathsf{E} \not\subseteq \mathsf{ioSIZE}(2^{\Omega(n)})$ directly implies the existence of a (n^c, n^ϵ)-leakage-resilient locally hard function for every c. To see this, consider some

E-computable function $g_n : \{0,1\}^n \to \{0,1\}$ that does not have circuits of size $2^{\Omega(n)}$; in other words, g_n cannot be computed by a $2^{\Omega(n)}$ time algorithm even when given any (potentially non computable) advice string. Define the multi-output function f that is *constant* on each input length n and simply outputs the truthtable of $g_{\log n}$; that is,

$$f(x) = g_m(1)||g_m(2)||\dots||g_m(|x|)$$

where $m = \log |x|$. Note that $f(x)$ is polynomial-time computable (since we are only evaluating g_m on input of logarithmic length in $|x|$), and additionally by the hardness of g, it directly follows that f is almost-all-input leakage-resilient *locally* hard (since locally computing f on some input x in sublinear time with *efficiently computable* leakage implies computing $g_{\log |x|}(y)$ on every input $y \in \{0,1\}^{\log |x|}$ in subexponential time with advice).[1] In fact, it directly follows that this construction is almost-all-input leakage-resilient locally hard even with respect to *uncomputable* leakage.

In other words, the IW assumption "overshoots" the minimal assumption needed for derandomizing BPP in two ways: (a) it considers leakage-resilient hardness of a "degenerated" multi-output function that is *constant* on each input length, and (b) it requires leakage resilient hardness also with respect to *uncomputable* leakage, whereas the minimal assumption only requires it w.r.t. *polynomial-time* computable leakage.

Effective Derandomization. Goldreich [9] (see also [10, 16]) considers a notion of "effective" derandomization of prBPP where the derandomizer does not need to work on all inputs – rather, it can fail sometimes, but only on inputs that are "hard to find" – in more detail, no PPT finder/refuter can find instances on which the derandomization fails except with negligible probability. In essence, effective derandomization is good enough for all efficient applications of derandomization.

For technical reasons, however, Goldreich, is not able to characterize such effective derandomization, but rather only a notion of $p(\cdot)$-effective derandomization where the finder/refuter running time is bounded by $p(n)$ for some fixed polynomial p and it success probability is bounded by $\frac{1}{p(n)}$. (In more details, for every a-priori fixed polynomial bound $p(\cdot)$ on the running-time/success probability of a refuter, we require the existence of derandomization that works for that particular bound. In contrast, effective derandomization (as we consider it here) requires the existence of a single derandomization procedure that works for *any* polynomial-time refuter, and with only negligible failure probability.)

Using our leakage-resilient framework, we can get a clean characterization also of effective derandomization through average-case leakage-resilient hardness, where average-case leakage-resilient hardness with respect to some distribution $\mathcal{D}$ is defined just like before except we now consider instances x sampled from $\mathcal{D}$ and we allow the attacker A to succeed on at most a negligible fraction of instances.

More precisely, we say that Π is *effectively contained* in Π' (denoted $\Pi \subseteq_{\mathsf{poly}} \Pi'$) if for every PPT A there exists a negligible μ such that the probability that $A(1^n)$ is able to output an n-bit element in the symmetric difference between Π and Π' is bounded by $\mu(n)$; we may extend this notion of classes of problems in the usual way: $\mathcal{D} \subseteq_{\mathsf{poly}} \mathcal{D}'$ iff for every $\Pi \in \mathcal{D}$, there exists some $\Pi' \in \mathcal{D}'$ such that $\Pi \subseteq_{\mathsf{poly}} \Pi'$.

[1] There is a minor subtlety here. If we have an attacker A that locally computes $f(x)$ on some input x in sublinear time, then A can compute $g_{\log |x|}(y)$ for every $y \in \{0,1\}^{\log |x|}$ in sublinear time *given access to* x which can be of exponential length compared to $|y|$. But since A runs in sublinear time, it can access at most a sublinear number of bits of x, and thus we can compute g_m by a circuit of subexponential size.

▶ **Theorem 1.5.** *There exists a constant c such that for every $0 < \epsilon < 1$, the following are equivalent:*

- $\mathsf{prBPP} \subseteq_{\mathsf{poly}} \mathsf{prP}$.
- *The existence of a multi-output function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic polynomial time such that f is average-case (n^c, n^ϵ)-leakage-resilient hard for every efficiently samplable distribution $\mathcal{D}$.*

The result also extends to the low-end regime but only if we consider a stronger form of effective containment (which allows the refuter to have super polynomial running time).

Derandomizing prMA. Finally, we consider the problem of derandomizing prMA (as opposed to just prBPP). Here, we require considering the notion of a leakage-resilient hard *relation* [25], which is identically defined to that of a leakage-resilient hard function, except that any input x can be mapped to multiple values $y \in R(x)$. We show:

▶ **Theorem 1.6.** *There exists a constant c such that for every "nice" class of running-time bounds $\mathcal{C}$, and for every $0 < \epsilon < 1$, the following are equivalent:*

- $\mathsf{prMA} \subseteq \cup_{T \in \mathcal{C}} \mathsf{prNTIME}[T(n)]$.
- *The existence of a relation $R \subset \{0,1\}^n \times \{0,1\}^n$ computable in non-deterministic time $T \in \mathcal{C}$ such that R is almost-all-input (n^c, n^ϵ)-leakage-resilient hard.*

In particular, $\mathsf{prMA} = \mathsf{prNP}$ iff there exists a relation $R \in \mathsf{NP}$ that is almost-all-input $(n^c, \sqrt{n})$-leakage-resilient hard.

Proof Overview. We focus our attention on the proof of Theorem 1.1 (and Theorem 1.4); afterwards, we briefly discuss how to extend these techniques to prove the remaining results. For simplicity, here focusing only on the high-end setting, but the key point is that the same technique *directly* extends also to the low-end setting.

- **Leakage-resilient Hardness implies $\mathsf{prBPP} = \mathsf{prP}$:** Following Goldreich [9], we consider the notion of a *targeted PRG* – roughly speaking, this is a PRG g that gets an additional *target* z as input, and indistinguishability holds with respect to uniform algorithms that also get the target z as input. In other words, g is just like a normal PRG, but with the exception that both the PRG and the distinguisher get access to the auxiliary "target" string z, and we require security to hold for all strings z. (Since we consider PRGs in the context of derandomization, we allow the running-time of the PRG to be (polynomially) larger than the running-time of the distinguisher.) Using standard techniques, it follows that the existence of such a targeted PRG, with sufficiently large stretch, implies that $\mathsf{prBPP} = \mathsf{prP}$ (simply let the instance to be decided be the target for the PRG).

  We next show how to use leakage-resilient hardness to construct a targeted PRG. Assume the existence of a leakage-resilient hard multi-output function $f : \{0,1\}^n \to \{0,1\}^n$. Given a target $z \in \{0,1\}^n$, we compute $g_z = \mathsf{ECC}(f(z))$, where ECC is an appropriate list-decodable error-correcting code with good parameters, and interpret g_z as a hard function to use in the Impaglizzo-Wigderson (IW)/Nisan-Wigderson (NW) pseudo-random generator generator [14, 24]. That is, we are relying on the Sudan-Trevisan-Vadhan PRG [27]. The IW-NW proof essentially shows that given a distinguisher D for the PRG, and some (bounded-length) advice about the truthtable (and D), we efficiently compute the evaluation of g_z with probability $1/2 + \frac{1}{p(n)}$ for some polynomial p over random n-bit inputs. We observe that this advice in fact can be efficiently computed if we have access to the truthtable of g_z and the distinguisher D, and we can thus view it as efficiently leakage on $(z, f(z))$. We can next list-decode (again efficiently) g_z and

recover a polynomial-length list of candidates for $f(z)$; given $f(z)$, we can efficiently determine which of these candidates is the correct one, and also include the index of this candidate in the leakage (which again will be short). Given both these leakage, we can now re-compute $f(z)$ by simply again running the list-decoding algorithm and outputting the string specified by the index. We note that we here rely on the fact that once this leakage has been fixed, the rest of the NW reconstruction procedure is deterministic, and furthermore, the list-decoding procedure is also deterministic, so the attacker A can recompute the same list of candidates in the same order, and thus we are guaranteed that it also recovers the exact same string.

In other words, if anyone can break the targeted PRG on infinitely many targets z, we can compute $f(z)$ on infinitely many inputs z given short and efficiently computable leakage on $(z, f(z))$; we note that, somewhat curiously, the leakage function actually also needs to access z and not just $f(z)$ in order to simulate the distinguisher D (that gets z as an input).

We finally observe that if the error-correcting code additionally satisfies a *local* list decoding property – e.g., by using the error-correcting code of [27] – then we can actually locally compute each bit of $f(z)$ in sublinear time in the length of $f(z)$ which we can use to conclude also the implication in the proof of Theorem 1.4. There is just one small catch; the local list decoding procedure will be randomized, so we may not necessarily recover the same list of candidates, or the same ordering of them. But the list-decoding procedure has a small running time and thus also uses a small amount of randomness, so we can just include this additional randomness as part of the leakage.

In the actual proof, we show the above in a more modular way:

- We first consider a notion of a *strongly black-box PRG* – roughly speaking a PRG based on f for which there exists (a) an efficient algorithm that given black-box access to f and some distinguisher for the PRG outputs some advice string, and (b) another efficient algorithm that given this advice string and black-box access to the same distinguisher, is able to efficiently compute f. This notion is a strengthening of the notion of a black-box PRG from [29] where the advice string did not need to be efficiently computable. Nevertheless, following [12], we note that the advice string needed in the reduction to prove security of the [27] PRG construction actually can be efficiently computed.
- Next, we show that any such strongly black-box PRG construction can be used to get a targeted PRG from leakage-resilient hardness.

- $\mathsf{prBPP} = \mathsf{prP}$ **implies Leakage-resilient Hardness:** Our proof, roughly speaking, proceeds in two steps. First, we show using an information theoretic argument that a random function f is almost-all-input leakage-resilient hard. Next, we show how this function can be "derandomized" assuming $\mathsf{prBPP} = \mathsf{prP}$ – the crucial aspect that makes this derandomization possible is that it is possible to *efficiently* verify whether there exists some attacker that efficiently computes the function on some input given efficient leakage (by enumerating the $\log n$ first Turing machines and evaluating them).

  For the first step, we use a simple compression argument to show that for every attacker, leakage-function pair (A, leak), for any input x, with high probability over the choice of $y = F(x) \in \{0,1\}^n$, it is the case that the attacker A can compute $F(x)$ with probability at most, say, $1/6$. In fact, we will show a slightly stronger statement: for any A, x, with high probability over y, there does not exists any leakage function, leak, such that (A, leak) computes y with probability $1/6$. The advantage of this stronger formulation is that now it is without loss of generality to restrict attention to *deterministic* leakage functions leak (since for any fixed A, x, y we can always consider the deterministic leakage function that fixes the best randomness).

To prove the (stronger) statement, let us first consider the case when also the attacker A is deterministic. Since the length ℓ of the leakage is significantly shorter than $|y|$, it follows that for any fixed x, most strings y are not in the range of what the attacker can output given x and *any* leakage, and thus for every x, with high probability over the choice of y, the attacker fails to output y no matter what the function leak is.

Next, note that even if A is randomized, there can be at most 6 strings that A outputs with probability $1/6$ given any fixed x and any fixed leakage output, so the above argument actually also extends to randomized A (except that we increase the number of string y that can be hit by a factor 6). This thus concludes a random choice of y will with high probability not be computable by any (A, leak).

Next, we show that for any x, this random choice of y can actually be derandomized assuming $\mathsf{prBPP} = \mathsf{prP}$, and relying on the fact that we only need y to be hard to compute with respect to *uniform* (bounded) polynomial-time computable (A, leak). Towards this, we follow the approach of Goldreich [9] and show that for any x, we can greedily compute the bits of $y = f(x)$ one at a time, relying on the fact that we know that a random selection will work, and then use the fact that $\mathsf{prBPP} = \mathsf{prP}$ to efficiently find a good selection. In more details, we know that with high probability over the choice of y, the first $\log n$ uniform (A, leak) machines with running time bounded n^c will fail to compute y so we can start by picking bit 1 y_1 of y that leads to a high probability over the continuation of y of all those $\log n$ machines failing to compute y. Estimating this probability requires randomness, but if $\mathsf{prBPP} = \mathsf{prP}$ then it can also be done deterministically. This second step can be done in a modular way by appealing to the elegant BPP-decision-to-search reduction of Goldreich [9] and by appropriately specifying the above problem of finding a "good" y that fools the $\log n$ first uniform (A, leak) machine with running time n^c (in the sense that their estimated success probability is significantly smaller than $1/6$) as a BPP-search problem.

Effective Derandomization. To characterize "effective derandomization", the proof follows a very similar structure, except to perform the derandomization we instead pass through a new notion of an "average-case targeted PRG". The converse direction (showing necessity of the assumption) becomes a bit more complicated than before as we no longer have access to a perfect derandomization, but these additional subtleties can be dealt with. (Roughly speaking, the issue is that since the derandomizer only succeeds on average, we may run into trouble during the decision to search process. On a high-level, the way we get around these issues is by relying on the fact that we only need to derandomize a *single* fixed problem, and that effective derandomization yields a single derandomized algorithm for solving it, and we can next show that if an error occurs during the decision to search process, the location of the first mistake can be efficiently guessed.)

Characterizing Derandomization of prMA. To show how to derandomize prMA, we instead pass through a new notion of a non-deterministic targeted PRG, which can be instantiated from our assumption and which implies derandomization of prMA.

To prove the converse direction, we proceed a bit different from the proof of Theorem 1.1 – we can no longer passing through the above-mentioned search-to-decision reduction, as no such reduction is known for prMA. Instead, we show how to directly construct a leakage-resilient hard relation from the assumption that prMA can be derandomized using a diagonalization argument. Roughly speaking, we show that the above described BPP-search problem (of given a string x, finding some y that is leakage-resilient hard to compute w.r.t. the first

$\log n$ algorithms) actually is a leakage-resilient hard relation assuming that prMA can be derandomized! First note that this problem trivially is in prMA and thus also in prNP under the assumption that prMA = prNP. More interestingly, if there exists some attacker A that given an input x and given some efficient leakage on x can compute a y in this relation, then this y cannot be in the relation (since by definition, y cannot be computed by any efficient algorithm with small description), which is a contradiction.

Paper Overview. In Section 3, we present an equivalence between derandomizing prBPP and leakage-resilient hardness. In Section 4, we present a characterization of derandomizing prMA via the notion of leakage-resilient relation. The results on average-case derandomization are deferred to the full version [19].

2 Preliminaries

We assume familiarity with basic concepts such as Turing machines, polynomial-time algorithms, and probabilistic algorithms and computational classes such as prBPP and prP. We say that a function f is *time-constructible* if f is increasing and for all $n \in \mathbb{N}$, $f(n)$ can be computed by a Turing machine in time $\mathsf{poly}(f(n))$. We say that a class of functions $\mathcal{C}$ is *nice* if for all $T \in \mathcal{C}$, $t(p(n))q(n) \in \mathcal{C}$ for all polynomials p, q.

We say that $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$ is an *ensemble* if for all $n \in \mathbb{N}$, D_n is a probability distribution over $\{0,1\}^n$. We say that an ensemble $\mathcal{D} = \{D_n\}_{n\in\mathbb{N}}$ is *samplable* if there exists a probabilistic polynomial time Turing machine S such that $S(1^n)$ samples D_n.

2.1 Leakage Resilient Hardness of (Multi-output) Functions

We consider multi-output functions $f : \{0,1\}^n \to \{0,1\}^n$ (as opposed to binary functions, traditionally considered in the derandomization literature). We will focus on a leakage-resilient notion of hardness. Roughly speaking, we say that f is (T, ℓ)-*leakage resilient hard* if no T-time attacker can compute $f(x)$ given x and $\mathsf{leak}(x, f(x))$, where leak is any T-time computable leakage function such that $\mathsf{leak}(x, f(x)) \leq \ell(|x|)$. We will consider this notion of in the context of *almost-all-input hardness* [6] which requires all potential attackers to succeed only on finitely many inputs.

▶ **Definition 2.1** (Almost-all-input leakage-resilient hardness)**.** *Let $f : \{0,1\}^n \to \{0,1\}^n$ be a (multi-output) function. We say that f* is almost-all-input (T, ℓ)-leakage resilient hard *if for all T-time[2] probabilistic algorithms leak, A satisfying $\mathsf{leak}(x, f(x)) \leq \ell(|x|)$, for all sufficiently long strings x, $A(x, \mathsf{leak}(x, f(x))) \neq f(x)$ with probability $\geq 2/3$ (over their internal randomness).*

The notion of leakage-resilient *local* hardness will be useful for us. In the local hardness condition, we require no attacker A can produce each bit of $f(x)$ given the input x together with the coordinate. This allows us to consider attackers A that run in $|x|^\varepsilon$ time on input x.

▶ **Definition 2.2** (Almost-all-input leakage-resilient local hardness)**.** *Let $f : \{0,1\}^n \to \{0,1\}^n$ be a (multi-output) function. We say that f* is almost-all-input (T, ℓ)-leakage resilient t-local hard *for all T-time probabilistic algorithms leak satisfying $\mathsf{leak}(x, f(x)) \leq \ell(|x|)$, for all t-time probabilistic algorithms A, for all sufficiently long strings x, $A(x, \mathsf{leak}(x, f(x)))$* locally computes

[2] To simplify notation, we say that an algorithm $A(\cdot,\cdot)$ runs in time T if A runs in $T(n)$ time where n is the size of the first input.

$f(x)$ with probability at most $\geq 1/3$ (over their internal randomness), where we say that $A(x, \mathsf{leak}(x, f(x)))$ locally computes *$f(x)$ if for all $i \in \{0,1\}^{\log|x|}$, $i \leq |x|$, $A(x, \mathsf{leak}(x, f(x)), i) = f(x)_i$.*

We simply say that f is almost-all-input (T, ℓ)-leakage resilient local hard *if there exists $\varepsilon > 0$ such that f is almost-all-input (T, ℓ)-leakage resilient n^ε-local hard.*

We remark that we are decoupling the running time T of the leakage function and the running time t of the computing machine A. In addition, we only require hardness with respect to $|x|^\varepsilon$-time attackers A which can only read the first $|x|^\varepsilon$ bits of the string x.[3] As mentioned in the introduction, the notion of leakage-resilient local hardness enables us to capture the assumption that $\mathsf{E} \not\subseteq \mathsf{ioSIZE}(2^{\Omega(n)})$.

▶ **Lemma 2.3.** *If there exists a constant $\varepsilon > 0$ such that $\mathsf{E} \not\subseteq \mathsf{ioSIZE}(2^{\varepsilon n})$, then there exists a function $f : \{0,1\}^n \to \{0,1\}^n$ that is $(n^c, n^{\varepsilon/2})$-leakage resilient $n^{\varepsilon/2}$-locally hard for every $c \geq 1$.*

Proof. Let $g_n : \{0,1\}^n \to \{0,1\}$ be a E-computable function that requires circuits of size $> 2^{\varepsilon n}$ to compute (which is guaranteed to exist by the assumption). Consider the following multi-output function f such that for each $x \in \{0,1\}^*$,

$$f(x) = g_k(1)||\dots||g_k(2^k)||0||\dots||0$$

where $k = \lfloor \log|x| \rfloor$. Note that $f(x)$ can be computed in time $2^{O(k)} = 2^{O(\log|x|)} = |x|^{O(1)}$ so f is poly-time computable. Assume for contradiction that f is not $(n^c, n^{\varepsilon/2})$-leakage resilient $n^{\varepsilon/2}$-locally hard. Then there exist attackers (A, leak) such that $\mathsf{leak}(x, f(x))$ outputs at most $|x|^{\varepsilon/2}$ bits and $A(x, \mathsf{leak}(x, f(x)))$ computes $f(x)$ locally in time $|x|^{\varepsilon/2}$ for infinitely many x. For each such x and input length $k = \lfloor \log|x| \rfloor$, we will construct a $2^{\varepsilon k}$-size circuit C_k to compute g on input length k, which is a contradiction. Since $A(x, \mathsf{leak}(x, f(x)))$ locally computes $f(x)$ in time $|x|^{\varepsilon/2}$, there exists a leakage string $w \in \{0,1\}^{|x|^{\varepsilon/2}}$ and a random tape $r \in \{0,1\}^{|x|^{\varepsilon/2}}$ such that for each $i \in \{0,1\}^{\log|x|}$, $A(x, w, i; r)$ will compute $f(x)_i$ within time $|x|^{\varepsilon/2}$, and it follows that $A(x, w, i; r)$ will only read the first $|x|^{\varepsilon/2}$ bits of the string x; let x' denote the $|x|^{\varepsilon/2}$-bit prefix of x. Consider a circuit C_k having the strings x', w, r hardwired in it, and on input $i \in \{0,1\}^k$, it will compute $A(x, w, i; r)$. C_k is of size $O(|x|^{\varepsilon/2} \log|x|) \leq 2^{\varepsilon k}$ that computes g on input length k, which concludes the proof. ◀

In addition, we can also consider just "plain" (as opposed to leakage-resilient) hardness. As above, we also require the hardness condition holds on almost all inputs.

▶ **Definition 2.4** (Almost-all-input hardness)**.** *Let $f : \{0,1\}^n \to \{0,1\}^n$ be a (multi-output) function. We say that f* is almost-all-input T-hard *if for all T-time probabilistic algorithms A, for all sufficiently long strings x, $A(x) \neq f(x)$ with probability $\geq 2/3$ (over their internal randomness).*

2.2 Targeted Pseudorandom Generator

We consider the notion of a *targeted pseudorandom generator (targeted PRG)* [9]. Roughly speaking, a targeted pseudorandom generator G takes a seed along with a "target" string x as input, and we require that its output is indistinguishable from uniform by (computationally-bounded) distinguishers that additionally get the target x as input. In other words, G is just

[3] We assume that A is a standard Turing machine with each input on a separate tape, and we do not assume that A has oracle access to its inputs. This is a *weaker* hardness assumption than letting A have oracle access to the inputs.

like a normal PRG, but with the exception that both the PRG and the distinguisher get access to the auxiliary "target" string x, and we require security to hold for all strings x. Since we consider PRGs in the context of derandomization, we allow the running-time of the PRG to be larger than the running-time of the distinguisher.

▶ **Definition 2.5** (Targeted pseudorandom generator [9]). *Let $G : 1^n \times \{0,1\}^{\ell(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^n$ be a computable function. We say that G is an $T(n)$-secure $(\ell(n), m(n))$-targeted* pseudorandom generator (T-secure $(\ell(n), m(n))$-targeted PRG) *if for all deterministic attackers D that run in $T(n)$ time (where n is the length of its first input), for all sufficiently large $n \in \mathbb{N}$ and all strings $x \in \{0,1\}^{\ell(n)}$, it holds that*

$$|\Pr[s \leftarrow \{0,1\}^{m(n)} : D(1^n, x, G(1^n, x, s)) = 1] - \Pr[y \leftarrow \{0,1\}^n : D(1^n, x, y) = 1]| < \frac{1}{6}.$$

For any targeted PRG G, we say that G is $O(T(n))$-secure if for all constant $c > 0$, G is $(cT(n))$-secure. Note that the length of the target string $\ell(n)$ can be potentially larger than the running time bound of the distinguisher $T(n)$. In such cases, we only require security with respect to distinguishers D which can only read the first $T(n)$ bits of the target string. Note that this is a weaker security requirement than allowing D to have oracle access to the target string.

It is well-known that a (non-uniformly) secure PRG can derandomize prBPP. When considering a targeted uniformly-secure PRG, the same derandomization result still holds. This in essence follows by the standard proof (that non-uniformly secure PRG derandomize prBPP), but with an additional padding argument to deal with the "target"/auxiliary input.

▶ **Lemma 2.6** ([9]). *Assume that there exist constants $\sigma \geq 1, \theta \geq 1$ and a $O(n)$-secure $(n^\theta, \sigma \log n)$-targeted PRG G that runs in time $t(n) \geq n$. Then,* $\mathsf{prBPP} \subseteq \cup_{p,q \in \mathsf{poly}} \mathsf{prDTIME}[t(p(n))q(n)]$.

For the sake of completeness, we refer the reader to the full version for a detailed proof.

3 Derandomization and Leakage Resilient Hardness

In this section, we show a characterization between derandomizing prBPP and the existence of almost-all-input leakage resilient hard functions. Our result can be adapted to both the high-end and the low-end setting.

▶ **Theorem 3.1.** *There exists a constant $c \geq 1$ such that for all nice classes of functions $\mathcal{C}$, the following are equivalent.*

1. $\mathsf{prBPP} \subseteq \cup_{T \in \mathcal{C}} \mathsf{prDTIME}[T]$.
2. *The existence of a constant $\varepsilon > 0$, a function $T \in \mathcal{C}$, and an almost-all-input (n^c, n^ε)-leakage resilient locally hard function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic time T.*
3. *For all $d \geq 1$, there exist $T \in \mathcal{C}$ and an almost-all-input $(n^d, n - 3\log n)$-leakage resilient hard function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic time T.*

Proof. The implication (1) ⇒ (3) follows from Theorem 3.8 (stated and proved in Section 3.1). To show (2) ⇒ (1), we apply Lemma 3.9 (stated and proved in Section 3.2) to obtain a targeted PRG and (1) follows from Lemma 2.6. (3) trivially implies (2). ◀

We then state corollaries of Theorem 3.1 in both the high-end regime and the low end regime. To characterize derandomizing prBPP in polynomial time, we take the class $\mathcal{C}$ in Theorem 3.1 to be the class of polynomials $\mathsf{poly}(\cdot)$.

▶ **Corollary 3.2.** *There exists a constant $c \geq 1$ such that the following holds.* prBPP = prP *if and only if there exists an efficiently computable multi-output function f that is almost-all-input $(n^c, \sqrt{n})$-leakage resilient hard.*

To characterize derandomizing prBPP in subexponential time prSUBEXP $= \cap_{\varepsilon>0}$prDTIME$[2^{n^\varepsilon}]$, we consider the class $\mathcal{C}$ consisting of (all) time-constructible functions T such that $T(n)$ is smaller than 2^{n^ε} for all $\varepsilon > 0$[4], and we refer to a function f as being computable in subexponential time if f runs in time 2^{n^ε} for all $\varepsilon > 0$.

▶ **Corollary 3.3.** *There exists a constant $c \geq 1$ such that the following holds.* prBPP $\subseteq$ prSUBEXP *if and only if there exists an subexponential time computable multi-output function f that is almost-all-input $(n^c, \sqrt{n})$-leakage resilient hard.*

In addition, we note that the proof of Theorem 3.1 also yields the following amplification result for leakage resilient hardness.

▶ **Theorem 3.4.** *There exists a constant c such that if there exist a constant $\varepsilon > 0$ and an almost-all-input (n^c, n^ε)-leakage-resilient hard function computable in time $t(n)$, then for all $d \geq 1$, there exist polynomials p, q and an almost-all-input $(n^d, n - 3\log n)$-leakage-resilient hard function computable in time $t(p(n))q(n)$.*

Proof. The theorem follows from Lemma 3.9, Lemma 2.6, and Theorem 3.8. ◀

Besides amplifying leakage resilience, we observe that by combining the result of Chen and Tell with Theorem 3.1, we obtain a leakage-resilient hard function from a low-depth function with just plain hardness.

▶ **Theorem 3.5.** *There exists some c such that the following holds. If there exists a function f computable by polynomial-size logspace-uniform circuits with depth bounded by n^2 that is almost-all-input n^c-hard, then for any constant $d \geq 1$, there exists a polynomial-time computable almost-all-input $(n^d, n - 3\log n)$-leakage-resilient hard function.*

Proof. Chen and Tell [6] showed that the existence of such f implies prBPP = prP. The existence of a leakage-resilient hard function then follows from Theorem 3.1. ◀

3.1 Leakage Resilient Hardness from Derandomization

We proceed to constructing a multi-output function that is almost-all-input leakage resilient hard. Towards this, it is instructive to recall some ingredients from [9]. We first recall the definition of a prBPP search problem.

▶ **Definition 3.6** (prBPP search problem)**.** *Let R_{YES} and R_{NO} be two disjoint binary relations $\subseteq \{0,1\}^* \times \{0,1\}^*$. We say that $(R_{\mathsf{YES}}, R_{\mathsf{NO}})$ is a* prBPP search problem *if the following two conditions hold.*

1. *The decisional problem $(R_{\mathsf{YES}}, R_{\mathsf{NO}}) \in$ prBPP; that is, there exists a* PPT *algorithm V such that for every $(x, y) \in R_{\mathsf{YES}}$ it holds that $\Pr[V(x, y) = 1] \geq 2/3$, and for every $(x, y) \in R_{\mathsf{NO}}$ it holds that $\Pr[V(x, y) = 1] \leq 1/3$.*
2. *There exists a* PPT *algorithm A such that, for every $x \in S_{R_{\mathsf{YES}}}$, it holds that $\Pr[A(x) \in R_{\mathsf{YES}}(x)] \geq 2/3$, where $R_{\mathsf{YES}}(x) = \{y : (x, y) \in R_{\mathsf{YES}}\}$ and $S_{R_{\mathsf{YES}}} = \{x : R_{\mathsf{YES}} \neq \emptyset\}$*

[4] Note that this class is indeed nice since if $T(n) < 2^{n^\varepsilon}$ for all $\varepsilon > 0$, it holds that $T(p(n))q(n) < 2^{n^\varepsilon}$ for all $\varepsilon > 0$ and all polynomials p, q.

It has also been shown in [9] that there exists a deterministic search to decision reduction for prBPP by using techniques resembling the Conditional Expectation Method.

▶ **Theorem 3.7** (Search to decision reduction [9]). *For every* prBPP *search problem* $(R_{\mathsf{YES}}, R_{\mathsf{NO}})$*, there exists a binary relation* R *such that* $R_{\mathsf{YES}} \subseteq R \subseteq (\{0,1\}^* \times \{0,1\}^*) \backslash R_{\mathsf{NO}}$ *and solving the search problem of* R *is polynomial-time deterministically reducible to some decisional problem in* prBPP*.*

Now we return to showing the existence of a multi-output function that is hard to compute on almost all inputs with leakage assuming prBPP can be derandomized.

▶ **Theorem 3.8.** *If* $\mathsf{prBPTIME}[O(n)] \subseteq \mathsf{prDTIME}[t(n)]$*, then for any constant* $c \geq 1$*, there exists a function* $f : \{0,1\}^n \to \{0,1\}^n$ *running in time* $t(p(n))q(n)$ *(for some polynomials* p, q*) such that* f *is almost-all-input* $(n^c, n - 3\log n)$*-leakage resilient hard.*

Proof. We start by defining a prBPP-search problem which the task of constructing a leakage-resilient hard function can be reduced to. Consider any constant $c \geq 1$ and let $\ell(n) = n - 3\log n$. To construct an almost-all-input $(n^c, \ell(n))$-leakage resilient hard function, for each input $x \in \{0,1\}^n$, we need to find (uniformly) a string $f(x) = r$ such that r is hard to compute (for any n^c-time algorithms) given x and any (n^c-time) "side information" leaked from r. We observe that for *any* attacker/leakage functions g, leak – even non-computable g, leak – with high probability over r, r will satisfy the hardness with leakage condition w.r.t. g, leak.

▷ Claim 1. For any probabilistic algorithms leak, g, for all $n \in \mathbb{N}$, $\ell \leq n$, $x \in \{0,1\}^n$, with probability at most $6 \cdot 2^{-n+\ell+1}$ over random $r \in \{0,1\}^n$, it holds that

$$\Pr\left[|\mathsf{leak}(x,r)| \leq \ell \wedge g(x, \mathsf{leak}(x,r)) = r\right] \geq \frac{1}{6} \tag{1}$$

Proof. Consider any $n \in \mathbb{N}$, $x \in \{0,1\}^n$, and any probabilistic algorithm g. We will show that with probability at most $6 \cdot 2^{-n+\ell+1}$ over random $r \in \{0,1\}^n$, for any *deterministic* function leak' that outputs $\leq \ell$ bits, it holds that

$$\Pr[g(x, \mathsf{leak}'(x,r)) = r] \geq \frac{1}{6} \tag{2}$$

The proof of Claim 1 will directly follow from Equation 2 by noting that given any probabilistic leak, g, any $n \in \mathbb{N}, \ell \leq n$, $x \in \{0,1\}^n$, we can consider the deterministic leak' obtained by fixing the random tape of leak to be such that it maximize the number r's that satisfy Equation 1.

To show Equation 2, consider the set of "bad" r's: $B = \{r : \exists w, |w| \leq \ell, \Pr[g(x,w) = r] \geq 1/6\}$. For any $r \in \{0,1\}^n$, if there exists a function leak' that outputs $\leq \ell$ bits and $\Pr[g(x, \mathsf{leak}'(x,r)) = r] \geq \frac{1}{6}$, it follows that $r \in B$. We now bound $|B|$. Note that are at most $2^{\ell+1}$ strings w such that $|w| \leq \ell$, and for each such w, there can be at most 6 strings ouput by g with probability $\geq 1/6$; thus we have that $|B| \leq 6 \cdot 2^{\ell+1}$. It follows that the probability that a random r falls into B is at most $6 \cdot 2^{-n+\ell+1}$. ◁

Next, we note that if we only consider efficiently (and uniformly) computable g, leak, it suffices to consider attacker/leakage functions of description length no more than $\log n$. We can thus defined an prBPP-search problem that will enable us to find a "hard" r w.r.t. all such efficient attacker/leakage functions.

The BPP search problem. Let R_{YES} be a binary relation such that $(x, r) \in R_{\mathsf{YES}}$ if

1. $|x| = |r|$
2. For all probabilistic machines leak, g such that $|\mathsf{leak}| \leq \log n, |g| \leq \log n$, it holds that

$$\Pr\left[|\mathsf{leak}'(x, r)| \leq \ell(n) \wedge g'(x, \mathsf{leak}'(x, r)) = r\right] < \frac{1}{6} \tag{3}$$

where n denotes $|x|$, and leak' and g' denote "time-truncated" versions of leak, g that are only executed for n^c steps, where c is the constant in the lemma statement.

Let R_{NO} be a binary relation such that $(x, r) \in R_{\mathsf{NO}}$ if for at least one pair of leak and g with $|\mathsf{leak}|, |g| \leq \log n$, the above equation *with* $\frac{1}{6}$ *replaced by* $\frac{1}{3}$ does *not* hold.

We turn to showing that $(R_{\mathsf{YES}}, R_{\mathsf{NO}})$ is a prBPP search problem by presenting a verifying algorithm V and a solution finding algorithm A.

The search problem verifier. On input (x, r), the verifier V enumerates all probabilistic machines leak, g such that $|\mathsf{leak}|, |g| \leq \log n$. V estimates the value

$$p_{\mathsf{leak},g} = \Pr\left[|\mathsf{leak}'(x, r)| \leq \ell(n) \wedge g'(x, \mathsf{leak}(x, r)) = r\right]$$

by running the following experiment for sufficiently many times and computing the average acceptance probability. In each experiment, V emulates $\mathsf{leak}(x, r)$ for n^c steps, and emulates $g(x, \mathsf{leak}(x, r))$ for n^c steps. V accepts in this experiment if $g(x, \mathsf{leak}(x, r)) = r$. After estimating the average acceptance probability for each pair of leak and g, V outputs 1 if the estimated values of $p_{\mathsf{leak},g}$ are $< \frac{3}{12}$ for all pairs of leak, g. By the Chernoff bound and the Union bound, V will accept if $(x, r) \in R_{\mathsf{YES}}$ (and reject if $(x, r) \in R_{\mathsf{NO}}$) with very high probability.

The solution finder. We next construct a solution finding algorithm A such that $(x, A(x)) \in R_{\mathsf{YES}}$ with high probability for all x. On input x, A simply outputs a random string of the same length. For any fixed $x \in \{0,1\}^n$, by Claim 1 and a Union bound over the choice of leak and g, we conclude that $A(x)$ outputs a valid witness with probability at least $1 - 6n^2 \cdot 2^{-n+\ell(n)+1} \geq \frac{2}{3}$.

Constructing the hard function f. Finally, we show how to construct a function f that is hard to compute in the presence of any leakage, by making use of the prBPP search problem $(R_{\mathsf{YES}}, R_{\mathsf{NO}})$. By Theorem 3.7, there exists a binary relation R such that $R_{\mathsf{YES}} \subseteq R \subseteq (\{0,1\}^* \times \{0,1\}^*) \backslash R_{\mathsf{NO}}$ and solving the search problem of R is polynomial-time deterministic reducible to some decisional prBPP problem. This leads us to our construction of f. On input x, f solves the search problem of R and outputs a R-witness of x. We first show that f is almost-all-input $(n^c, \ell(n))$-leakage resilient hard. Consider any n^c-time algorithms leak and g satisfying $|\mathsf{leak}(x, f(x))| \leq \ell(|x|)$, all sufficiently large inputs $x \in \{0,1\}^n$ such that $|g| \leq \log n$ and $|\mathsf{leak}| \leq \log n$. Since R and R_{NO} are disjoint and f solves the search problem of R, $(x, f(x)) \notin R_{\mathsf{NO}}$ and this implies that

$$\Pr\left[g(x, \mathsf{leak}(x, f(x))) \neq f(x)\right] \geq \frac{2}{3}$$

We turn to proving that f runs in deterministic time $t(p(n))q(n)$ for some polynomials p, q, which will conclude our proof. Recall that f can be polynomial-time deterministically reduced to a decisional problem $\Pi \in \mathsf{prBPP}$. Since $\mathsf{prBPTIME}[O(n)] \subseteq \mathsf{prDTIME}[t(n)]$, by padding instances in Π so that the probabilistic algorithm for Π now runs in linear time in the length

of the padded instance, it follows that $\Pi \in \mathsf{prDTIME}[t(p'(n))]$ (for some polynomial p'). This, combined with the fact that the reduction runs in deterministic polynomial time, shows that f can be computed in deterministic time $t(p'(a(n)))b(n)$ for some polynomials a, b and the claim follows. ◀

3.2 Derandomization from Leakage Resilient Hardness

We turn our attention to the converse direction and we will show how to obtain a targeted PRG, which is later used to derandomize prBPP, from an almost-all-input leakage resilient hard function. Combining the result from the previous section, we will obtain a characterization between derandomization and leakage resilient hardness.

▶ **Lemma 3.9.** *There exists a constant $c \geq 1$ such that the following holds. Assume that there exist a constant $\varepsilon > 0$ and an almost-all-input (n^c, n^ε)-leakage resilient n^ε-locally hard function $f : \{0,1\}^n \to \{0,1\}^n$ computable in deterministic time $t(n) \geq n$. Then there exist constants $\sigma, \theta \geq 1$ and a $O(n)$-secure $(n^\theta, \sigma \log n)$-targeted PRG computable in time $t(n^\theta)\mathsf{poly}(n)$.*

The proof of Lemma 3.9 relies on the notion of *black-box PRG construction from a worst-case hard function f* [27, 29]. Roughly speaking, this notion of black-box PRG from a function f requires the existence of an efficient oracle algorithm that given (a) some fixed advice string, and (b) black-box access to any distinguisher for the PRG, is able to compute function f. Following, [12], we will here consider a strengthening of this notion of a black-box construction, simply referred to as *strongly black-box*, where also the advice string can be efficiently computed using black-box access to f.

▶ **Definition 3.10.** *Let $g : 1^n \times 1^m \times \{0,1\}^d \to \{0,1\}^m$ be a (deterministic) oracle algorithm, and let $k(\cdot)$ be functions. We say that g is a* k-reconstructive PRG construction *if there exist* PPT *oracle algorithms R, M such that for every $f : [n] \to \{0,1\}$ and $T : \{0,1\}^m \to \{0,1\}$, if*

$$|\Pr[T(g^f(1^n, 1^m, \mathcal{U}_d)) = 1] - \Pr[T(\mathcal{U}_m) = 1]| \geq \frac{1}{6}$$

then $M^{f,T}(1^n, 1^m)$ will output at most $k(n,m)$ bits such that for all $i \in [n]$,

$$R^T(M^{f,T}(1^n, 1^m), i) = f(i)$$

with probability at least $2/3$.

We next observe that the Sudan-Trevisan-Vadhan PRG [27] obtain by combining a locally list-decodable error correcting code [27] and the Nisan-Wigderson PRG construction [24] yields a strongly black-box construction of a PRG. We note that [29] previously argued that this construction is black-box; we here simply observe that the advice string needed can be efficiently computed.

▶ **Theorem 3.11** (Extending [27]; see also [29, Theorem 7.67])**.** *There exists a k-reconstructive PRG construction $g : 1^n \times 1^m \times \{0,1\}^d \to \{0,1\}^m$ such that for every $m \in \mathbb{N}$, $n \geq m$, $f : [n] \to \{0,1\}$ the following conditions are satisfied:*

1. *Explicitness: g^f is computable in uniform time $\mathsf{poly}(m,n)$.*
2. *Seed length: $d(n,m) = O(\log^2 n / \log m)$.*
3. *Reduction advice length: $k(n,m) = \mathsf{poly}(m, \log n)$.*

Since the proof follows standard techniques, we have deferred it to the full version.

Return to proving Lemma 3.9. We are now ready to prove Lemma 3.9 by relying on the above result.

Proof of Lemma 3.9. Consider any constant $\varepsilon > 0$.

A padding trick. In this proof, we will use a padding argument to make the leakage we need as small as it is required. Let m denote the output length of the targeted PRG that we hope to construct. Let n denote the input length of the multi-output function f. Let $\theta = O(1/\varepsilon) \in \mathbb{N}$ be a constant such that $\frac{1}{\theta}$ is sufficiently smaller than ε. In this proof, we usually assume that $n = \mathsf{poly}(m)$ and it holds that $n = n(m) = m^\theta$. In some cases depending on the context, m is defined w.r.t. n and it holds that $m(n) = \lfloor n^{1/\theta} \rfloor$ (and we can think of m as being sublinear in n).

Constructing the PRG. Let g be the k-reconstructive PRG obtained from Theorem 3.11, and let R, M be the algorithms associated with g (as in Definition 3.10). We will consider a function $G : 1^m \times \{0,1\}^{m^\theta} \times \{0,1\}^d \to \{0,1\}^m$. On input $(1^m, x, y)$ where $x \in \{0,1\}^{m^\theta}, y \in \{0,1\}^d$, the algorithm G proceeds in the following steps.

- G first computes $z = f(x)$. Let $n = m^\theta = |z|$.
- G outputs

$$G(1^m, x, y) = g^z(1^n, 1^m, y)$$

Note that the seed length of the PRG $d(n, m) = O(\log^2 n / \log m)$ so we can let σ be a constant such that $d = \sigma \log m$ and G is now a function of the form $1^m \times \{0,1\}^{m^\theta} \times \{0,1\}^{\sigma \log m} \to \{0,1\}^m$.

We claim that G is a $O(m)$-secure $(m^\theta, \sigma \log m)$-targeted PRG. Suppose not; then there exists a $O(m)$-time deterministic distinguisher D such that for infinitely many $m \in \mathbb{N}$, $n = m^\theta$, $x \in \{0,1\}^n$,

$$\left|\Pr[v \leftarrow \{0,1\}^{\sigma \log m} : D(1^m, x, G(1^m, x, v)) = 1] - \Pr[w \leftarrow \{0,1\}^m : D(1^m, x, w) = 1]\right| \geq \frac{1}{6} \quad (4)$$

(Note that D runs in time $O(m)$ so it is unable to read the whole string x.) We will prove that f can be computed locally in n^ε time with n^c-time computable leakage, for some constant c which we will fix later.

Computing f with leakage. We will construct a n^c-time algorithm leak, and a n^ε-time (local) algorithm A, where $\mathsf{leak}(x, f(x))$ will produce a $|x|^\varepsilon$-bit leakage and $A(x, \mathsf{leak}(x, f(x))$ will locally compute the function $f(x)$ on input x for infinitely many x (i.e., those inputs x on which Equation 4 holds). The algorithms A and leak will collaboratively proceed as follows. On input x, z, leak computes $n = |x|$ and $m = \lfloor n^{1/\theta} \rfloor$, and leak simply outputs $M^{z, D(1^m, x, \cdot)}(1^n, 1^m)$. We turn to constructing the algorithm A. On input x, the output of leak (denoted by a), and a bit index $i \in [n]$, A simply outputs $R^{D(1^m, x, \cdot)}(a, i)$.

Analyzing the reduction. We turn to analyzing the reduction. We first show that $A(x, \mathsf{leak}(x, f(x)), i)$ will indeed compute $f(x)_i$ for all $i \in [|x|]$ on infinitely many inputs x. This follows from the correctness of the distinguisher D, and the security of the reconstructive PRG g. In more detail, let us fix a (sufficiently long) string $x \in \{0,1\}^n$ w.r.t. which Equation 4 holds. Note that the distinguisher $D(1^m, x, \cdot)$ will also distinguish the reconstructive PRG $g^z(1^n, 1^m, \cdot)$, and therefore $A(x, \mathsf{leak}(x, f(x)), i)$ will output

$$R^{D(1^m, x, \cdot)}(M^{z, D(1^m, x, \cdot)}(1^n, 1^m), i)$$

which equals $z_i = f(x)_i$ with probability at least $2/3$. In addition, $\mathsf{leak}(x, f(x))$ is short and of length at most n^ε (due to our choice of θ). Since $\mathsf{leak}(x, f(x))$ contains the reduction advice for the reconstructive PRG g, and by Theorem 3.11 it is at most of length $\mathsf{poly}(m, \log n) = \mathsf{poly}(n^{1/\theta}, \log n)$, which is at most n^ε (since θ is picked to be much larger than $1/\varepsilon$).

We proceed to showing that leak runs in time n^c (for some sufficiently large universal constant c) and A runs in time n^ε. Note that leak simply invokes the algorithm M on input $1^n, 1^m$ (given z and $D(1^m, x, \cdot)$), M runs in polynomial time, and D runs in time $O(m)$. It follows that leak runs in time $\mathsf{poly}(n)$ and we can pick c to be large enough such that leak runs in time n^c. A will call the algorithm R on input (a, i), which (as argued above) is of length at most $n^{2/\theta}$. Since R runs in polynomial time, A runs in time $\mathsf{poly}(n^{2/\theta})$ which will be at most n^ε if we pick θ much larger than $1/\varepsilon$.

It remains to show that G runs in time $t(m^\theta)\mathsf{poly}(m)$. Note that it takes $t(m^\theta)$ time to compute $z = f(x)$, and the construction g runs in time polynomial in $n = m^\theta$. Thus, it follows that $G(1^m, x, \cdot)$ runs in time $t(m^\theta)\mathsf{poly}(m)$. ◀

4 Characterizing Derandomization of prMA

In this section, we present a characterization between derandomization and leakage-resilient hardness regarding prMA and prNP. In the non-deterministic setting, we need to consider a notion of leakage-resilient hard relations (generalizing the notion of leakage-resilient hard functions), which will be both sufficient and necessary to derandomize prMA. Due to space limit, we will defer proofs in this section to the full version.

For any relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ and any class of languages $\mathcal{C}$, we say that R is computable in $\mathcal{C}$ if there exists a language $L \in \mathcal{C}$ such that $(x, y) \in R$ iff $(x, y) \in L$.

▶ **Definition 4.1** (Leakage Resilient Hard Relation). *Let $R \subseteq \{0,1\}^* \times \{0,1\}^*$ be a relation such that for every $(x, y) \in R$, $|x| = |y|$. We say that R is* almost-all-input $(T(\cdot), \ell(\cdot))$-leakage resilient hard *if the following two conditions hold.*

- *(Non-triviality.) For every $x \in \{0,1\}^*$, $R(x) = \{y : (x, y) \in R\} \neq \emptyset$.*
- *(Leakage-resilient.) For all T-time probabilistic algorithms leak, A satisfying $\mathsf{leak}(x, y) \leq \ell(|x|)$, for all sufficiently long strings x, y satisfying $(x, y) \in R$, $A(x, \mathsf{leak}(x, y)) \neq y$ with probability $\geq 2/3$ (over their internal randomness).*

Now we are ready to state our characterization of derandomizing prMA.

▶ **Theorem 4.2.** *There exists a constant $c \geq 1$ such that for all nice classes of functions $\mathcal{C}$, all constants $0 < \varepsilon < 1$, the following are equivalent.*

1. $\mathsf{prMA} \subseteq \cup_{T \in \mathcal{C}} \mathsf{prNTIME}[T]$.
2. *The existence of a function $T \in \mathcal{C}$ and an almost-all-input (n^c, n^ε)-leakage resilient hard relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ computable in $\mathsf{NTIME}[T]$.*

Proof. The implication (1) ⇒ (2) follows from Theorem 4.3 (stated and proved below). To show (2) ⇒ (1), we apply Lemma 4.6 (stated and proved in below) to obtain a targeted PRG and (1) follows from Lemma 4.5. ◀

As demonstrated in the proof above, the proof Theorem 4.2 contains two parts. In the first part, we show that we can directly obtain a leakage-resilient hard relation from the assumption that $\mathsf{prMA} = \mathsf{prNP}$ without relying on a search-to-decision reduction.

▶ **Theorem 4.3.** *If $\mathsf{prMATIME}[O(n)] \subseteq \mathsf{prNTIME}[t(n)]$, then for any constant $c \geq 1$, there exists a relation R computable in $\mathsf{NTIME}[t(p(n))]$ (for some polynomials p) that is almost-all-input $(n^c, n - 3\log n)$-leakage resilient hard.*

In the second part of the proof, we prove the converse implication of Theorem 4.3. We rely on the following non-deterministic variant of a targeted PRG (where the PRG takes as input an additional witness whose validity can be checked by an verifier).

▶ **Definition 4.4** (Targeted non-deterministic pseudorandom generator)**.** *Let* $G : 1^n \times \{0,1\}^{\ell(n)} \times \{0,1\}^{\ell(n)} \times \{0,1\}^{m(n)} \to \{0,1\}^n$ *be a computable function. We say that* G *is an* $T(n)$-secure $(\ell(n), m(n))$-targeted non-deterministic pseudorandom generator (T-secure $(\ell(n), m(n))$-targeted NPRG) *if there exists a non-deterministic verifier* V *such that the following two conditions hold:*

- *For all sufficiently large* $n \in \mathbb{N}$*, for all* $x \in \{0,1\}^{\ell(n)}$*, there exists* $w \in \{0,1\}^{\ell(n)}$*,* $|w| = |x|$*, and* $V(1^n, x, w) = 1$*.*
- *For all deterministic attackers* D *that run in* $T(n)$ *time (where* n *is the length of its first input), for all sufficiently large* $n \in \mathbb{N}$ *and all strings* $x, w \in \{0,1\}^{\ell(n)}$ *satisfying* $V(1^n, x, w) = 1$*, it holds that*

$$|\Pr[s \leftarrow \{0,1\}^{m(n)} : D(1^n, x, G(1^n, x, w, s)) = 1] - \Pr[y \leftarrow \{0,1\}^n : D(1^n, x, y) = 1]| < \frac{1}{6}.$$

We say that a targeted NPRG G is computable in time T (for some function T) if G is computable in time T (with respect to the length of its first input) and there exists a verifier for G computable in non-deterministic time T (w.r.t. the length of its first input).

We then show that the notion of a targeted NPRG is indeed useful by proving that it can be used to derandomize prMA.

▶ **Lemma 4.5.** *Assume that there exist constants* $\sigma \geq 1, \theta \geq 1$ *and a* $O(n)$*-secure* $(n^\theta, \sigma \log n)$*-targeted NPRG* G *that is computable in time* $t(n) \geq n$*. Then,* $\mathsf{prMA} \subseteq \cup_{p,q \in \mathsf{poly}} \mathsf{prNTIME}[t(p(n))q(n)]$*.*

It remains to show that the existence of a leakage-resilient hard relation implies the existence of a targeted NPRG, which can be proved by generalizing Lemma 3.9 to allow non-deterministic computation.

▶ **Lemma 4.6.** *There exists a constant* $c \geq 1$ *such that the following holds. Assume that there exist a constant* $\varepsilon > 0$ *and an almost-all-input* (n^c, n^ε)*-leakage resilient hard relation* R *computable in* $\mathsf{NTIME}[t]$*. Then there exist constants* $\sigma, \theta \geq 1$ *and a* $O(n)$*-secure* $(n^\theta, \sigma \log n)$*-targeted NPRG* G_{N} *computable in time* $t(n^\theta)\mathsf{poly}(n)$*.*

References

1. Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In *Theory of cryptography conference*, pages 474–495. Springer, 2009.
2. László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993.
3. Manuel Blum and Silvio Micali. How to generate cryptographically strong sequences of pseudo-random bits. *SIAM Journal on Computing*, 13(4):850–864, 1984.
4. Zvika Brakerski and Yael Tauman Kalai. A parallel repetition theorem for leakage resilience. In *Theory of Cryptography Conference*, pages 248–265. Springer, 2012.
5. Lijie Chen, Ron D Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 13–23. IEEE, 2020.
6. Lijie Chen and Roei Tell. Hardness vs randomness, revised: Uniform, non-black-box, and instance-wise. *Electronic Colloquium on Computational Complexity*, 2021. URL: `https://eccc.weizmann.ac.il/report/2021/080/l`.

7 Don Coppersmith. Small solutions to polynomial equations, and low exponent rsa vulnerabilities. *Journal of cryptology*, 10(4):233–260, 1997.

8 Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302, 2008.

9 Oded Goldreich. In a world of P=BPP. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 191–232. Springer, 2011.

10 Oded Goldreich. Two comments on targeted canonical derandomizers. In *Electron. Colloquium Comput. Complex.*, volume 18, page 47, 2011.

11 Shuichi Hirahara. Non-disjoint promise problems from meta-computational view of pseudorandom generator constructions. In *35th Computational Complexity Conference (CCC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

12 R Impagliazzo and A Wigderson. Randomness vs. time: de-randomization under a uniform assumption. In *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No. 98CB36280)*, pages 734–743. IEEE, 1998.

13 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

14 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if e requires exponential circuits: Derandomizing the xor lemma. In *STOC '97*, pages 220–229, 1997.

15 Yuval Ishai, Amit Sahai, and David Wagner. Private circuits: Securing hardware against probing attacks. In *Annual International Cryptology Conference*, pages 463–481. Springer, 2003.

16 Valentine Kabanets. Easiness assumptions and hardness tests: Trading time for zero error. *Journal of Computer and System Sciences*, 63(2):236–252, 2001.

17 Oliver Korten. Derandomization from time-space tradeoffs. In *37th Computational Complexity Conference (CCC 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

18 Yanyi Liu and Rafael Pass. Characterizing derandomization through hardness of levin-kolmogorov complexity. In CCC, 2022.

19 Yanyi Liu and Rafael Pass. Leakage-resilient hardness vs randomness. *Electronic Colloquium on Computational Complexity*, 2022. URL: `https://eccc.weizmann.ac.il/report/2022/113/`.

20 Ueli M Maurer. Factoring with an oracle. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 429–436. Springer, 1992.

21 Silvio Micali and Leonid Reyzin. Physically observable cryptography. In *Theory of Cryptography Conference*, pages 278–296. Springer, 2004.

22 Cody Murray and Ryan Williams. Circuit lower bounds for nondeterministic quasi-polytime: an easy witness lemma for np and nqp. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 890–901, 2018.

23 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

24 Noam Nisan and Avi Wigderson. Hardness vs randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

25 Rafael Pass. Unprovability of leakage-resilient cryptography beyond the information-theoretic limit. In *SCN*, 2020.

26 Ronald L Rivest and Adi Shamir. Efficient factoring based on partial information. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 31–34. Springer, 1985.

27 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the xor lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

28 Roei Tell. Proving that prBPP= prP is as hard as proving that "almost NP" is not contained in P/poly. *Information Processing Letters*, 152:105841, 2019.

29 Salil P Vadhan. Pseudorandomness. *Foundations and Trends® in Theoretical Computer Science*, 7(1–3):1–336, 2012.

30 Andrew Chi-Chih Yao. Theory and applications of trapdoor functions (extended abstract). In *23rd Annual Symposium on Foundations of Computer Science, Chicago, Illinois, USA, 3-5 November 1982*, pages 80–91, 1982.

On the Impossibility of General Parallel Fast-Forwarding of Hamiltonian Simulation

Nai-Hui Chia
Rice University, Houston, TX, USA

Kai-Min Chung
Academia Sinica, Taipei, Taiwan

Yao-Ching Hsieh
University of Washington, Seattle, WA, USA

Han-Hsuan Lin
National Tsing Hua University, Hsinchu, Taiwan

Yao-Ting Lin
University of California at Santa Barbara, CA, USA

Yu-Ching Shen
Academia Sinica, Taipei, Taiwan

Abstract

Hamiltonian simulation is one of the most important problems in the field of quantum computing. There have been extended efforts on designing algorithms for faster simulation, and the evolution time T for the simulation greatly affect algorithm runtime as expected. While there are some specific types of Hamiltonians that can be fast-forwarded, i.e., simulated within time $o(T)$, for some large classes of Hamiltonians (e.g., all local/sparse Hamiltonians), existing simulation algorithms require running time at least linear in the evolution time T. On the other hand, while there exist lower bounds of $\Omega(T)$ *circuit size* for some large classes of Hamiltonian, these lower bounds do not rule out the possibilities of Hamiltonian simulation with large but "low-depth" circuits by running things in parallel. As a result, physical systems with system size scaling with T can potentially do a fast-forwarding simulation. Therefore, it is intriguing whether we can achieve fast Hamiltonian simulation with the power of parallelism.

In this work, we give a negative result for the above open problem in various settings. In the oracle model, we prove that there are time-independent sparse Hamiltonians that cannot be simulated via an oracle circuit of depth $o(T)$. In the plain model, relying on the random oracle heuristic, we show that there exist time-independent local Hamiltonians and time-dependent geometrically local Hamiltonians on n qubits that cannot be simulated via an oracle circuit of depth $o(T/n^c)$, where the Hamiltonians act on n qubits, and c is a constant. Lastly, we generalize the above results and show that any simulators that are geometrically local Hamiltonians cannot do the simulation much faster than parallel quantum algorithms.

2012 ACM Subject Classification Theory of computation → Quantum complexity theory

Keywords and phrases Hamiltonian simulation, Depth lower bound, Parallel query lower bound

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.33

Funding NH. Chia is supported by NSF award FET-2243659 and Google Scholar Award. KM. Chung's research is partially supported by NSTC QC project under Grant no. NSTC 111-2119-M-001-004- and the 2021 Academia Sinica Investigator Award (AS-IA-110-M02). HH. Lin is supported by NSTC QC project under Grant no. NSTC 111-2119-M-001-004- and MOST Grant no. 110-2222-E-007-002-MY3. YC. Hsieh and YC. Shen are supported by NSTC QC project under Grant no. NSTC 111-2119-M-001-004-. YT. Lin is partially supported by Executive Yuan Data Safety and Talent Cultivation Project (AS-KPQ-110-DSTCP). Part of the work was done when YC. Hsieh and YT. Lin were working at Academia Sinica.

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 33; pp. 33:1–33:45

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

1 Introduction

Simulating a physical system with a specified evolution time is an essential approach to study the properties of the system. In particular, given a Hamiltonian H that presents the physical system of interest and the evolution time t, the goal is to use some well-studied physical system as a simulator to implement e^{-iHt} when H is time-independent or $\exp_{\mathcal{T}}\left(-i\int_0^t H(t')dt'\right)$ for time-dependent H, where $\exp_{\mathcal{T}}$ denotes the time-ordered matrix exponential. Intuitively, a simulator simulates a Hamiltonian H step by step and thus requires time at least linear in t. On the other hand, if one can use a well-studied physical system (e.g., digital or quantum computers) to simulate the Hamiltonian of interest with time significantly less than the specified evolution time, it can significantly benefit our study of physics. Following this line of thought, a fundamental question for simulating Hamiltonians is:

Can a simulator simulate Hamiltonians in time strictly less than the evolution time?

This is called *fast-forwarding of Hamiltonians*. In this work, we investigate the possibility of achieving fast-forwarding of Hamiltonian using quantum computation.

It is known that quantum algorithms can fast-forward some Hamiltonians. Atia and Aharonov [3] showed that commuting local Hamiltonians and quadratic fermionic Hamiltonians with evolution time $t = 2^{\Omega(n)}$ can be *exponentially* fast-forwarded by quantum algorithms, where the Hamiltonian applies on n qubits. This result implies the existence of quantum algorithms that simulate the two classes of Hamiltonians in $\mathrm{poly}(n)$ time. Gu et al. [22] showed that more Hamiltonians could be exponentially or polynomially fast-forwarded, such as the exponential fast-forwarding for block diagonalizable Hamiltonians and polynomial fast-forwarding method for frustration-free Hamiltonians at low energies.

The existence of general fast-forwarding methods for Hamiltonians using quantum computers has also been studied. In particular, people investigated whether all Hamiltonians with some "succinct descriptions", such as local and sparse Hamiltonians, can be fast-forwarded. Berry et al. [6] proved no general fast-forwarding for all sparse Hamiltonians of n qubits for evolution time $t = n\pi/2$ by a reduction from computing the parity of a binary string. In particular, computing the parity of an n-bit string requires $\Omega(n)$ quantum queries, and they showed that any algorithm that simulates the corresponding Hamiltonian in time $o(n)$ will violate the aforementioned query lower bound of parity. Atia and Aharonov [3] further showed that if all 2-sparse row-computable[1] Hamiltonians with evolution time $2^{o(n)}$ can be simulated in quantum polynomial time, then **BQP** = **PSPACE**. In other words, the result in [3] rules out the possibility of exponential fast-forwarding for Hamiltonians with evolution time superpolynomial in the dimension under well-known complexity assumptions. Haah et al. [23] showed that there exists a piecewise constant bounded 1D time-dependent Hamiltonian[2] $H(t)$ on n qubits, such that any quantum algorithm simulating $H(t)$ with evolution time T requires $\Omega(nT)$ gates.

All these works, however, mainly considered lower bounds on the *number of gates* required for simulation, and it does not rule out the possibility that one can complete the simulation with time strictly less than t by using *parallelism*. Briefly, if many local gates in an algorithm operate on disjoint sets of input, then these gates can be applied together, and the efficiency of the algorithm is captured by the *circuit depth* instead of the number of gates. For instance, the result in [6] was based on the fact that the query complexity of parity is $\Omega(n)$ and thus,

[1] Given the row index, one can efficiently compute the column indices and values of the non-zero entries of the row.

[2] The Hamiltonian is 1D local, and there is a time slicing such that $H(t)$ is time-independent within each time slice. See [23] for the formal definition.

one needs $\Omega(t)$ queries to simulate the Hamiltonian evolution; however, if one runs t queries in parallel, it is possible to solve the parity problem with one layer of queries. Therefore, a direct translation of [6] does not rule out the possibility of constant depth simulation of the Hamiltonian evolution for time t by running $O(t)$ simulations in parallel.

Parallel runtime (i.e., the circuit depth in the quantum circuit model) could be another suitable notion for capturing the efficiency of the Hamiltonian simulation. Broadly speaking, any physically controllable and implementable system can be used as a simulator, so-called quantum analogue computing [17, 18]; instead of having one local interaction at each time step, a simulator that is realized by some physical system will have the whole system evolve together. From a computational perspective, a positive result of fast-forwarding Hamiltonians using parallel algorithms can imply that the simulation can be done in time strictly less than the specified evolution time given sufficient computational resources. In particular, if there exists an algorithm that simulates all Hamiltonians in time less than the simulation time t, we might be able to further reduce the runtime by recursively applying the algorithm with sufficient quantum resources. Hence, such algorithms can be a powerful tool for studying quantum physics. In fact, parallel quantum algorithms for Hamiltonian simulation have been studied and showed some advantages. Zhang et al. [39] presented a parallel quantum algorithm, whose parallel runtime (circuit depth) has a doubly logarithmic dependency on the simulation precision. Moreover, Haah et al. [23] showed that the circuit depth of their algorithm for simulating geometrically constant-local Hamiltonians can be reduced to $O(t \cdot \mathrm{polylog}(tn/\epsilon))$ by using ancilla qubits. In the last, choosing other physical systems similar to the target Hamiltonians as simulators is possible to gain advantages, which is the idea of quantum analogue computing.

We first explore the possibility of achieving fast-forwarding of Hamiltonians using parallel quantum algorithms, i.e., quantum algorithms that have circuit depth strictly less than the simulation time. We call this *parallel fast-forwarding.* Our first goal is to address the following question:

For all sparse or local Hamiltonians, do there exist quantum algorithms that simulate the Hamiltonians with circuit depth strictly less than the required evolution time?

Furthermore, we noticed that more general simulators (in addition to quantum circuit models) are widely considered for Hamiltonian simulation, such as quantum analogue computing. So, we are also wondering about the following question.

For all sparse or local Hamiltonians, does there exist a natural simulator that simulates the Hamiltonians with evolution time strictly less than the required evolution time?

1.1 Our Results

In the work, we give negative answers to the above questions. Roughly speaking, we show that under standard cryptographic assumptions, there exists Hamiltonians that cannot be parallelly fast-forwarded by quantum computers and any simulators that are geometrically local physical systems.

We define parallel fast-forwarding as follows:

▶ **Definition 1** (Parallel fast-forwarding). *Let* $\mathbf{H}$ *be a subset of all normalized Hamiltonian* ($\|H\| = 1$) *and* $\mathbf{H}_n$ *be the subset of Hamiltonian in* $\mathbf{H}$ *which acts on* n *qubits. We say that the set* $\mathbf{H}$ *can be* $(T(\cdot), g(\cdot), \varepsilon(\cdot))$*-parallel fast forwarded if there exists an efficient classical algorithm* $\mathcal{A}(1^n, t)$ *that outputs a circuit* $C_{n,t}$*, i.e.,* $\{C_{n,t}\}$ *is a uniform quantum circuit, such that for all* $n \in \mathbb{N}, t \leq T(n)$*,* $C_{n,t}$ *satisfies the following two properties.*

- *The circuit $C_{n,t}$ has depth at most $g(t)$.*
- *For all $H \in \mathbf{H}_n, |\psi\rangle \in \mathbb{C}^{2^n}$, the circuit $C_{n,t}(H, |\psi\rangle)$ (or $C^H_{n,t}(|\psi\rangle)$ under the oracle setting) has output state that is $\varepsilon(n)$ close to the Hamiltonian evolution outcome $e^{-iHt}|\psi\rangle$.*

In other words, there exists uniform quantum circuit $C_{n,t}$ such that for every Hamiltonian $H \in \mathbf{H}_n$, the evolution of H to time t up to some predetermined time bound $T(\cdot)$ can be simulated by C.

Compared to [3], Definition 1 focuses on C's circuit depth instead of the number of gates and requires the depth of C to be smaller than t rather than being $\mathrm{poly}(n)$. In particular, when $t = \mathsf{superpoly}(n)$, the definition in [3] can only be satisfied by a circuit C that has gate number superpolynomially smaller than t, and C that has circuit depth slightly less than t can satisfy Definition 1. Therefore, we can also interpret the no fast-forwarding theorem in [3] as refuting the possibility of achieving Definition 1 with gate number (and also circuit depth) superpolynomially smaller than $t = \mathsf{superpoly}(n)$ for a specific family of Hamiltonians. However, given that negative result, one might ask the following question:

Can we achieve parallel fast-forwarding with $g(t)$ slightly smaller than t, such as $g(t) = \sqrt{t}$?

In this work, we address the aforementioned question and show impossibility results for parallel fast-forwarding with circuit depth $g(t)$ slightly smaller than t for local or sparse Hamiltonians. Our first result is an unconditional[3] result under the oracle model.[4]

▶ **Theorem 2** (No parallel fast-forwarding for sparse Hamiltonians relative to random permutations, simplified version of Theorem 37)**.** *Relative to a random permutation oracle over n-bit strings, for any polynomial $T(\cdot)$, there exists a family of time-independent sparse Hamiltonians $\mathbf{H}$ such that $\mathbf{H}$ cannot be $(T(\cdot), g(\cdot), \varepsilon(\cdot))$-fast forwarded for some $g = \Omega(t)$ and $\varepsilon = \Omega(1)$.*

To obtain no fast-forwarding result in the standard model,[5] we rely on cryptographic assumptions that provide hardness against low-depth algorithms. We assume the existence of *iterative parallel-hard functions*, formally defined in Definition 43. Roughly speaking, an iterative parallel-hard function is a function of the form $f(k, x) = g^{(k)}(x) := \underbrace{g(g(\ldots g(x)))}_{k \text{ times}}$, such that g is efficiently computable (by some circuit of size s), but $g^{(k)}(x)$ is not computable for circuits with depth much less than k.

With such a cryptographic assumption, we obtained the following two no-fast-forwarding theorems under the standard model.

▶ **Theorem 3** (No parallel fast-forwarding for local Hamiltonians, simplified version of Theorem 45)**.** *Assuming the existence of iterative parallel-hard functions with size parameter $s(n)$, then for every polynomial $T(n)$, there exists a family of time-independent local Hamiltonians $\mathbf{H}$ such that $\mathbf{H}$ cannot be $(T(\cdot), g(\cdot), \varepsilon(\cdot))$-fast forwarded for some $g = \Omega(t/s(n))$ and $\varepsilon = \Omega(1)$.*

▶ **Theorem 4** (No parallel fast-forwarding for time-dependent geometrically local Hamiltonians, simplified version of Theorem 46)**.** *Assuming the existence of iterative parallel-hard functions with size parameter $s(n)$, then for every polynomial $T(n)$, there exists a family of time-dependent geometrically local Hamiltonian H such that $\mathbf{H}$ cannot be $(T(\cdot), g(\cdot), \varepsilon(\cdot))$-fast forwarded for some $g = \Omega(t/ns(n))$ and $\varepsilon = \Omega(1)$.*

---

[3] That is, the result holds without making any computational assumptions.

[4] By the oracle model we mean that the algorithm can only access the Hamiltonian by making (quantum) queries to the oracle that encodes the description of the Hamiltonian. See Section 8 for the definition.

[5] By the standard (plain) model we mean that the algorithm is given the classical description of the Hamiltonian as input, which is the standard setting of the Hamiltonian simulation problem. Moreover, there is no oracle that can be accessed by algorithms. We will use the terms "standard model" and "plain model" interchangeably throughout this work.

Some loss in parameters are hidden in Theorem 3 and Theorem 4. Readers are referred to the full theorems in Section 9 for precise parameters.

We note that the existence of parallel-hard functions with an iterative structure is widely used in cryptography. Our definition of iterative parallel-hard functions adapts from the iterated sequential function proposed by Boneh et al. [9]. Functions of this form play a crucial role in the recent construction of verifiable delay functions (VDF) [19, 32, 35]. In contrast to its wide usage, there have not been many proposals on candidates for such iterative hard functions. Iterative squaring [33], which is probably the most widely used candidate, is not hard against quantum circuits. There are some recent attempts toward constructing iterated quantum-hard functions from isogenies [12, 20], but these assumptions are much less well-studied.

As a concrete instantiation of our iterated parallel-hard function, we adopted a hash chain, which is also widely assumed to be hard to compute within low depth. In Section 6, we justify the quantum parallel hardness of the hash chain by showing a depth lower bound of computing the hash chain in the quantum random oracle model [10].

Our results in Theorem 2, Theorem 3, and Theorem 4 imply that no quantum algorithm can simulate certain families of local or sparse Hamiltonians with circuit depth polynomially smaller than t. For instance, suppose $t = n^c$ for some constant c and $s(n) = n^2$, then by Theorem 3, no quantum algorithm can simulate the local Hamiltonians with circuit depth smaller than t^{c-2}.

Since local Hamiltonians are sparse, Theorem 3 also implies no parallel fast-forwarding of sparse Hamiltonians in the standard model. Finally, Theorem 4 and Theorem 3 are incomparable due to the fact that the Hamiltonians in Theorem 4 are time-dependent and the depth lower bound has a factor of n.

It is worth noting that the results above show no parallel fast-forwarding when using "quantum circuits" as simulators, which does not directly imply hardness results when considering other physical systems as simulators. Especially, choosing physical systems that are similar to the Hamiltonians to be simulated is possible to gain advantages, and physical systems naturally evolve the whole system together instead of applying local operators one by one. Therefore, it is nontrivial whether similar results hold for other simulators. Fortunately, we are able to generalize Theorem 4 and Theorem 3 to show that natural simulators that are geometrically local Hamiltonians cannot do much better than quantum circuits.

▶ **Theorem 5** (No fast-forwarding for local Hamiltonians with natural simulators, simplified version of Corollary 55)**.** *Assuming the existence of iterative parallel-hard function with size parameter $s(n)$, then for every polynomial $T(n)$, there exists a family of time-independent local Hamiltonians $\mathbf{H}$ over $\widetilde{O}(n)$ qubits satisfying the following. For any geometrically constant-local Hamiltonian H_B acting on $\mathrm{poly}(n)$ qubits, using H_B to simulate any $H_A \in \mathbf{H}$ for any evolution time $t \in [0, s(n)T(n)]$ needs an evolution time at least $(t/2s(n) - O(s(n)))/\mathrm{polylog}(tn)$.*

▶ **Theorem 6** (No fast-forwarding for geometrically local Hamiltonians with natural simulators, simplified version of Corollary 54)**.** *Assuming the existence of iterative parallel-hard functions with size parameter $s(n)$, then for every polynomial $T(n)$, there exists a family of time-dependent geometrically local Hamiltonians $\mathbf{H}$ over $\widetilde{O}(n)$ qubits satisfying the following. For any geometrically constant-local Hamiltonian H_B acting on $\mathrm{poly}(n)$ qubits, using H_B to simulate any $H_A \in \mathbf{H}$ for any evolution time $t \in [0, ns(n)T(n)]$ needs an evolution time at least $\left(\frac{t}{ns(n)} - O(ns(n)) - \mathrm{polylog}(n)\right)/\mathrm{polylog}(tn)$.*

2 Technical Overview

The main idea

Our idea is to reduce some tasks that have a circuit or query depth lower bounds (i.e., parallel-hard problems) to simulating specific Hamiltonians with evolution time t, such that the existence of parallel fast-forwarding of the Hamiltonians will contradict the circuit depth lower bound and also violate the parallel hardness of the task. For instance, one can reduce parity, which is not in $\mathbf{QNC}^0$ (the class of all constant-depth bounded fan-in circuits), to simulate a corresponding Hamiltonian H with some time t, such that e^{-iHt} outputs the parity of the input. Along this line, if e^{-iHt} can be implemented by a constant-depth quantum circuit, we can compute parity – this violates the quantum circuit lower bound on parity! Following the same idea, one can also derive some no-go results for parallel fast-forwarding from unstructured search, where the k-parallel quantum query complexity is $\Theta(\sqrt{N/k})$, where k-parallel means each "query layer" can have k queries in parallel [24, 36].

However, there are several challenges: First, those above-mentioned parallel-hard problems can be solved in depth smaller than the input size. This could result in a Hamiltonian simulation in which the evolution time is smaller than the number of qubits. Although this might still lead to an impossibility result for parallel fast-forwarding of an $o(n)$ evolution time, parallel fast-forwarding algorithms for such a short evolution time seem not that useful. In fact, to the best of our knowledge, it is not easy to find a problem that can be computed in quantum polynomial time while having a quantum depth strictly greater than the input size using polylog(n) parallel queries. So, one technical contribution of our work is finding such problems and proving their quantum depth.

Second, finding appropriate reductions from the parallel-hard problems to Hamiltonians of our interest and preserving the input size and the quantum depth is also challenging. Note that we are focusing on sparse or local Hamiltonians with evolution time, a polynomial in the number of qubits. One intuitive approach is trying the circuit-to-Hamiltonian reduction in [25, 30]. Briefly, the reduction uses a t-depth circuit on n qubits to simulate a local Hamiltonian on $n + t$ qubits with time t, where the additional t qubits are for the "clock register". This, as mentioned above, has an evolution time smaller than the number of qubits. In this work, we find reductions that map a d-depth n-qubit quantum computation with $d = \text{poly}(n)$ to a local or sparse Hamiltonian with the number of qubits and evolution time "close to" n and t respectively.

Another challenge is that we need the parallel-hard problem as an iterative structure to show no parallel fast-forwarding theorems. More specifically, our goal is to prove that some Hamiltonians cannot be parallel fast-forwarded with *any evolution time in the specified range*. Therefore, we might need a sequence of parallel-hard problems such that there are corresponding parallel-hard problems *for all* t in the range. In addition, given a parallel-hard problem with an iterative structure, it is not trivial how to reduce it to one Hamiltonian H with different evolution times t such that simulating H for different t gives the corresponding answers.

Parallel hardness of the underlining assumptions

One candidate for parallel-hard problems with an iterative structure of our purpose is the *hash chain*. Roughly speaking, let $\mathcal{X}$ be a finite set and $h : \mathcal{X} \to \mathcal{X}$ be a hash function. An s-chain of h is a sequence $x_0, x_1, \ldots, x_s \in \mathcal{X}$ such that $x_{i+1} = h(x_i)$ for any $i \in [s-1]$. Given quantum oracle access to h, the goal of the algorithm is to find an s-chain. Classically, it was

proven that classical algorithms require query depth of at least s to output an s-chain with constant probability. A similar result also holds for quantum algorithms that make quantum queries to the hash function [16]. Along this line, the hash chain seems ideal for our purpose because s can be a polynomial in $\log(|\mathcal{X}|)$ and has the iterative parallel hardness.

However, a hash function is generally irreversible, and this fails standard approaches for reducing the problem to Hamiltonian simulation. Briefly, one encodes h as a Hamiltonian H such that evaluating h is equivalent to applying e^{-iH}. Since e^{-iH} is a unitary that is reversible, evaluating h also needs to be reversible. Here, we give *permutation chain* and *twisted hash chain* that are iteratively parallel-hard and the underlining function is reversible. However, the reversibility imposes another challenge, as the ability to query the inverse of the permutation breaks the known composed oracle techniques used to prove the hardness of hash chain [16][6]. Therefore we tailored a two-step-hybrid argument to prove the hardness of the random permutation chain with the ability to query the inverse of the permutations.

Note that for oracle lower bounds of parallel query algorithms, while [24] gives optimal bounds by generalizing the adversary method, it is notoriously hard to find the suitable adversary matrices. Therefore we derive the query lower bounds for our problems by crafting a hybrid argument and using the compressed oracle technique [38] respectively.

2.1 No parallel fast-forwarding for sparse Hamiltonians relative to random permutation oracle

We first introduce the permutation chain and demonstrate how to prove Theorem 2 via the *graph-to-Hamiltonian reduction* based on the permutation chain. This shows no parallel fast-forwarding for sparse matrices relative to a random permutation oracle.

Permutation chain

One of the reversible parallel-hard problem we formulated is the *permutation chain*. In this problem, we are given as inputs q permutations of $N := 2^n$ elements $\Pi_1, \Pi_2, \ldots, \Pi_q$.[7] Let $\mathrm{S}\Pi$ be the unitary that enables one to query to each Π_i and their inverses in superposition. Let $\bar{x}_1 = 1$ and $\bar{x}_{i+1} = \Pi_i(\bar{x}_i)$ so that $\bar{x}_{q+1} = \Pi_q(\ldots\Pi_2(\Pi_1(1)))$. With q queries to $\mathrm{S}\Pi$, it is easy to calculate $\bar{x}_i$, while we prove that it is only possible to calculate $\bar{x}_i$ with probability $O(q\sqrt{k/N})$ using $\lfloor (q-1)/2 \rfloor$ k-parallel queries[8] to $\mathrm{S}\Pi$. Therefore if we have $q, k = O(\mathrm{polylog}(N))$, the success probability is negligible in n, even when k is larger than q and having access to the inverses of $\Pi_1, \Pi_2, \ldots, \Pi_q$.

To bound the success probability, we employed a two-step hybrid. First, we show that we can replace each $\mathrm{S}\Pi$ with $\mathrm{S}\tilde{\Pi}$. $\mathrm{S}\tilde{\Pi}$ is a set of functions that return zeros almost everywhere except at $\{\bar{x}_i\}$, where they behave the same as $\mathrm{S}\Pi$ (see Figure 2(a)(b)). We prove that we can approximately simulate one call to a random $\mathrm{S}\Pi$ with two calls to $\mathrm{S}\tilde{\Pi}$. Now, $\mathrm{S}\tilde{\Pi}$ looks like a constant zeros function, we can erase some of its values without getting caught. In the second step, we show that we can release the $\tilde{\Pi}_i$'s on a finely controlled schedule, with only negligible change in the output probability. Define $\Pi^{\perp}$ to be a constant zero function. Define $\mathrm{S}\tilde{\Pi}_\ell$ to be the unitary corresponding to $\tilde{\Pi}_1, \tilde{\Pi}_2, \ldots, \tilde{\Pi}_\ell, \Pi^{\perp}, \ldots$, i.e., all but the first ℓ permutations are erased (see Figure 1). We show that if we replace the first k-parallel queries

[6] One can use the technique in [37] to convert random permutations to random functions, but the conversion only works when the algorithm has no access to the inverse oracle.
[7] They can be viewed as a special case of one permutation of qN elements.
[8] k-parallel means each "query layer" can have k queries in parallel

of $\mathrm{S}\tilde{\Pi}$ with $\mathrm{S}\tilde{\Pi}_1$, second k-parallel queries of $\mathrm{S}\tilde{\Pi}$ with $\mathrm{S}\tilde{\Pi}_2$, third k-parallel queries of $\mathrm{S}\tilde{\Pi}$ with $\mathrm{S}\tilde{\Pi}_3$, etc, we can only be caught with negligible probability. Intuitively, this is because while we are at the i-th query layer, it is hard to find any non-zero values of $\tilde{\Pi}_{i+1}, \ldots, \tilde{\Pi}_q$. Therefore, if an algorithm only makes $q-1$ queries to $\mathrm{S}\tilde{\Pi}$, we can replace the queries with $\mathrm{S}\tilde{\Pi}_1, \mathrm{S}\tilde{\Pi}_2, \ldots, \mathrm{S}\tilde{\Pi}_{q-1}$. It is impossible to find $\bar{x}_{q+1}$ with non-negligible probability since these oracles do not have information of $\tilde{\Pi}_q$.

Graph-to-Hamiltonian reductions

The purpose of graph-to-Hamiltonian reduction is using quantum walk on a line [14] to solve the permutation chain. Briefly, we use a graph to encode the permutation chain and let Hamiltonian H be the adjacency matrix that represents the graph. Then, the time evolution operator e^{-iHt} helps to find the solution of permutation chain. Therefore, a low-depth Hamiltonian simulation algorithm for H could result in breaking the hardness of permutation chain. This gives our first impossibility result of parallel fast-forwarding for sparse Hamiltonians.

Let $\Pi_1, \Pi_2, \ldots \Pi_L$ be L permutations over N elements. We use a graph with $N(L+1)$ vertices in which each vertex labelled by (j, x) to record the permutation chain, where $j \in \{0, 1, \ldots, L\}$ and $x \in [N]$. The vertices (j, x) and $(j+1, x')$ are adjacent if and only if $x' = \Pi_{j+1}(x)$. The construction of the graph has followings properties. First, the graph consists of N disconnected line because each Π_j is a permutation. Second, each vertex (q, x) that connects to $(0, x_0)$ satisfies $x_q = \Pi_q(\Pi_{q-1} \cdots (\Pi_1(x_0))$. To solve the permutation chain problem, we start from the vertex $(0, x_0)$ and walk along the connected line. When stopping at a vertex (q, x_q), the pair (x_q, x_0) would be a solution of permutation chain. It is obvious that the adjacency matrix of the corresponding graph is *sparse*. We let the Hamiltonian H determining the dynamics of the walk be the adjacency matrix of the graph, and our goal is to find (x_q, x_0) by simulating e^{-iHt} given *sparse access* to H.

There are two main challenges for building such a reduction: First, we need to implement the sparse oracle access to the corresponding Hamiltonian. This requires oracle access to the permutation and inverse permutation oracle. More specific, we need to implement two oracles that are used in the Hamiltonian simulation algorithm to execute the quantum walk. The first one is the entry oracle $\mathcal{O}_H$, which answers the element value of H when queried on the matrix index. The second one is the sparse structure oracle $\mathcal{O}_L$, which answers the indices of the nonzero entries when queried on the row index. To implement $\mathcal{O}_H$, it is equivalent to checking if two vertices (j, x) and $(j+1, x')$ are adjacency. It can be done by querying $\Pi_{j+1}(x)$. To implement $\mathcal{O}_L$, it is equivalent to finding the vertices that are adjacent to (j, x). Finding $(j+1, x')$ adjacent to (j, x) can be done by querying $\Pi_{j+1}(x)$, but finding $(j-1, x'')$ needs to query $\Pi_j^{-1}(x)$. Hence, we need to consider the security of permutation chain when the inversion oracle Π_j^{-1} is given to the adversary. We bypass this challenge by showing that the our permutation chain is secure against quantum adversaries even if inverse permutation oracle is given as we previously discussed.

Second, we need to show that the simulation algorithm is able to walk fast enough so that simulating H for evolution time close to the length of the chain gives the solution to the permutation chain. To be more precise, we aim to design the system such that after walking for time t, it reaches the vertex further than t with high probability. Recall that H determining the dynamics of the walk is the adjacency matrix of the graph corresponding to the permutation chain. We observe that for such quantum walk system, it indeed reaches some points beyond t for the walking time t with high probability. At any time t, the system is described by the quantum state $e^{-iHt}|0, x_0\rangle$. The probability of stopping on the vertex

(q, x_q) at time t is $P(q) = \left|\langle q, x_q|e^{-iHt}|0, x_0\rangle\right|^2$. We have $\left|\langle q, x_q|e^{-iHt}|0, x_0\rangle\right| = qJ_q(2t)/t$ for $t \in [0, L/2]$, where $J_q(\cdot)$ is the q-th order Bessel function [14]. By the properties of Bessel function, we show that $\sum_{q=\lceil t\rceil}^{L} P(q) = O(1)$, which means that the probability of stopping at a vertex (l, x_l) such that $l > t$ is high. As a result, it breaks the hardness of permutation chain if e^{-iHt} can be implemented with $o(t)$ queries.

2.2 No parallel fast-forwarding for (geometrically) local Hamiltonians in the plain model

To show no fast-forwarding of (geometrically) local Hamiltonians in the plain model, the combination of the permutation chain and the graph-to-Hamiltonian reduction used in Section 2.1 might be insufficient. First, it is unclear how to instantiate random permutation oracle. In addition, even if we can translate the permutation chain to a parallel-hard quantum circuit in the plain model, the graph-to-Hamiltonian reduction inherently provides sparse oracle access to the Hamiltonian from oracle access to the permutation chain. However, we need to have the full classical descriptions of each local term for simulating local Hamiltonians.

Observing these difficulties, we introduce the twisted hash chain and the circuit-to-Hamiltonian reduction for proving Theorem 3 and Theorem 4.

Twisted hash chain

In order to implement a reversible operation (or a permutation), we follow the idea of the Feistel network [29]. Roughly speaking, the Feistel network is an implementation of block ciphers by using cryptographic hash functions. By means of chaining quantum query operators as in Figure 3, the outputs in each layer satisfy $x_i = H(x_{i-1}) \oplus x_{i-2}$. Therefore, we can think of it as a "quantum version" of the Feistel networks. Informally, the goal of the algorithm is to output the head and tail of a chain of length $q + 1$ by using at most q depth of queries.

For proving the parallel hardness, we use the compressed oracle technique by Zhandry [38]. In particular, the analysis is undergone in the framework of Chung et al. [16] where they generalize the technique to the parallel query model. Our proof is inspired by the parallel hardness of the standard hash chain proven in [16]. For technical reasons, the challenge is the following: in the twisted hash chain problem, the algorithm is not required to output *all* elements of the chain and their hash values. Therefore, we cannot directly apply the tools provided in [16]. In addition, we cannot simply ask the algorithm to spend extra queries for outputting the hash values since this would lead to a trivial bound (we call the extra queries for generating the whole chain the "verification" procedure). Instead, we need a more fine-grained analysis of the verification procedure. First, we notice that since $x_i = H(x_{i-1}) \oplus x_{i-2}$, the verification requires *sequential* queries. Therefore, unlike Theorem 5.9 in [16] where the verification procedure only requires parallel queries, the analysis for our purpose is more involved.

We bypass the aforementioned issue by reduction. Suppose there is an algorithm $\mathcal{A}$ outputs x_0, x_q, x_{q+1} such that $x_0, \ldots, x_{q+1}$ form a $(q + 1)$-chain by making q k-parallel queries. Then we can construct a reduction $\mathcal{B}$ which first runs $\mathcal{A}$ and obtain x_0, x_q, x_{q+1}. Next, $\mathcal{B}$ starts with x_0, x_{q+1} and queries each element of the chain iteratively *in parallel* until approaching x_{q-1}, x_{2q}. If $\mathcal{A}$ successfully outputs a $(q + 1)$-chain, then it implies that $\mathcal{B}$ also outputs the *complete* $(2q + 1)$-chain with hash values but $H(x_{2q+1})$ by making a total of $2q$ k-parallel queries. As a result, it remains to analyze the success probability of making the last additional query on x_{2q+1}. In this way, it significantly simplifies the proof.

Circuit-to-Hamiltonian reductions

For our results in the plain model, we leverage the power of the random oracle heuristic. From the parallel hardness of twisted hash chain, we can obtain a heuristically parallel-hard circuit that preserves the iterative structure. Evaluation of this circuit to large depth directly translates to computing a hash chain of large length, which is assumed to be hard for low depth circuit. To translate the hardness to a no parallel fast-forwarding result, we embed the computation of the circuit to a Hamiltonian via two different approaches.

To embed circuit computation to a time-independent Hamiltonian, we use the technique from Nagaj [30], which demonstrate how to transform a circuit computation with size T to a Hamiltonian evolution problem of time $O(T \log T)$. In our work, we make two major modification upon Nagaj's technique. First, we observed that Nagaj's technique fits well with our iterated structure of circuit. At a high level, simulating Hamiltonian obtained from Nagaj's compiler can be interpreted as a quantum walk on a line, where each point on the line correspond to a computation step/gate of the circuit. Again by the detailed analysis on Bessel function that we used in the graph-to-Hamiltonian reduction, we observe that we can obtain a "depth $O(t)$" intermediate state of computing C by evolving H for time $O(t)$. This not only gives a better fast-forward lower bound, but also allows us to obtain a Hamiltonian that is hard to fast-forward on *every* evolution time within time bound T. Second, Nagaj's construction gives a Hamiltonian of $O(n + T)$-qubits, where n is the circuit input size and T is the circuit size. This is an issue because it restricts our no fast-forwarding results to evolution times small than the Hamiltonian size. We overcome this by introducing a new design for the clock state via the Johnson graph. Our restructured clock state allows a fine-grained tradeoff between the locality parameter and the Hamiltonian size.

For our second construction, we achieve the geometrically local property with the power of time-dependent Hamiltonians. Our idea is to use the piecewise-time-independent construction from [23], in which simulating the Hamiltonian for each time segment on the initial state behaves equivalently to applying a gate on the state. We take one step further by transforming our circuit to contain gates operating on neighboring gates only. This gives us a geometrically 2-local Hamiltonian which is hard to fast-forward. Combined with the algorithm that simulates geometrically local Hamiltonians also by [23], our result tightens the gap between upper bounds and lower bounds to a small polynomial in qubit number n.

▶ Remark 7. Two things worth to be noted for the two approaches in Section 2.1 and Section 2.2:

- If one can instantiate random permutations by hash functions or other algorithms without using keys, one can obtain Theorem 3 and Theorem 4 by combining the permutation chain and the circuit-to-Hamiltonian reduction.
- The combination of the twisted hash chain and the circuit-to-Hamiltonian reduction can give no parallel fast-forwarding for Hamiltonians in the random oracle model. This is similar to Theorem 2; however, Theorem 2 using the permutation chain and the graph-to-Hamiltonian reduction provides a better size of the Hamiltonians. In particular, the Hamiltonian in Theorem 2 has the number of qubits independent of the evolution time, while the Hamiltonians given from the circuit-to-Hamiltonian reduction has the number of qubits that is poly-logarithmic in the evolution time.

3 Open Questions

In this work, we showed that the existence of a parallel-hard problem with an iterative structure implies no parallel fast-forwarding of sparse and (geometrically) local Hamiltonians under cryptographic assumptions. Along this line, the first question that is natural to ask is whether there exist more Hamiltonians that have succinct descriptions and cannot be parallelly fast-forwarded under other computational assumptions.

We are also wondering whether the existence of parallel-hard problems with an iterative structure is equivalent to no parallel fast-forwarding. This is equivalent to proving or disproving that no parallel fast-forwarding results in parallel-hard problems with an iterative structure. Intuitively, One can show that the existence of Hamiltonians that cannot be parallelly fast-forwarded implies some quantum circuits that have no smaller circuit depth. This follows from the fact that if one can implement a quantum circuit with a depth smaller than the quantum simulation algorithm for the Hamiltonian, one can achieve parallel fast-forwarding. However, this task asks the algorithm to output quantum states close to $e^{-iHt}|\psi\rangle$ and thus is not a "classical computational problem" as parallel-hard problems with an iterative structure.

In addition, we want to match the upper and lower bounds for parallel fast-forwarding of Hamiltonian simulation. For instance, for geometrically local Hamiltonians, the algorithms in [23] require depth $O(t \cdot \mathrm{polylog}(tn/\epsilon))$, where n is the number of qubits and ϵ is the precision parameter. There is still a $O(1/ns(n))$ gap compared to our result in Theorem 4. Likewise, our results for sparse (Theorem 2) and local Hamiltonians (Theorem 3) have not matched the upper bounds from known quantum simulation algorithms, such as [27, 28, 39].

The questions mentioned above are to investigate the optimal quantum circuit depth for Hamiltonian simulation under certain computational assumptions. Note that the Hamiltonian simulation problem has classical inputs and quantum outputs. Inspired by this, we are wondering a more general question: *is it possible to prove quantum circuit lower bounds for complexity classes that have classical inputs and quantum outputs?* For example, can we unconditionally show quantum circuit depth lower bounds for Hamiltonian simulation or some quantum states with succinct classical descriptions? Note that although showing circuit depth lower bounds for languages is challenging and has some barriers, complexity classes with quantum outputs might have specific properties and provide new insights into showing quantum circuit depth lower bounds.

4 Preliminaries and Notation

4.1 Notation

For $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, 2, \ldots, n\}$. The trace distance between two density matrices ρ and σ is denoted by $\Delta(\rho, \sigma) := \frac{1}{2}\|\rho - \sigma\|_1 = \frac{1}{2}\mathrm{tr}\left(\sqrt{(\rho-\sigma)^\dagger(\rho-\sigma)}\right)$. Let x_1, x_2 be n-bit strings, we use $x_1 \oplus x_2$ to denote the bitwise XOR of x_1 and x_2. The Kronecker delta is denoted by δ_{jk} where $\delta_{jk} = 0$ if $j \neq k$ and $\delta_{jk} = 1$ if $j = k$.

4.2 Hamiltonian simulation

▶ **Definition 8** (Hamiltonian simulation). *A Hamiltonian simulation algorithm $\mathcal{A}$ takes as inputs the description of the Hamiltonian H, an initial state $|\psi_0\rangle$, the evolution time $t \geq 0$ and an error parameter $\epsilon \in (0, 1]$. Let $|\widetilde{\psi_t}\rangle$ be the ideal state under the Hamiltonian H for evolution time t with the initial state $|\psi_0\rangle$. In other words, $|\widetilde{\psi_t}\rangle := e^{-iHt}|\psi_0\rangle$ for a*

time-independent H*, and* $|\widetilde{\psi_t}\rangle := \exp_{\mathcal{T}}\left(-i\int_0^t H(t')dt'\right)|\psi_0\rangle$ *for a time-dependent* H*, where* $\exp_{\mathcal{T}}$ *is the time-ordered matrix operator. The goal of* $\mathcal{A}$ *is to generate an approximation* $|\psi_t\rangle$ *of the evolved quantum state* $|\widetilde{\psi_t}\rangle$ *such that*

$$\Delta\left(|\psi_t\rangle\langle\psi_t|, |\widetilde{\psi_t}\rangle\langle\widetilde{\psi_t}|\right) \leq \epsilon.$$

4.3 Basic quantum computation

Below, we provide a brief introduction to quantum computation. For more basics, we refer the readers to [31]. Throughout this work, we use the standard bra-ket notation.

▶ **Definition 9** (Quantum circuit model)**.** *A quantum circuit consists of qubits, a sequence of quantum gates, and measurements. A qubit is a two-dimensional complex Hilbert space. Each qubit is associated with a* register*. A quantum gate is a unitary operator acting on quantum registers. We say a quantum gate is a* k*-qubit gate if it acts non-trivially on* k *qubits.*

▶ **Theorem 10** (Universal gate sets [11])**.** *There exists a* universal gate set *that consists of a finite number of quantum gates such that any unitary operator can be approximated by composing elements in the universal gate set within an arbitrary error. Furthermore, every element in the universal gate set is a one- or two-qubit gate.*

▶ **Definition 11** (Quantum circuit depth)**.** *Given a finite-sized gate set* $\mathcal{G}$*, a* d-depth quantum circuit *or a* quantum circuit of depth d with respect to $\mathcal{G}$ *consists of a sequence of* d *layers of gates such that (i) each gate belongs in* $\mathcal{G}$ *and (ii) each gate within the same layer acts on disjoint qubits. We omit the gate set* $\mathcal{G}$ *when it is clear from the context.*

▶ **Definition 12** (Quantum query operator)**.** *Given an oracle* $f : \{0,1\}^n \to \{0,1\}^m$*, the* query operator $\mathcal{O}_f$ *is defined as*

$$\mathcal{O}_f|x,y\rangle := |x, y \oplus f(x)\rangle.$$

▶ **Definition 13** (Parallel quantum query operator)**.** *Given an oracle* $f : \{0,1\}^n \to \{0,1\}^m$*. The* k-parallel query operator $\mathcal{O}_f^{\otimes k}$ *is defined as*

$$\mathcal{O}_f^{\otimes k}|\mathbf{x},\mathbf{y}\rangle := |\mathbf{x}, \mathbf{y} \oplus f(\mathbf{x})\rangle,$$

where $\mathbf{x} = (x_1, \ldots, x_k)$*,* $\mathbf{y} = (y_1, \ldots, y_k)$ *and* $f(\mathbf{x}) := (f(x_1), \ldots f(x_k))$*.*

4.4 Useful tools

In this subsection, we introduce several definitions and lemmas for analyzing quantum random walk in Section 7 and the clock state construction in Section 9.

4.4.1 Bessel functions

The Bessel functions of the first kind of order n are denoted by $J_n(x)$. We present the required properties of Bessel functions for our use.

- The integration form of the Bessel function:

$$J_n(x) = \frac{1}{2\pi}\int_{-\pi}^{\pi} dp\ e^{inp - ix\sin p} = \frac{i^n}{2\pi}\int_{-\pi}^{\pi} dp\ e^{inp - ix\cos p}. \tag{1}$$

- The relation between J_n and J_{-n}:

$$J_{-n}(x) = (-1)^n J_n(x). \tag{2}$$

- The recursion formula for integer orders:

$$J_{n+1}(x) = \frac{2n}{x} J_n(x) - J_{n-1}(x). \tag{3}$$

- The asymptotic form for large order

$$J_n(n \operatorname{sech}\xi) \sim \frac{e^{-n(\xi - \tanh\xi)}}{\sqrt{2\pi n \tanh\xi}} \tag{4}$$

suggests that when $x < |n|$, the value of $J_n(x)$ is exponentially small in n.

The following lemmas provide upper bounds for Bessel functions for large argument x.

▶ **Lemma 14** (Theorem 2 in [26]). *Let $n > -1/2$ and $\mu := (2n+1)(2n+3)$. For any $x > \sqrt{\mu + \mu^{2/3}}/2$, it holds that*

$$J_n^2(x) \le \frac{4\left(4x^2 - (2n+1)(2n+5)\right)}{\pi\left((4x^2-\mu)^{3/2} - \mu\right)}.$$

By Lemma 14, we have the following lemma which is more convenient for our use.

▶ **Lemma 15.** *Let n be a positive integer. For any real $x \ge 2n$, it holds that*

$$J_n^2(x) \le \frac{2}{n\pi}.$$ [9]

Proof. We discuss the behavior of Bessel functions in three cases: $n = 1$, $n = 2$, and $n \ge 3$. For $n = 1$, the maximum of $J_1^2(x)$ is $0.339\ldots$ which is less than $2/\pi \approx 0.637$. For $n = 2$, the maximum of $J_2^2(x)$ is $0.237\ldots$ which is less than $1/\pi \approx 0.318$.

Now let us analyze the case in which $n \ge 3$. First, we notice that when $x > 2n$, the conditions in Theorem 14 hold. This is because $\mu + \mu^{2/3} < 2\mu$ and then

$$\frac{\sqrt{\mu + \mu^{2/3}}}{2} < \frac{\sqrt{2\mu}}{2} = \sqrt{2n^2 + 4n + \frac{3}{2}} < \sqrt{4n^2} = 2n < x,$$

where the second inequality holds when $n \ge 3$.

Now, we will finish the proof by bounding the numerator and the denominator of the RHS in Lemma 14. For the numerator, we have

$$4\left(4x^2 - (2n+1)(2n+5)\right) < 4\left(4x^2 - (2n+1)(2n+3)\right) = 4\left(4x^2 - \mu\right).$$

For the denominator, we will show that

$$(4x^2-\mu)^{\frac{3}{2}} - \mu > \frac{2}{3}(4x^2-\mu)^{\frac{3}{2}}$$

or equivalently

$$\frac{1}{3}(4x^2-\mu)^{\frac{3}{2}} > \mu.$$

First, since $x \geq \sqrt{2\mu}/2$, we have $4x^2 - \mu \geq \mu$. Furthermore, when $n \geq 3$ we have $\mu \geq 35$, which would imply $\frac{1}{3}\mu^{3/2} > \mu$. Hence, we conclude that $\frac{1}{3}(4x^2-\mu)^{\frac{3}{2}} \geq \frac{1}{3}\mu^{3/2} > \mu$. Putting things together, we obtain

$$J_n^2(x) < \frac{4}{\pi} \cdot \frac{(4x^2-\mu)}{\frac{2}{3}(4x^2-\mu)^{3/2}} = \frac{4}{\pi} \cdot \frac{1}{\frac{2}{3}\sqrt{4x^2-\mu}}.$$

When $x > 2n$ and $n > 3$, it holds that $4x^2 - \mu \geq 16n^2 - (4n^2 + 8n + 3) \geq 9n^2$. Therefore, we finally obtain

$$J_n^2(x) \leq \frac{4}{\pi} \cdot \frac{1}{\frac{2}{3}\sqrt{4x^2-\mu}} \leq \frac{4}{\pi} \cdot \frac{1}{\frac{2}{3} \cdot 3n} = \frac{2}{n\pi}.$$

This finishes the proof. ◀

4.4.2 Johnson graph

▶ **Definition 16** (Johnson Graph). *For all integers $n \geq k \geq 1$, the (n,k)-*Johnson graph *$J_{n,k} = (V, E)$ is an undirected acyclic graph defined as follows.*

- *$V := \{S \subseteq [n] \colon |S| = k\}$, i.e., the vertices are the k-element subsets of an n-element set.*
- *$E := \{(S_0, S_1) \colon |S_0 \cap S_1| = k-1\}$, i.e., there is an edge if and only if the intersection of the two vertices (subsets) contains $k-1$ elements.*[10]

The number of vertices in $J_{n,k}$ is $\binom{n}{k}$. It was proven that for all integers $n \geq k \geq 1$, there exists a Hamiltonian path[11] *in $J_{n,k}$ [2].*

5 Lower Bounding Permutation Chain

▶ **Definition 17** (Permutation notations). *Here we define several notations for the later proofs. Let $\Pi_1, \Pi_2, \ldots, \Pi_q$ be permutations of N elements. Let $\Pi_1^{-1}, \Pi_2^{-1}, \ldots, \Pi_q^{-1}$ be the corresponding inverse permutations. Define the sets $[-q] := \{-q, -q+1, \ldots, -1\}$ and $[\pm q] := [q] \cup [-q]$. We define the unitary $S\Pi$ as the controlled version of the above permutations as*

$$S\Pi|j,x,r\rangle := \begin{cases} |j, x, r \oplus \Pi_j(x)\rangle & , j > 0 \\ |j, x, r \oplus \Pi_{|j|}^{-1}(x)\rangle & , j < 0 \end{cases} \tag{5}$$

where $j \in [\pm q]$ and $x, r \in [N]$.

We denote the elements of the chain by $\bar{x}_1 := 1$ and $\bar{x}_{i+1} := \Pi_i(\bar{x}_i)$ for all $i \in [k]$. Next, we define $\tilde{\Pi}_i$ to be the "erased" Π_i for all $i \in [q]$. Formally, $\tilde{\Pi}_i$ is defined to be the function $[N] \to [N] \cup \{0\}$ such that

$$\tilde{\Pi}_i(x) = \begin{cases} \bar{x}_i & , x = \bar{x}_{i-1} \\ 0 & , \text{otherwise.} \end{cases} \tag{6}$$

Similarly, we define the corresponding controlled unitary $S\tilde{\Pi}$. For all $i \in [k]$, define $\Delta\bar{x}_i := \bar{x}_{i+1} - \bar{x}_i \mod N$. Note that $S\tilde{\Pi}$ can be parameterized by either $\{\bar{x}_2, \ldots, \bar{x}_{q+1}\}$ or $\{\Delta\bar{x}_1, \ldots, \Delta\bar{x}_q\}$. Denote the transformation from $S\Pi$ to $S\tilde{\Pi}$ by $S\tilde{\Pi} = F(S\Pi)$. For all $\ell \in [q]$, define the hybrid oracle $S\tilde{\Pi}_\ell$ as

[10] Equivalently, we can define $E := \{(S_0, S_1) \colon |S_0 \cup S_1| = k+1\}$.

[11] A Hamiltonian path is a path that visits every vertex in the graph exactly once. Do not confuse it with the physical quantity we want to simulate.

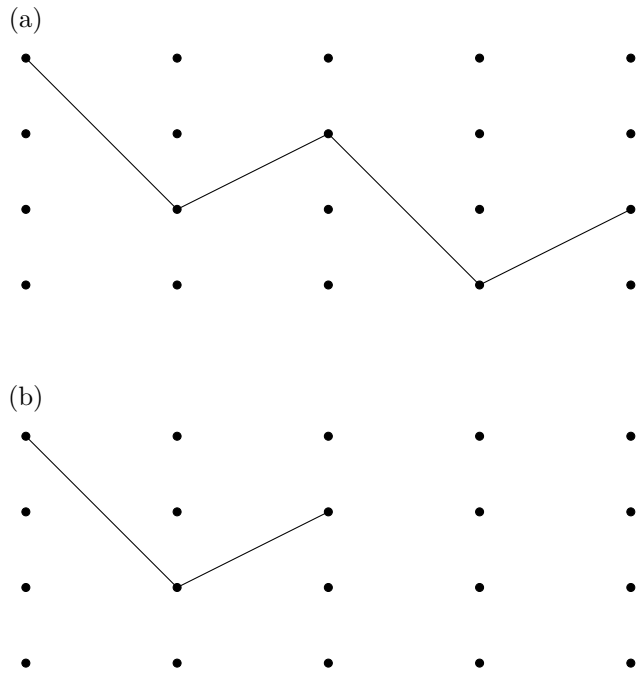


Figure 1 Schematic diagram of $\mathrm{S}\tilde{\Pi}$ and $\mathrm{S}\tilde{\Pi}_\ell$. (a) The permutation chain $\bar{x}_1, \bar{x}_2, \ldots \bar{x}_{q+1}$ specified by $\mathrm{S}\tilde{\Pi}$. (b) $\mathrm{S}\tilde{\Pi}_\ell$, where the permutations are removed after ℓ queries.

$$S\tilde{\Pi}_\ell|j,x,r\rangle := \begin{cases} |j,x,r \oplus \tilde{\Pi}_j(x)\rangle & ,|j| \le \ell, j > 0 \\ |j,x,r\rangle & ,j > \ell \\ |j,x,r \oplus \tilde{\Pi}_{|j|}^{-1}(x)\rangle & ,|j| \le \ell, j < 0 \\ |j,x,r\rangle & ,j < -\ell. \end{cases} \tag{7}$$

▶ **Theorem 18.** *Use the notations of Definition 17. Let q, k be integers such that $k = O(\mathrm{polylog}(N))$ and $q = O(\mathrm{polylog}(N))$. For any quantum algorithm $\mathcal{A}$ using $\lfloor (q-1)/2 \rfloor$ k-parallel queries to $S\Pi$, we have*

$$\mathop{\mathbb{E}}_{S\Pi}\left[\Pr[\mathcal{A}^{S\Pi} \text{ outputs } \bar{x}_{q+1}]\right] = O\left(q\sqrt{\frac{k}{N}}\right).$$

Before proving Theorem 18, we first introduce several lemmas as follows.

▶ **Lemma 19.** *For any $|\phi\rangle$, $|\psi\rangle$ such that $\||\phi\rangle\| = \||\psi\rangle\| = 1$ and $\||\phi\rangle - |\psi\rangle\| \le \varepsilon$, it holds that*

$$\Delta(|\phi\rangle\langle\phi|, |\psi\rangle\langle\psi|) \le \varepsilon.$$

Proof. The trace distance between two pure states is given by $\sqrt{1 - |\langle\phi|\psi\rangle|^2}$. The Euclidean norm of $|\phi\rangle - |\psi\rangle$ is given by $\||\phi\rangle - |\psi\rangle\| = \sqrt{(\langle\phi| - \langle\psi|)(|\phi\rangle - |\psi\rangle)} = \sqrt{2 - 2\mathsf{Re}[\langle\phi|\psi\rangle]}$, where $\mathsf{Re}[\cdot]$ denote the real part of a complex number.

First, it is true that $0 \le (\mathsf{Re}[\langle\phi|\psi\rangle] - 1)^2 + \mathsf{Im}[\langle\phi|\psi\rangle]^2$, where $\mathsf{Im}[\cdot]$ denote the imaginary part of a complex number. Rearranging the terms, we obtain

$$1 - |\langle\phi|\psi\rangle|^2 = 1 - (\mathsf{Re}[\langle\phi|\psi\rangle]^2 + \mathsf{Im}[\langle\phi|\psi\rangle]^2) \le 2 - 2\mathsf{Re}[\langle\phi|\psi\rangle]. \quad \blacktriangleleft$$

▶ **Lemma 20** (q-bin k-parallel Grover search lower bound). *Let $\mathcal{F}$ be the set of all functions f from $[qN]$ to $\{0,1\}$ with the following promise. For all $i \in \{0,1,\dots,q-1\}$, it holds that $|\{x \in \{iN+1, iN+2, \dots, iN+N\} \colon f(x)=1\}| = 1$. Let g be the constant zero function with domain $[qN]$. Then for every algorithm that makes ℓ k-parallel queries to f (or g), the final state of the algorithm, denoted by $|\psi^f\rangle$ (or $|\psi^g\rangle$), satisfies*

$$\mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\Delta\left(|\psi^f\rangle\langle\psi^f|, |\psi^g\rangle\langle\psi^g|\right)\right] = O\left(\ell\sqrt{\frac{k}{N}}\right).$$

Proof. Let $|\psi_i^g\rangle \coloneqq U_i O_g^{\otimes k} \dots U_1 O_g^{\otimes k} U_0 |0\rangle$. For any $f \in \mathcal{F}$, let Π_f be the projector acting on the query register of the algorithm that is defined as $\Pi_f \coloneqq \sum_{x:f(x)=1} |x\rangle\langle x|$. Let $\overline{\Pi}_f \coloneqq I - \Pi_f$.

Notice that $(O_f^{\otimes k} - O_g^{\otimes k})\overline{\Pi}_f^{\otimes k} = 0$ because O_f and O_g are identical beyond the set of the 1-preimages. Therefore, we have

$$(O_f^{\otimes k} - O_g^{\otimes k})(I - \overline{\Pi}_f^{\otimes k}) = O_f^{\otimes k} - O_g^{\otimes k}. \tag{8}$$

Also, it holds that

$$I - \overline{\Pi}_f^{\otimes k} \le \sum_{i=j}^{k} \Pi_{f,j}, \tag{9}$$

where $\Pi_{f,j}$ denote the operator the acts as Π_f on the register of the j-th query branch and as identity on other registers; for matrices A, B, by $A \ge B$ we mean that $A - B$ is a positive semi-definite matrix. Then by standard hybrid arguments [5], we have

$$\begin{aligned}
\mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\||\psi^f\rangle - |\psi^g\rangle\|\right] &\le \sum_{i=0}^{\ell-1} \mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\|(O_f^{\otimes k} - O_g^{\otimes k})|\psi_i^g\rangle\|\right] \\
&= \sum_{i=0}^{\ell-1} \mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\|(O_f^{\otimes k} - O_g^{\otimes k})(I - \overline{\Pi}_f^{\otimes k})|\psi_i^g\rangle\|\right] \\
&\le \sum_{i=0}^{\ell-1} \mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\|(O_f^{\otimes k} - O_g^{\otimes k})\| \cdot \|(I - \overline{\Pi}_f^{\otimes k})|\psi_i^g\rangle\|\right],
\end{aligned}$$

where the first equality is due to (8) and the last inequality is due to the fact that $\|A|\phi\rangle\| \le \|A\| \cdot \||\phi\rangle\|$, where $\|A\|$ denotes the operator norm of A.

Since $\|O_f\| = \|O_g\| = 1$, by the triangle inequality we can bound it as

$$\begin{aligned}
&\le 2 \sum_{i=0}^{\ell-1} \mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\|(I - \overline{\Pi}_f^{\otimes k})|\psi_i^g\rangle\|\right] \\
&= 2 \sum_{i=1}^{\ell-1} \mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\sqrt{\langle\psi_i^g|(I - \overline{\Pi}_f^{\otimes k})|\psi_i^g\rangle}\right] \\
&\le 2 \sum_{i=0}^{\ell-1} \sqrt{\mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\langle\psi_i^g|(I - \overline{\Pi}_f^{\otimes k})|\psi_i^g\rangle\right]} \qquad \text{(Jensen's inequality)} \\
&\le 2 \sum_{i=0}^{\ell-1} \sqrt{\mathop{\mathbb{E}}_{f \leftarrow \mathcal{F}} \left[\langle\psi_i^g|\sum_{j=1}^{k} \Pi_{f,j}|\psi_i^g\rangle\right]}
\end{aligned}$$

$$\leq 2\sum_{i=0}^{\ell-1}\sqrt{\sum_{j=1}^{k}\mathop{\mathbb{E}}_{f\leftarrow\mathcal{F}}[\langle\psi_i^g|\Pi_{f,j}|\psi_i^g\rangle]} \qquad \text{(linearity of expectation)}$$

$$= 2\sum_{i=0}^{\ell-1}\sqrt{\sum_{j=1}^{k}\langle\psi_i^g|\frac{I}{N}|\psi_i^g\rangle} = 2\ell\sqrt{\frac{k}{N}},$$

where the first equality holds since $I-\overline{\Pi}_f^{\otimes k}$ is a projection operator; the third inequality holds due to (9) and the second equality holds because the probability of any $x\in[qN]$ being a 1-preimage of f is $1/N$. So we have $\mathbb{E}_{f\leftarrow\mathcal{F}}[\Pi_{f,j}]=I/N$ for every j. Finally, by Lemma 19, we have

$$\mathop{\mathbb{E}}_{f\leftarrow\mathcal{F}}\left[\Delta\left(|\psi^f\rangle\langle\psi^f|,|\psi^g\rangle\langle\psi^g|\right)\right]\leq 2\ell\sqrt{\frac{k}{N}}.$$

as desired. ◀

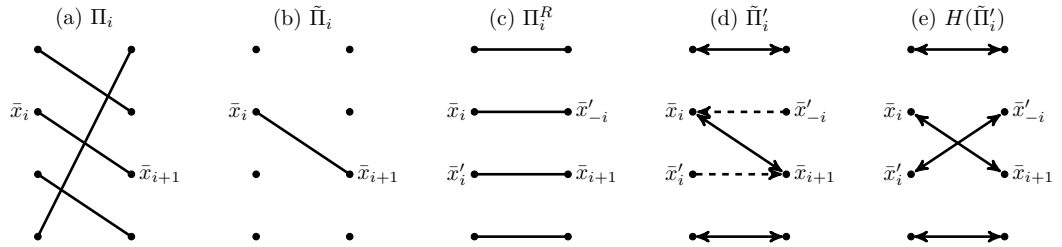


Figure 2 Schematic diagrams of permutation in the hybrid proof. (a) Π_i is the permutation given by the problem. (b) $\tilde{\Pi}_i$ maps $\bar{x}_i$ to $\bar{x}_{i+1}:=\Pi_i(\bar{x}_i)$ and maps other inputs to a dummy image 0. (c) Π_i^R is a random permutation that is independent of Π_i. (d) $\tilde{\Pi}'_i$ merges $\tilde{\Pi}_i$ and Π_i^R. It maps $\bar{x}_i$ to $\bar{x}_{i+1}$, and maps other input x to $\Pi_i^R(x)$. There is a collision on inputs $\bar{x}_i$ and $\bar{x}'_i:={\Pi^R}_i^{-1}(\bar{x}_{i+1})$. When executing the "inverse" of $\tilde{\Pi}'_i$, it follows the rules of ${\Pi^R}_i^{-1}$. Note that the "inverse" is not exactly equal to ${\tilde{\Pi}'}_i^{-1}$. (e) The truth table of $H(\tilde{\Pi}'_i)$ is equal to $\tilde{\Pi}'_i$ except that $H(\tilde{\Pi}'_i)(\bar{x'}_i)=\bar{x'}_{-i}$.

▶ **Lemma 21.** *Use the notations of Definition 17. Let ℓ,k be integers such that $\ell=O(\mathrm{polylog}(N))$ and $k=O(\mathrm{polylog}(N))$. For any quantum algorithm $\mathcal{A}$ using ℓ k-parallel queries to $S\Pi$, there is a quantum algorithm $\tilde{\mathcal{A}}$ using 2ℓ k-parallel queries to $S\tilde{\Pi}$ such that for all $S\tilde{\Pi}$,*

$$\mathop{\mathbb{E}}_{S\Pi\in F^{-1}(S\tilde{\Pi})}\left[\Pr[\mathcal{A}^{S\Pi}\neq\tilde{\mathcal{A}}^{S\tilde{\Pi}}]\right]=O\left(\ell\sqrt{\frac{k}{N}}\right).$$

Proof. Let $\Pi_1^R,\Pi_2^R,\ldots,\Pi_q^R$ be permutations of N elements. Let $S\Pi^R$ be the corresponding unitary.

We construct $\tilde{\mathcal{A}}$ as follows: it samples a uniformly random $S\Pi^R$ and runs $\mathcal{A}$ but replaces every query to $S\Pi$ with $S\tilde{\Pi}'$ constructed from uniformly random $S\Pi^R$, where $S\tilde{\Pi}'$ is defined as

$$S\tilde{\Pi}'|i,x,r\rangle:=\begin{cases}|i,x,r\oplus\tilde{\Pi}_i(x)\rangle & ,\tilde{\Pi}_i(x)\neq 0,i>0\\ |i,x,r\oplus\Pi_i^R(x)\rangle & ,\tilde{\Pi}_i(x)=0,i>0\\ |i,x,r\oplus\tilde{\Pi}_{|i|}^{-1}(x)\rangle & ,\tilde{\Pi}_{|i|}^{-1}(x)\neq 0,i<0\\ |i,x,r\oplus{\Pi^R}_{|i|}^{-1}(x)\rangle & ,\tilde{\Pi}_{|i|}^{-1}(x)=0,i<0.\end{cases} \tag{10}$$

One query to $\mathrm{S}\tilde{\Pi}'$ can be constructed from two queries to $\mathrm{S}\tilde{\Pi}$ coherently by doing the obvious classical calculation and uncomputing the garbage. Therefore $\tilde{\mathcal{A}}$ uses 2ℓ queries to $\mathrm{S}\tilde{\Pi}$ as required.

Now we prove that it is very hard to distinguish $\mathcal{A}$ from $\tilde{\mathcal{A}}$. This is done by a reduction to the hardness of the modified Grover's search algorithm in Lemma 20.

For all $i \in [q]$, define $\bar{x}'_i$ to be x such that $\tilde{\Pi}'(x) = \bar{x}_{i+1}$ and $x \neq \bar{x}$. For all $i \in [-q]$, define $\bar{x}'_i$ to be x such that $\tilde{\Pi}'^{-1}_{|i|}(x) = \bar{x}_{|i|}$ and $x \neq \bar{x}_{|i|+1}$. Note that $\bar{x}'_i$ does not exist if $\Pi^R_i(\bar{x}_i) = \tilde{\Pi}_i(\bar{x}_i)$.

Note that each $\tilde{\Pi}'_i$ looks like a random permutation except on the collisions $\{\bar{x}'_i\}$. We define a function H which relates these similar $\mathrm{S}\Pi$ and $\mathrm{S}\tilde{\Pi}'$:

$$H(\mathrm{S}\tilde{\Pi}')|i,x,r\rangle := \begin{cases} |i,x,r\oplus\tilde{\Pi}_i(x)\rangle & ,\tilde{\Pi}_i(x)\neq 0, i>0 \\ |i,x,r\oplus\Pi^R_i(x)\rangle & ,\tilde{\Pi}_i(x)=0, x\neq\bar{x}'_i, i>0 \\ |i,x,r\oplus\bar{x}'_{-i}\rangle & ,x=\bar{x}'_i, i>0 \\ |i,x,r\oplus\tilde{\Pi}^{-1}_{|i|}(x)\rangle & ,\tilde{\Pi}^{-1}_{|i|}(x)\neq 0, i<0 \\ |i,x,r\oplus\Pi^{R\ -1}_{|i|}(x)\rangle & ,\tilde{\Pi}^{-1}_{|i|}(x)=0, x\neq\bar{x}'_i, i<0 \\ |i,x,r\oplus\bar{x}'_{-i}\rangle & ,x=\bar{x}'_i, i<0. \end{cases} \tag{11}$$

It is easy to check that for all $\mathrm{S}\tilde{\Pi}'$, $H(\mathrm{S}\tilde{\Pi}')$ is a valid $\mathrm{S}\Pi$, and for a given $\mathrm{S}\Pi$, there are N^q elements in $H^{-1}(\mathrm{S}\Pi)$. It is also easy to verify that for a fixed $\mathrm{S}\tilde{\Pi}$, different $H^{-1}(\mathrm{S}\Pi)$ partitions all possible $\mathrm{S}\tilde{\Pi}'$.

Let $\rho_{\mathrm{S}\Pi}$ be the final density matrix of $\mathcal{A}^{\mathrm{S}\Pi}$. Let $\rho_{\mathrm{S}\tilde{\Pi}'}$ be the final density matrix of $\tilde{\mathcal{A}}^{\mathrm{S}\tilde{\Pi}}$ with a fixed $\mathrm{S}\Pi'$. By the strong convexity of trace distance, we have

$$\Delta\left(\mathop{\mathbb{E}}_{\mathrm{S}\Pi\in F^{-1}(\mathrm{S}\tilde{\Pi})}[\rho_{\mathrm{S}\Pi}], \mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}'\in H^{-1}(F^{-1}(\mathrm{S}\tilde{\Pi}))}[\rho_{\mathrm{S}\tilde{\Pi}'}]\right) \leq \mathop{\mathbb{E}}_{\mathrm{S}\Pi\in F^{-1}(\mathrm{S}\tilde{\Pi})}\left[\Delta\left(\rho_{\mathrm{S}\Pi}, \mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}'\in H^{-1}(\mathrm{S}\Pi)}[\rho_{\mathrm{S}\tilde{\Pi}'}]\right)\right]. \tag{12}$$

Finally, we prove that $\Delta(\rho_{\mathrm{S}\Pi}, \mathbb{E}_{\mathrm{S}\tilde{\Pi}'\in H^{-1}(\mathrm{S}\Pi)}[\rho_{\mathrm{S}\tilde{\Pi}'}]) = O\left(\ell\sqrt{\frac{k}{N}}\right)$ for all $\mathrm{S}\Pi$ by a reduction to the modified Grover search problem. Consider a Grover oracle G defined in Lemma 20 that might be f or g. Given free calls to $\mathrm{S}\Pi$, we use two calls to G to construct an oracle $\mathrm{S}\Pi^G$ which equals $\mathrm{S}\Pi$ when $G = g$ and equals a random $\mathrm{S}\tilde{\Pi}'$ when $G = f$. The construction is as follows:

$$\mathrm{S}\Pi^G|i,x,r\rangle := \begin{cases} |i,x,r\oplus\tilde{\Pi}_i(x)\rangle & ,\tilde{\Pi}_i(x)\neq 0, i>0 \\ |i,x,r\oplus\Pi_i(x)\rangle & ,\tilde{\Pi}_i(x)=0, G((i-1)N+x)=0, i>0 \\ |i,x,r\oplus\bar{x}_{i+1}\rangle & ,\tilde{\Pi}_i(x)=0, G((i-1)N+x)=1, i>0 \\ |i,x,r\oplus\tilde{\Pi}^{-1}_{|i|}(x)\rangle & ,\tilde{\Pi}^{-1}_{|i|}(x)\neq 0, i<0 \\ |i,x,r\oplus\Pi^{-1}_{|i|}(x)\rangle & ,\tilde{\Pi}^{-1}_{|i|}(x)=0, G((i-1)N+\Pi^{-1}_{|i|}(x))=0, i<0 \\ |i,x,r\oplus\bar{x}_i\rangle & ,\tilde{\Pi}^{-1}_{|i|}(x)=0, G((i-1)N+\Pi^{-1}_{|i|}(x))=1, i<0. \end{cases} \tag{13}$$

By Lemma 20, if we try to distinguish f from g by distinguishing $\mathcal{A}^{\mathrm{S}\Pi^G}$ of the two cases, we can only succeed with probability $O\left(\ell\sqrt{\frac{k}{N}}\right)$ since we only have $O(\ell)$ k-parallel queries to G. Therefore, one can only distinguish $\mathrm{S}\Pi$ from $\mathrm{S}\tilde{\Pi}'$ with probability $O\left(\ell\sqrt{\frac{k}{N}}\right)$, i.e.,

$$\Delta\left(\rho_{\mathrm{S}\Pi}, \mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}'\in H^{-1}(\mathrm{S}\Pi)}[\rho_{\mathrm{S}\tilde{\Pi}'}]\right) = O\left(\ell\sqrt{\frac{k}{N}}\right). \tag{14}$$

Then by (12),

$$\Delta\left(\mathop{\mathbb{E}}_{\mathrm{S}\Pi\in F^{-1}(\mathrm{S}\tilde{\Pi})}\left[\rho_{\mathrm{S}\Pi}\right], \mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}'\in H^{-1}(F^{-1}(\mathrm{S}\tilde{\Pi}))}\left[\rho_{\mathrm{S}\tilde{\Pi}'}\right]\right) = O\left(\ell\sqrt{\frac{k}{N}}\right). \tag{15}$$

By the operational interpretation of the trace distance, we have

$$\mathop{\mathbb{E}}_{\mathrm{S}\Pi\in F^{-1}(\mathrm{S}\tilde{\Pi})}\left[\Pr[\mathcal{A}^{\mathrm{S}\Pi}\neq\tilde{\mathcal{A}}^{\mathrm{S}\tilde{\Pi}}]\right] = O\left(\ell\sqrt{\frac{k}{N}}\right). \tag{16}$$

◀

▶ **Lemma 22.** *Use the notations of Definition 17. Let ℓ, k be integers such that $\ell \in [q]$, $k = O(\mathrm{polylog}(N))$. For all algorithm $\mathcal{A}$ using ℓ k-parallel queries to $S\tilde{\Pi}$, w.l.o.g. we can assume the final output of $\mathcal{A}$ has the form*

$$|\psi\rangle = U_\ell S\tilde{\Pi}^{\otimes k} U_{\ell-1} S\tilde{\Pi}^{\otimes k} \dots U_2 S\tilde{\Pi}^{\otimes k} U_1 S\tilde{\Pi}^{\otimes k}|\psi_0\rangle.$$

For all $m, p \in [\ell]$, $p \le m$, define the hybrid state

$$|\psi_{m,p}\rangle := U_m S\tilde{\Pi}^{\otimes k} U_{m-1} S\tilde{\Pi}^{\otimes k} \dots U_{p+1} S\tilde{\Pi}^{\otimes k} U_p S\tilde{\Pi}_p^{\otimes k} U_{p-1} S\tilde{\Pi}_{p-1}^{\otimes k} \dots U_2 S\tilde{\Pi}_2^{\otimes k} U_1 S\tilde{\Pi}_1^{\otimes k}|\psi_0\rangle.$$

Then for all $\mathcal{A}$,

$$\mathop{\mathbb{E}}_{S\tilde{\Pi}} \||\psi\rangle - |\psi_{\ell,\ell}\rangle\| \le 2\ell\sqrt{\frac{k}{N}}. \tag{17}$$

Proof. Note that $|\psi\rangle = |\psi_{\ell,0}\rangle$. By triangle inequality we have

$$\mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}} \||\psi\rangle - |\psi_{\ell,\ell}\rangle\| \le \sum_{i=1}^{\ell} \mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}} \||\psi_{\ell,i-1}\rangle - |\psi_{\ell,i}\rangle\|. \tag{18}$$

We proceed by proving $\mathbb{E}_{\mathrm{S}\tilde{\Pi}} \||\psi_{\ell,i-1}\rangle - |\psi_{\ell,i}\rangle\| = 2\sqrt{k/N}$ for all $i \in [\ell]$. Note that

$$\begin{aligned}
\||\psi_{\ell,i-1}\rangle - |\psi_{\ell,i}\rangle\| &= \||\psi_{i,i-1}\rangle - |\psi_{i,i}\rangle\| \\
&= \left\|U_i \mathrm{S}\tilde{\Pi}^{\otimes k}|\psi_{i-1,i-1}\rangle - U_i \mathrm{S}\tilde{\Pi}_i^{\otimes k}|\psi_{i-1,i-1}\rangle\right\| \\
&= \left\|(\mathrm{S}\tilde{\Pi}^{\otimes k} - \mathrm{S}\tilde{\Pi}_i^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|.
\end{aligned} \tag{19}$$

Note that $\mathrm{S}\tilde{\Pi}|j,x,r\rangle$ and $\mathrm{S}\tilde{\Pi}_i|j,x,r\rangle$ differs only when $j > i$ and $x = \bar{x}_j$ or $j < -i$ and $x = \bar{x}_{j+1}$. Therefore

$$(\mathrm{S}\tilde{\Pi} - \mathrm{S}\tilde{\Pi}_i)|j,x,r\rangle = (\mathrm{S}\tilde{\Pi} - \mathrm{S}\tilde{\Pi}_i)P_i|j,x,r\rangle \tag{20}$$

$$\mathrm{S}\tilde{\Pi}(1-P_i) = \mathrm{S}\tilde{\Pi}_i(I-P_i) \tag{21}$$

$$(\mathrm{S}\tilde{\Pi}^{\otimes k} - \mathrm{S}\tilde{\Pi}_i^{\otimes k})(I-P_i)^{\otimes k} = 0 \tag{22}$$

where

$$P_i := \left(\sum_{j=i+1}^{k} |j,\bar{x}_j\rangle\langle j,\bar{x}_j| + \sum_{-j=i+1}^{k} |j,\bar{x}_{j+1}\rangle\langle j,\bar{x}_{j+1}|\right) \otimes I. \tag{23}$$

Note that P_i actually depends on $\mathrm{S}\tilde{\Pi}$, but we omit the dependence for cleaner notation. Therefore we have

$$
\begin{aligned}
&\left\|(\mathrm{S}\tilde{\Pi}^{\otimes k}-\mathrm{S}\tilde{\Pi}_i^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|^2\\
=&\left\|(\mathrm{S}\tilde{\Pi}^{\otimes k}-\mathrm{S}\tilde{\Pi}_i^{\otimes k})(I-(I-P_i)^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|^2\\
\leq&4\left\|(I-(I-P_i)^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|^2\\
=&4\langle\psi_{i-1,i-1}|(I-(I-P_i)^{\otimes k})|\psi_{i-1,i-1}\rangle\\
\leq&4\langle\psi_{i-1,i-1}|\sum_{j=1}^{k}P_i^j|\psi_{i-1,i-1}\rangle
\end{aligned}
\tag{24}
$$

where $P_i^j = I^{\otimes j-1}\otimes P_i\otimes I^{\otimes k-j}$. In the fourth line, we use the fact that $I-(I-P_i)^{\otimes k}$ is a projector, and in the fifth line we use the standard union bound calculation.

By (23), for all $i\in[q]$ and $j\in[k]$, P_i^j is normalized by

$$
\sum_{\Delta\bar{x}_i=0}^{N-1}P_i^j=\left(\sum_{j=i+1}^{k}+\sum_{-j=i+1}^{k}\right)|j\rangle\langle j|\otimes\sum_{\bar{x}\in[N]}|\bar{x}\rangle\langle\bar{x}|\leq I \tag{25}
$$

where we omitted the tensor product of identities and assume $\Delta\bar{x}_{i+1},\ldots,\Delta\bar{x}_q$ to be fixed. Therefore

$$
\begin{aligned}
&\sum_{\Delta\bar{x}_i=0}^{N-1}\left\|(\mathrm{S}\tilde{\Pi}^{\otimes k}-\mathrm{S}\tilde{\Pi}_i^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|^2\\
\leq&\sum_{\Delta\bar{x}_i=0}^{N-1}4\langle\psi_{i-1,i-1}|\sum_{j=1}^{k}P_i^j|\psi_{i-1,i-1}\rangle\\
\leq&4k
\end{aligned}
\tag{26}
$$

Finally, combining (26), (24), and (19), we have

$$
\begin{aligned}
&\mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}}\||\psi_{\ell,i-1}\rangle-|\psi_{\ell,i}\rangle\|\\
=&\mathop{\mathbb{E}}_{\Delta\bar{x}_1,\ldots,\Delta\bar{x}_{i-1}}\mathop{\mathbb{E}}_{\Delta\bar{x}_i}\mathop{\mathbb{E}}_{\Delta\bar{x}_{i+1},\ldots,\Delta\bar{x}_q}\left\|(\mathrm{S}\tilde{\Pi}^{\otimes k}-\mathrm{S}\tilde{\Pi}_i^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|\\
=&\mathop{\mathbb{E}}_{\Delta\bar{x}_1,\ldots,\Delta\bar{x}_{i-1}}\mathop{\mathbb{E}}_{\Delta\bar{x}_{i+1},\ldots,\Delta\bar{x}_q}\frac{1}{N}\sum_{\Delta\bar{x}_i}\left\|(\mathrm{S}\tilde{\Pi}^{\otimes k}-\mathrm{S}\tilde{\Pi}_i^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|\\
\leq&\mathop{\mathbb{E}}_{\Delta\bar{x}_1,\ldots,\Delta\bar{x}_{i-1}}\mathop{\mathbb{E}}_{\Delta\bar{x}_{i+1},\ldots,\Delta\bar{x}_q}\frac{1}{N}\sqrt{\sum_{\Delta\bar{x}_i}\left\|(\mathrm{S}\tilde{\Pi}^{\otimes k}-\mathrm{S}\tilde{\Pi}_i^{\otimes k})|\psi_{i-1,i-1}\rangle\right\|^2}\cdot\sqrt{N}\\
=&2\sqrt{\frac{k}{N}}.
\end{aligned}
\tag{27}
$$

Plugging back to (18) we have

$$
\mathop{\mathbb{E}}_{\mathrm{S}\tilde{\Pi}}\||\psi\rangle-|\psi_{\ell,\ell}\rangle\|\leq 2\ell\sqrt{\frac{k}{N}}. \tag{28}
$$

◀

▶ **Corollary 23.** *Any algorithm $\mathcal{A}$ using $(q-1)$ k-parallel queries to $\mathrm{S}\tilde{\Pi}$ can only output $\bar{x}_{q+1}$ with probability $O\left(\ell\sqrt{\frac{k}{N}}\right)$.*

Proof. By Lemma 19 and Lemma 22, output probability of $|\psi\rangle$ and $|\psi_{\ell,\ell}\rangle$ only differ by $O(\ell\sqrt{k/N})$. Since $|\psi_{q-1,q-1}\rangle$ does not depend on $\tilde{\Pi}_q$, it can only find $\bar{x}^{q+1}$ with probability $1/N$. Thus the maximum probability of $|\psi\rangle$ finding $\bar{x}^{q+1}$ is $O(\ell\sqrt{k/N}) + 1/N = O(\ell\sqrt{k/N})$. ◀

Proof of Theorem 18. Let $\mathcal{A}$ be an algorithm that uses $\lfloor (q-1)/2 \rfloor$ k-parallel queries to $\mathrm{S}\Pi$ and outputs $\bar{x}_{q+1}$ with some probability p. By Lemma 21, there is an algorithm $\tilde{\mathcal{A}}$ that uses $(q-1)$ k-parallel queries to $\mathrm{S}\tilde{\Pi}$ and outputs $\bar{x}_{q+1}$ with probability at least $p - O\left(q\sqrt{\frac{k}{N}}\right)$. By Corollary 23, it holds that $p - O\left(q\sqrt{\frac{k}{N}}\right) = O\left(q\sqrt{\frac{k}{N}}\right)$. Therefore, we have $p = O\left(q\sqrt{\frac{k}{N}}\right)$. ◀

6 Parallel Hardness of Twisted Hash Chains

The (standard) hash chain problem is a natural candidate for parallel hardness. However, hash functions (modeled as random functions) with equal-length inputs and outputs are not injective with overwhelming probability. Importantly, quantum operations need to be reversible. Therefore, if we evaluate the standard hash chain straight-forwardly, the intermediate hash values are required to be stored in ancilla qubits. Consequently, the width of the circuit would become proportional to the length of the chain, which means the computation could not be done within width λ.

Inspired by the construction of the Feistel cipher which implements a permutation by hash functions, we introduce the *twisted hash chain* problem. Let $\mathcal{X}$ be $\{0,1\}^n$. An *s-chain* is a sequence $x_0, x_1, \ldots, x_s \in \mathcal{X}$ such that $x_i = H(x_{i-1}) \oplus x_{i-2}$ for $i \in [s]$ where we use the convention that $x_{-1} \coloneqq 0^n$. Informally, the task of a q-query k-parallel algorithm that interacts with a random oracle $H : \mathcal{X} \to \mathcal{Y}$ with $|\mathcal{X}| = |\mathcal{Y}|$ is to output $x_0, x_q, x_{q+1} \in \mathcal{X}$ of a $(q+1)$-chain, i.e., there exists a sequence $x_1, \ldots, x_{q-1} \in \mathcal{X}$ such that $x_i = H(x_{i-1}) \oplus x_{i-2}$ for $i \in [q+1]$. The computation of a twisted hash chain is shown in Fig. 3.

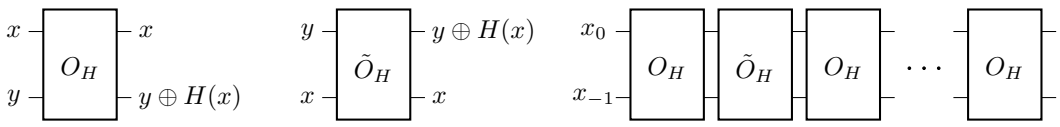


Figure 3 Schematic diagram of the twisted hash chain.

In this section, we aim to prove the following theorem.

▶ **Theorem 24** (Twisted hash chain is sequential). *For any k-parallel q-query oracle algorithm $\mathcal{C}$, the probability $p_{\mathcal{C}}$ (parameterized by k and q) that $\mathcal{C}$ outputs $x_0, x_q, x_{q+1} \in \mathcal{X}$ satisfying the following condition:*

- *there exist $x_1, \ldots, x_{q-1} \in \mathcal{X}$ such that $H(x_{i-1}) = x_i \oplus x_{i-2}$ for $i \in [q+1]$, where $x_{-1} \coloneqq 0^n$*

is at most $F(k, 2q) = O(k^4 q^4 / |\mathcal{Y}|)$, where the function F is defined in Lemma 32.

Toward proving the hardness, we exploit the framework of [16]. Below, we borrow the notations and definitions of [16]. Let $H \colon \mathcal{X} \to \mathcal{Y}$ be a random oracle. Let $\hat{\mathcal{Y}}$ be the dual group of $\mathcal{Y}$. Let $\overline{\mathcal{Y}}$ denote the set $\mathcal{Y} \cup \{\perp\}$. We say that $D : \mathcal{X} \to \overline{\mathcal{Y}}$ is a *database*. By $\mathfrak{D}$ we mean the set of all databases, i.e., the set of all functions from $\mathcal{X}$ to $\overline{\mathcal{Y}}$. For any tuple $\mathbf{x} = (x_1, \ldots, x_k)$ with pairwise disjoint $x_i \in \mathcal{X}$, tuple $\mathbf{r} = (r_1, \ldots, r_k) \in \overline{\mathcal{Y}}^k$ and database $D \in \mathfrak{D}$, we define the database $D[\mathbf{x} \mapsto \mathbf{r}]$ as

$$D[\mathbf{x} \mapsto \mathbf{r}](x) \coloneqq \begin{cases} r_i & \text{if } x = x_i \text{ for some } i \in [k] \\ D(x) & \text{if } x \notin \{x_1, \dots, x_k\}. \end{cases}$$

By database property P we mean a set of databases, that is $P \subseteq \mathfrak{D}$. In this section, we assume that $\mathcal{X} = \mathcal{Y}$. For any database $D \in \mathfrak{D}$ and tuple $\mathbf{x} = (x_1, \dots, x_k)$ of pairwise distinct $x_i \in \mathcal{X}$, we let

$$D|^{\mathbf{x}} \coloneqq \{D[\mathbf{x} \mapsto \mathbf{r}] \mid \mathbf{r} \in \overline{\mathcal{Y}}^k\} \subseteq \mathfrak{D}$$

be the set of databases that coincide D outside of $\mathbf{x}$. Furthermore, for any database property $\mathsf{P} \subseteq \mathfrak{D}$, we let

$$\mathsf{P}|_{D|^{\mathbf{x}}} \coloneqq \mathsf{P} \cap D|^{\mathbf{x}}.$$

▶ **Definition 25** (Definition 5.5 in [16]). *Let P, P' be two database properties. Then, the* quantum transition capacity *(of order k) is defined as*

$$[\![\mathsf{P} \xrightarrow{k} \mathsf{P}']\!] \coloneqq \max_{\mathbf{x}, \hat{\mathbf{y}}, D} \|\mathsf{P}'|_{D|^{\mathbf{x}}} \mathsf{cO}_{\mathbf{x}\hat{\mathbf{y}}} \mathsf{P}|_{D|^{\mathbf{x}}}\|,$$

where the maximum is over all possible $\mathbf{x} \in \mathcal{X}^k$, $\hat{\mathbf{y}} \in \hat{\mathcal{Y}}^k$ and $D \in \mathfrak{D}$. Furthermore, we define

$$[\![\mathsf{P} \xRightarrow{k} \mathsf{P}']\!] \coloneqq \sup_{U_1, \dots, U_{q-1}} \left\| \mathsf{P}' U_{q-1} \mathsf{cO}^k U_{q-1} \mathsf{cO}^k \dots U_1 \mathsf{cO}^k \mathsf{P} \right\|,$$

where $\|\cdot\|$ is the operator norm; the supremum is over all positive $d \in \mathbb{Z}$ and all unitaries $U_1, \dots, U_{q-1}$ acting on $\mathbb{C}[\mathcal{X}] \otimes \mathbb{C}[\mathcal{Y}] \otimes \mathbb{C}^d$.[12] *For the formal definitions of $\mathsf{cO}_{\mathbf{x}\hat{\mathbf{y}}}$ and cO^k, we refer to [16]. We note that their definitions are not required for the following proof.*

▶ **Definition 26.** *The database property* twisted hash chain of length *s, denoted by $\mathsf{TCHN}^{\mathsf{s}}$, is defined as*

$$\mathsf{TCHN}^{\mathsf{s}} \coloneqq \{D \mid \exists x_0, x_1, \dots, x_s \in \mathcal{X} \colon x_i = D(x_{i-1}) \oplus x_{i-2}, \forall i \in [s]\} \subseteq \mathfrak{D},$$

where we use the convention that $x_{-1} \coloneqq 0^n$ for convenience.

▶ **Definition 27** (Definition 5.20 in [16], with ℓ fixed to 1). *A database transition $\mathsf{P} \to \mathsf{P}'$ is said be* k-non-uniformly weakly recognizable *by* 1*-local properties*[13] *if for every $\mathbf{x} = (x_1, \dots, x_k)$ with pairwise disjoint entries, and for every $D \in \mathfrak{D}$, there exists a family of* 1*-local properties $\{L_i^{D,\mathbf{x}}\}$ where each $L_i^{D,\mathbf{x}} \subseteq \overline{\mathcal{Y}}$ and the support of $L_i^{D,\mathbf{x}}$ is $\{x_i\}$ or empty, so that*

$$D[\mathbf{x} \mapsto \mathbf{r}] \in \mathsf{P} \wedge [\mathbf{x} \mapsto \mathbf{u}] \in \mathsf{P}' \implies \exists i \colon u_i \in L_i^{D,\mathbf{x}} \wedge r_i \neq u_i.$$

▶ **Theorem 28** (Theorem 5.23 in [16]). *Let P and P' be k-non-uniformly weakly recognizable by* 1*-local properties $L_i^{\mathbf{x},D}$, where the support of $L_i^{\mathbf{x},D}$ is $\{x_i\}$ or empty. Then*

$$[\![\bot \xRightarrow{q,k} \mathsf{TCHN}^{q+1}]\!] \leq \max_{\mathbf{x}, D} e \sum_i \sqrt{10 P[U \in L_i^{\mathbf{x},D}]},$$

where U is defined to be uniformly random in $\mathcal{Y}$ and $\bot \coloneqq \{D \mid D(x) = \bot \text{ for all } x \in \mathcal{X}\}$.

[12] Namely, over all q-query quantum algorithms.
[13] We refer to Definition 5.10 in [16] for the formal description of local properties.

Here, we define the family of local properties for our purpose. For any $D \in \mathfrak{D}$ and any $\mathbf{x} \coloneqq (x_1, \ldots, x_k) \in \mathcal{X}^k$ with disjoint entries, we defined the following 1-local properties $L_i^{D,\mathbf{x}} \subseteq \overline{\mathcal{Y}}$[14] with support $\{x_i\}$ for $i \in [k]$ as

$$L_i^{D,\mathbf{x}} \coloneqq L_{i,1}^{D,\mathbf{x}} \cup L_{i,2}^{D,\mathbf{x}},$$

where

$$L_{i,1}^{D,\mathbf{x}} \coloneqq \{x \in \mathcal{X} \mid D(x) \neq \perp \vee x \in \{x_1, \ldots, x_k\}\}$$

and

$$L_{i,2}^{D,\mathbf{x}} \coloneqq \{x \in \mathcal{X} \mid \exists x', x'' \in L_{i,1}^{D,\mathbf{x}} \colon x = x' \oplus x''\}.$$

The following lemma, in line with Lemma 2.1 in [16], shows that the local properties $\{L_i^{D,\mathbf{x}}\}$ recognize the database transition $\neg\mathsf{TCHN}^{\mathsf{s}} \to \mathsf{TCHN}^{\mathsf{s}+1}$, and allows us to exploit Theorem 28. First, we briefly explain the intuition. We pick an arbitrary $(s+1)$-chain in $D[\mathbf{x} \mapsto \mathbf{u}]$ and call it the *new chain*. We denote the elements of the new chain by $\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{s+1}$. There are two possible consequences of this database transition:

First, some branch x_i of the query $\mathbf{x}$ becomes the first elements $\hat{x}_0$ of the new chain, i.e., $x_i = \hat{x}_0$ and x_i is responded with $D[\mathbf{x} \mapsto \mathbf{u}](x_i) = u_i$ such that $u_i = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_0) = \hat{x}_1$. In addition, $D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_1) = \hat{x}_0 \oplus \hat{x}_2 \neq \perp$. This means that $\hat{x}_1$ must either already be sampled ($D(\hat{x}_1) \neq \perp$) or be one of the branch of $\mathbf{x}$ ($\hat{x}_1 \in \{x_1, \ldots, x_k\}$) (or both). In other words, u_i must be in $L_{i,1}^{D,\mathbf{x}}$.

Second, some branch x_i of the query $\mathbf{x}$ becomes the $(j+1)$-th elements $\hat{x}_j$ $(1 \leq j \leq s)$ of the new chain, i.e., $x_i = \hat{x}_j$ and x_i is responded with $D[\mathbf{x} \mapsto \mathbf{u}](x_i) = u_i$ such that $u_i = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_j) = \hat{x}_{j-1} \oplus \hat{x}_{j+1}$. Similarly, we can conclude that either $D(\hat{x}_{j-1}) \neq \perp$ or $\hat{x}_{j-1} \in \{x_1, \ldots, x_k\}$ (or both) and either $D(\hat{x}_{j+1}) \neq \perp$ or $\hat{x}_{j+1} \in \{x_1, \ldots, x_k\}$ (or both). This means that u_i must be the XOR of two elements in $L_{i,1}^{D,\mathbf{x}}$. That is, $u_i \in L_{i,2}^{D,\mathbf{x}}$.

The intuition above can be formalized as the following lemma.

▶ **Lemma 29.** $D[\mathbf{x} \mapsto \mathbf{r}] \notin \mathsf{TCHN}^{\mathsf{s}} \wedge D[\mathbf{x} \mapsto \mathbf{u}] \in \mathsf{TCHN}^{\mathsf{s}+1} \implies \exists i \in [k] \colon r_i \neq u_i \wedge u_i \in L_i^{D,\mathbf{x}}$.

Proof. Suppose $D[\mathbf{x} \mapsto \mathbf{u}] \in \mathsf{TCHN}^{\mathsf{s}+1}$ and let $\hat{x}_0, \hat{x}_1, \ldots, \hat{x}_{s+1}$ be such a chain, i.e., $\hat{x}_1 = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_0)$ and $\hat{x}_{j+2} = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{j+1}) \oplus \hat{x}_j$ for $j = 0, 1, \ldots, q-1$. Let $s_\circ$ be the smallest j such that $D[\mathbf{x} \mapsto \mathbf{r}](\hat{x}_{s_\circ}) \neq D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ})$. If $s_\circ \geq s$ or j does not exist, then $D[\mathbf{x} \mapsto \mathbf{r}] \in \mathsf{TCHN}^{\mathsf{s}+1}$, and we are done. Suppose now $0 \leq s_\circ \leq s-1$. Since $D[\mathbf{x} \mapsto \mathbf{u}]$ and $D[\mathbf{x} \mapsto \mathbf{u}]$ are identical outside of $\mathbf{x}$, there exists a coordinate i of $\mathbf{x}$ such that $x_i = \hat{x}_{s_\circ}$. Therefore, $u_i = D[\mathbf{x} \mapsto \mathbf{u}](x_i) = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ}) \neq D[\mathbf{x} \mapsto \mathbf{r}](\hat{x}_{s_\circ}) = D[\mathbf{x} \mapsto \mathbf{r}](x_i) = r_i$.

Below, we divide the analysis into three cases according to the value of $s_\circ$:

1. If $s_\circ = 0$, we have $u_i = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ}) = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_0) = \hat{x}_1$. And $D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_1) = \hat{x}_2 \oplus \hat{x}_0 \neq \perp$, which implies either $D(\hat{x}_1) \neq \perp$ or $\hat{x}_1 \in \{x_1, \ldots, x_k\}$ (or both). This means $u_i \in L_{i,1}^{D,\mathbf{x}}$.
2. If $s_\circ = 1$, we have $u_i = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ}) = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_1) = \hat{x}_2 \oplus \hat{x}_0$. And $D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_2) = \hat{x}_3 \oplus \hat{x}_1 \neq \perp$, which implies either $D(\hat{x}_2) \neq \perp$ or $\hat{x}_2 \in \{x_1, \ldots, x_k\}$ (or both). And $D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_0) = \hat{x}_1 \neq \perp$, which implies either $D(\hat{x}_0) \neq \perp$ or $\hat{x}_0 \in \{x_1, \ldots, x_k\}$ (or both). This means $u_i \in L_{i,2}^{D,\mathbf{x}}$.

[14] Recall that we assume $\mathcal{X} = \mathcal{Y}$.

3. If $2 \le s_\circ \le s-1$, then $r_i = D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ}) = \hat{x}_{s_\circ+1} \oplus \hat{x}_{s_\circ-1}$. Similarly, $D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ+1}) = \hat{x}_{s_\circ} \oplus \hat{x}_{s_\circ+2} \neq \perp$, which implies either $D(\hat{x}_{s_\circ+1}) \neq \perp$ or $\hat{x}_{s_\circ+1} \in \{x_1, \dots, x_k\}$ (or both). And $D[\mathbf{x} \mapsto \mathbf{u}](\hat{x}_{s_\circ-1}) = \hat{x}_{s_\circ} \oplus \hat{x}_{s_\circ-2} \neq \perp$, which implies either $D(\hat{x}_1) \neq \perp$ or $\hat{x}_1 \in \{x_1, \dots, x_k\}$ (or both). This means $u_i \in L_{i,2}^{D,\mathbf{x}}$.

In all of the above cases, u_i must be in $L_i^{D,\mathbf{x}}$ which concludes the proof. ◀

We need Corollary 4.2 in [16] which is rephrased from Lemma 5 in [38].

▶ **Lemma 30** (Lemma 5 in [38]). *Let $R \subseteq \mathcal{X}^\ell \times \mathcal{Y}^\ell$ be a relation. Let $\mathcal{A}$ be an algorithm that outputs $\mathbf{x} \in \mathcal{X}^\ell$ and $\mathbf{y} \in \mathcal{Y}^\ell$. Let p be the probability that $\mathbf{y} = H(\mathbf{x}) \coloneqq (H(x_1), \dots, H(x_\ell))$ and $(\mathbf{x}, \mathbf{y}) \in R$ when $\mathcal{A}$ has interacted with the standard random oracle, initialized with a uniformly random function H. Similarly, let p' be the probability that $\mathbf{y} = D(\mathbf{x}) \coloneqq (D(x_1), \dots, D(x_\ell))$ and $(\mathbf{x}, \mathbf{y}) \in R$ when $\mathcal{A}$ has interacted with the compressed oracle and D is obtained by measuring its internal state in the computational basis. Then*

$$\sqrt{p} \le \sqrt{p'} + \sqrt{\frac{\ell}{|\mathcal{Y}|}}.$$

▶ **Lemma 31.** $[\![\perp \xrightarrow{q,k} \mathsf{TCHN}^{q+1}]\!] \le qek\sqrt{\frac{5kq(kq+1)}{|\mathcal{Y}|}}$.

Proof. By Lemma 5.6 in [16], we have

$$[\![\perp \xrightarrow{q,k} \mathsf{TCHN}^{q+1}]\!] \le \sum_{s=1}^{q} [\![\mathsf{SZ}_{\le k(s-1)} \setminus \mathsf{TCHN}^{\mathsf{s}} \xrightarrow{k} \mathsf{TCHN}^{\mathsf{s}+1}]\!].$$

Choosing the local properties $\{L_i^{D,\mathbf{x}}\}$ as above whenever $D \in \mathsf{SZ}_{\le k(s-1)}$, and to be constant-false otherwise, Lemma 29 ensures that we can apply Theorem 5.23 in [16] to bound quantum transition capacity. Therefore, applying Theorem 5.23 in [16], for each $s \in [q]$ we have

$$[\![\mathsf{SZ}_{\le k(s-1)} \setminus \mathsf{TCHN}^{\mathsf{s}} \xrightarrow{k} \mathsf{TCHN}^{\mathsf{s}+1}]\!] \le e \max_{D,\mathbf{x}} \sum_{i=1}^{k} \sqrt{10 \Pr[U \in L_i^{D,\mathbf{x}}]} \le ek\sqrt{\frac{5kq(kq+1)}{|\mathcal{Y}|}}.$$

The last inequality holds because for every $s \in [q]$ and every $D \in \mathsf{SZ}_{\le k(s-1)}$, it holds that

$$|\{x \in \mathcal{X} \mid D(x) \neq \perp\} \cup \{x_1 \dots, x_k\}| \le k(q-1) + k = kq.$$

Thus, we have $|L_{i,1}^{D,\mathbf{x}}| \le kq$ and $|L_{i,2}^{D,\mathbf{x}}| \le \binom{kq}{2} = kq(kq-1)/2$ for $i \in [k]$, which then implies $|L_i^{D,\mathbf{x}}| \le kq(kq+1)/2$. Finally, summing over $s \in [q]$ completes the proof. ◀

First, note that Lemma 30 is tailored for algorithms that output *all elements of the chain and their hash values*. However, to obtain a bound when the algorithm $\mathcal{A}$ is required to output *only* x_0, x_q *and* x_{q+1} is more challenging. In most of situations, one could define another algorithm $\mathcal{B}$ that simply runs $\mathcal{A}$ followed by calculating the whole chain with $q+2$ extra queries. This would increase the number of queries by at most $q+2$. However, this gives us a meaningless bound for the twisted hash chain problem.

Below, we first provide the following lemma for algorithms that do *not* have to output the last hash value y_{q+1}. The proof is similar to Theorem 5.9 in [16].

▶ **Lemma 32.** *For any k-parallel q-query oracle algorithm $\mathcal{A}$ that interacts with a standard random oracle, the probability $p_{\mathcal{A}}$ (parameterized by k and q) that $\mathcal{A}$ outputs $x_0, x_1, \dots, x_{q+1} \in \mathcal{X}$ and $y_0, y_1, \dots, y_q \in \mathcal{Y}$ (without y_{q+1}) satisfying*

- $y_i = H(x_i)$ *for* $0 \le i \le q$
- $y_{i-1} = x_i \oplus x_{i-2}$ *for* $i \in [q+1]$, *where* $x_{-1} := 0^n$

is upper bounded by the function $F(k,q)$ *where*

$$F(k,q) := \left(qek\sqrt{\frac{5kq(kq+1)}{|\mathcal{Y}|}} + e(q+2)\sqrt{\frac{5(q+2)(q+3)}{|\mathcal{Y}|}} + \sqrt{\frac{q+2}{|\mathcal{Y}|}} \right)^2 = O\left(\frac{q^4k^4}{|\mathcal{Y}|}\right).$$

Proof. We define an algorithm $\mathcal{B}$ that runs $\mathcal{A}$ and obtains the output $x_0, \dots, x_{q+1}, y_0, \dots, y_q$. And then $\mathcal{B}$ makes a classical query x_{q+1} to the random oracle. Finally, $\mathcal{B}$ outputs $x_0, \dots, x_{q+1}, y_0, \dots, y_q, H(x_{q+1})$.

Let $p_{\mathcal{B}}$ be the probability that $\mathcal{B}$ outputs $x_0, x_1, \dots, x_{q+1} \in \mathcal{X}$ and $y_0, y_1, \dots, y_{q+1} \in \mathcal{Y}$ satisfying

- $y_i = H(x_i)$ for $0 \le i \le q+1$
- $y_{i-1} = x_i \oplus x_{i-2}$ for $i \in [q+1]$, where $x_{-1} := 0^n$

when $\mathcal{B}$ interacts with the standard random oracle.

Let $p'_{\mathcal{B}}$ be the probability that $\mathcal{B}$ outputs $x_0, x_1, \dots, x_{q+1} \in \mathcal{X}$ and $y_0, y_1, \dots, y_{q+1} \in \mathcal{Y}$ satisfying

- $y_i = D(x_i)$ for $0 \le i \le q+1$
- $y_{i-1} = x_i \oplus x_{i-2}$ for $i \in [q+1]$, where $x_{-1} := 0^n$

when $\mathcal{B}$ interacts with the compressed oracle.

Let $\overline{p}'_{\mathcal{B}}$ be the probability that $\mathcal{B}$ outputs $x_0, x_1, \dots, x_{q+1} \in \mathcal{X}$ and $y_0, y_1, \dots, y_{q+1} \in \mathcal{Y}$ satisfying

- $D(x_{i-1}) = x_i \oplus x_{i-2}$ for $i \in [q+1]$, where $x_{-1} := 0^n$

when $\mathcal{B}$ interacts with the compressed oracle.

We trivially have $p_{\mathcal{A}} = p_{\mathcal{B}}$ and $p'_{\mathcal{B}} \le \overline{p}'_{\mathcal{B}}$. Since $\mathcal{B}$ now outputs all the hash values as well, we can apply Lemma 30 to $\mathcal{B}$ which gives

$$\sqrt{p_{\mathcal{B}}} \le \sqrt{p'_{\mathcal{B}}} + \sqrt{\frac{q+2}{|\mathcal{Y}|}}.$$

In the rest of the proof, it remains to bound $\overline{p}'_{\mathcal{B}}$.

$$\begin{aligned}
\sqrt{\overline{p}'_{\mathcal{B}}} &\le \sup_{U_1,\dots,U_q} \left\| \sum_{\mathbf{x}} \mathsf{TCHN}^{q+1}_{\mathbf{x}}(|\mathbf{x}\rangle\langle\mathbf{x}| \otimes \mathsf{cO}_{x_{q+1}}) U_q \mathsf{cO}^k U_{q-1} \mathsf{cO}^k \dots U_1 \mathsf{cO}^k \perp \right\| \\
&\le \left\| \sum_{\mathbf{x}} \mathsf{TCHN}^{q+1}_{\mathbf{x}}(|\mathbf{x}\rangle\langle\mathbf{x}| \otimes \mathsf{cO}_{x_{q+1}}) \neg\mathsf{TCHN}^{q+1} \right\| \\
&\quad + \sup_{U_1,\dots,U_q} \|\mathsf{TCHN}^{q+1} U_q \mathsf{cO}^k U_{q-1} \mathsf{cO}^k \dots U_1 \mathsf{cO}^k \perp\| \\
&\le \max_{\mathbf{x}} \|\mathsf{TCHN}^{q+1}_{\mathbf{x}} \mathsf{cO}_{x_{q+1}} \neg\mathsf{TCHN}^{q+1}\| + [\![\perp \xRightarrow{q,k} \mathsf{TCHN}^{q+1}]\!] \\
&\le \max_{\mathbf{x}} \|\mathsf{TCHN}^{q+1}_{\mathbf{x}} \mathsf{cO}_{x_{q+1}} \neg\mathsf{TCHN}^{q+1}_{\mathbf{x}}\| + [\![\perp \xRightarrow{q,k} \mathsf{TCHN}^{q+1}]\!],
\end{aligned}$$

where the summation is over all $\mathbf{x} = (x_0, \dots, x_{q+1}) \in \mathcal{X}^{q+2}$; $\{|\mathbf{x}\rangle\langle\mathbf{x}|\}$ denotes the measurement acting on $\mathcal{B}$'s output register to produce the output $\mathbf{x}$; the database property $\mathsf{TCHN}^{q+1}_{\mathbf{x}}$ is defined as

$$\mathsf{TCHN}^{q+1}_{\mathbf{x}} := \{D \mid x_i = D(x_{i-1}) \oplus x_{i-2} \text{ for } i \in [q+1]\} \subseteq \mathfrak{D}.$$

That is, the sequence $x_0, \dots, x_{q+1}$ forms a $(q+1)$-chain.

Now, notice that for every $\mathbf{x} \in \mathcal{X}^{q+2}$,

$$\begin{aligned}
&\|\mathsf{TCHN}^{q+1}_{\mathbf{x}}\mathsf{cO}_{x_{q+1}}\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}}\| \\
=&\|\mathsf{TCHN}^{q+1}_{\mathbf{x}}(\mathsf{cO}_{x_{q+1}} \otimes \mathsf{cO}_{x_q\hat{0}} \otimes \cdots \otimes \mathsf{cO}_{x_0\hat{0}})\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}}\| \\
\leq &\max_{\hat{\mathbf{y}}} \|\mathsf{TCHN}^{q+1}_{\mathbf{x}}\mathsf{cO}_{\mathbf{x}\hat{\mathbf{y}}}\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}}\| \leq [\![\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}} \xrightarrow{q+2} \mathsf{TCHN}^{q+1}_{\mathbf{x}}]\!],
\end{aligned}$$

where the first equality holds since $\mathsf{cO}_{x\hat{0}}$ is equal to the identity operator for every $x \in \mathcal{X}$.

Following similar arguments as in Lemma 29, we now show there exist local properties that recognize the database transition $[\![\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}} \xrightarrow{q+2} \mathsf{TCHN}^{q+1}_{\mathbf{x}}]\!]$. For any tuple $\mathbf{x} = (x_1, \ldots, x_{q+2})$ with pairwise distinct entries, any tuple $\mathbf{x}' = (x'_0, \ldots, x'_{q+1})$[15] and database $D \in \mathfrak{D}$, we define the following local properties for $i \in [q+2]$

$$L_i^{\mathbf{x},D} \coloneqq \{x'_0, \ldots, x'_{q+1}\} \cup \{x \mid \exists a, b \in \{1, \ldots, q+2\} \colon x = x'_a \oplus x'_b\}.$$

Note that $|L_i^{\mathbf{x},D}| \leq (q+2) + \binom{q+2}{2} = (q+2)(q+3)/2$ for each $i \in [q+2]$.

Suppose $D[\mathbf{x} \mapsto \mathbf{r}] \notin \mathsf{TCHN}^{q+1}_{\mathbf{x}'}$ yet $D[\mathbf{x} \mapsto \mathbf{u}] \in \mathsf{TCHN}^{q+1}_{\mathbf{x}'}$. Then $\{x'_0, \ldots, x'_{q+1}\}$ is a $(q+1)$-chain. Let $s_\circ$ be the smallest j such that $D[\mathbf{x} \mapsto \mathbf{r}](x'_j) \neq D[\mathbf{x} \mapsto \mathbf{u}](x'_j)$. If $s_\circ = q+1$ or j does not exist, then $D[\mathbf{x} \mapsto \mathbf{r}] \in \mathsf{TCHN}^{q+1}_{\mathbf{x}'}$ and we are done. So we assume $0 \leq s_\circ \leq q$. Since $D[\mathbf{x} \mapsto \mathbf{r}]$ coincides $D[\mathbf{x} \mapsto \mathbf{u}]$ outside of $\mathbf{x}$, there must exists an index $i \in [q+2]$ such that $x_i = x'_{s_\circ}$. Therefore, we have $r_i = D[\mathbf{x} \mapsto \mathbf{r}](x_i) = D[\mathbf{x} \mapsto \mathbf{r}](x'_{s_\circ}) \neq D[\mathbf{x} \mapsto \mathbf{u}](x'_{s_\circ}) = D[\mathbf{x} \mapsto \mathbf{u}](x_i) = u_i$.

In addition, if $s_\circ = 0$, then $u_i = D[\mathbf{x} \mapsto \mathbf{u}](x'_0) = x'_1 \in \{x'_0, \ldots, x'_{q+1}\}$. If $1 \leq s_\circ \leq q$, then $u_i = D[\mathbf{x} \mapsto \mathbf{u}](x'_{s_\circ}) = x'_{s_\circ - 1} \oplus x'_{s_\circ + 1}$ which means u_i is the XOR of two distinct elements in $\{x'_0, \ldots, x'_{q+1}\}$. In either case, u_i must lie in $L_i^{\mathbf{x},D}$. Therefore, by Theorem 5.23 in [16], for every $\mathbf{x}' \in \mathcal{X}^{q+2}$ we have

$$[\![\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}'} \xrightarrow{q+2} \mathsf{TCHN}^{q+1}_{\mathbf{x}'}]\!] \leq e(q+2)\sqrt{\frac{5(q+2)(q+3)}{|\mathcal{Y}|}}.$$

Thus, we can bound $\max_{\mathbf{x}'} \|\mathsf{TCHN}^{q+1}_{\mathbf{x}'}\mathsf{cO}_{x'_{q+1}}\neg\mathsf{TCHN}^{q+1}_{\mathbf{x}'}\|$ by the above quantity.

Putting things together, we have

$$p_{\mathcal{A}} \leq \left([\![\bot \xrightarrow{q,k} \mathsf{TCHN}^{q+1}]\!] + e(q+2)\sqrt{\frac{5(q+2)(q+3)}{|\mathcal{Y}|}} + \sqrt{\frac{q+2}{|\mathcal{Y}|}}\right)^2$$

Bounding the first term by Lemma 31, this concludes the proof. ◀

Now, we are ready to prove the main theorem.

Proof of Theorem 24. We finish the proof by reduction. Define the algorithm $\mathcal{D}$ as follows:

1. Run $\mathcal{C}$ and obtain x_0, x_q and x_{q+1}.
2. For $i \in [q-1]$:
   - Make a classical 2-parallel query (x_{i-1}, x_{q+i}) to the random oracle and then obtain $(H(x_{i-1}), H(x_{q+i}))$.
   - Set $x_i \coloneqq H(x_{i-1}) \oplus x_{i-2}$ and $x_{q+i+1} \coloneqq H(x_{q+i}) \oplus x_{q+i-1}$.
3. Make a classical 2-parallel query (x_{q-1}, x_{2q}) to the random oracle and then obtain $(H(x_{q-1}), H(x_{2q}))$.
4. Output $x_0, x_1, \ldots, x_{2q+1}$ and $H(x_0), H(x_1), \ldots, H(x_{q-1}), H'(x_q), H(x_{q+1}), \ldots, H(x_{2q})$, where $H'(x_q) \coloneqq x_{q-1} \oplus x_{q+1}$[16].

[15] In the rest of the proof, we switch the variable $\mathbf{x}$ of $\mathsf{TCHN}^{q+1}_{\mathbf{x}}$ into $\mathbf{x}'$ for convenience.

[16] Note that in Step 2 and 3, $\mathcal{D}$ makes a total of $2q$ queries including $x_0, \ldots, x_{2q}$ except x_q.

First, it is trivial that $p_{\mathcal{C}}$ is equivalent to the probability $p_{\mathcal{D}}$ that $H'(x_q) = H(x_q)$ and $\mathcal{D}$ outputs a $(2q+1)$-chain. Now, we calculate the total number of queries made by $\mathcal{D}$. In Step 1, $\mathcal{D}$ makes q k-parallel queries to execute $\mathcal{C}$. In Steps 2 and 3, $\mathcal{D}$ makes q 2-parallel queries. To sum up, $\mathcal{D}$ makes a total of $2q$ k-parallel queries. By Lemma 32, the probability $p_{\mathcal{D}}$ is at most $F(k, 2q)$. Therefore, this finishes the proof. ◀

Considering the situation in which the algorithm is assigned to a particular starting point $x_0 \in \mathcal{X}$ of the chain, we have the following corollary which is trivially implied by Theorem 24.

▶ **Corollary 33.** *For any k-parallel q-query oracle algorithm $\mathcal{E}$, the probability $p_{\mathcal{E}}$ (parameterized by k and q) that the algorithm takes a uniformly random $x_0 \in \mathcal{X}$ as input, and outputs $x_q, x_{q+1} \in \mathcal{X}$ satisfying*

- *there exist $x_1, \ldots, x_{q-1} \in \mathcal{X}$ such that $H(x_{i-1}) = x_i \oplus x_{i-2}$ for $i \in [q+1]$, where $x_{-1} := 0^n$*

is at most $F(k, 2q) = O(k^4 q^4 / |\mathcal{Y}|)$, where the function F is defined in Lemma 32.

Proof. We finish the proof by reduction. Let $\mathcal{C}$ be the algorithm that first samples $x_0 \in \mathcal{X}$ uniformly at random and invokes $\mathcal{E}(x)$. $\mathcal{C}$ responds to every $\mathcal{E}$'s oracle query by its oracle access directly. Then $\mathcal{C}$ outputs whatever $\mathcal{E}$ outputs.

Let $p_{\mathcal{C}}$ be the probability defined as in Theorem 24. Since the success of $\mathcal{E}$ implies the success of $\mathcal{C}$, we have $p_{\mathcal{C}} \geq p_{\mathcal{E}}$. By Theorem 24 and the construction of $\mathcal{C}$, we have $F(k, 2q) \geq p_{\mathcal{C}}$, which concludes the proof. ◀

▶ Remark 34. Here, we explain the challenging issue of our case. Given only x_0, x_q and x_{q+1}, in order to output the whole chain, the algorithm cannot make the query in parallel but is required to make *adaptive* queries. For example, to reveal the next point $x_1 = H(x_0)$, the algorithm must first query x_0. Therefore, we cannot use Theorem 5.9 in [16] in a black-box way.

7 Quantum Walk on a Line

Our proof of Hamiltonian simulation lower bound relies on the continuous-time quantum walk on a line [14]. We introduce quantum walks on a line in this section.

Consider a particle moving on a graph, which is a line with L vertices. Each vertex on the line is labeled by an integer $1, 2, \ldots L$. We use a quantum state $|j\rangle$ to denote the particle locating at the vertex j. Figure 4(a) illustrates our system, a finite segment with length L. We let the Hamiltonian H_L of the system be the adjacency matrix of the graph. In physics terminology, H_L couples adjacent vertices with the coupling constant 1. We have

$$H_L = \sum_{j=1}^{L-1} |j\rangle\langle j+1| + |j+1\rangle\langle j|, \tag{29}$$

or

$$H_L = \begin{pmatrix} 0 & 1 & 0 & & & 0 \\ 1 & 0 & 1 & \cdots & \cdots & 0 \\ 0 & 1 & 0 & & & 0 \\ & \vdots & & \ddots & & \vdots \\ & \vdots & & & 0 & 1 \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{pmatrix}.$$

in the $\{|j\rangle\}_{j=1}^{L}$ basis.

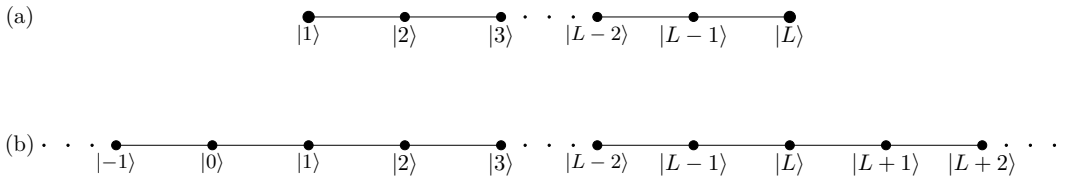


Figure 4 Quantum walk on a line. (a) Quantum walk on a finite segment with length L. (b) Quantum walks on an infinite line.

The dynamics of the particle are determined by the time evolution operator $e^{-iH_L t}$, where t is the evolution time. We call the dynamics of the system "quantum walk on a line."

We are interested in the dynamics of a particle initially at the end of the line. In other words, we consider the evolution of a particle under H_L with the initial state $|1\rangle$. We have the following result.

▶ **Lemma 35.** *Given a system that evolves under the Hamiltonian H_L described in* (29) *with initial state $|1\rangle$, if the system is measured at time $t \in [0, L/2]$ in the $\{|j\rangle\}_{j=0}^{L}$ basis with outcome l, the probability that $l > t$ is at least $1/3$.*

Before the formal proof of Lemma 35, we first discuss the general behavior of the quantum walk. Let the particle initially locate at $|k\rangle$ and evolve under the Hamiltonian H_L. When measuring the system at time t in the $\{|j\rangle\}_{j=1}^{L}$ basis, the probability $P(k, l, t)$ of measurement outcomes being l is

$$P(k,l,t) = \left|\langle l|e^{-iH_L t}|k\rangle\right|^2. \tag{30}$$

By diagonalizing H_L, we can calculate $P(k, l, t)$ as follows:

$$P(k,l,t) = \left|\langle l|e^{-iH_L t}|k\rangle\right|^2 = \sum_{p,q=1}^{L} e^{-i(\lambda_p-\lambda_q)t} v_l^{(p)} v_k^{(p)*} v_l^{(q)*} v_k^{(q)}, \tag{31}$$

where λ_p's are the eigenvalues of H_L – each with the corresponding eigenstate $|v^{(p)}\rangle = \sum_{j=1}^{L} v_j^{(p)}|j\rangle$. The eigenvalues and the eigenstates of H_L have a closed-form expression. That is, $\lambda_p = 2\cos(\frac{p\pi}{L+1})$ and $v_j^{(p)} = \sqrt{\frac{2}{L+1}}\sin(\frac{jp\pi}{L+1})$ [30] [17].

We use the propagation of the wave function of a free particle to analogize the quantum walk.[18] For example, we plot the result of the quantum walk on a segment of length $L = 100$ in Figure 5. The initial state is $|k = 1\rangle$ and we focus on the time interval $t \in [0, L/2]$. Figure 5(a) shows $P(1, l, t)$, the probability of obtaining the measurement outcome $|l\rangle$, for

[17] In fact, there is a simpler form of $P(k, l, t)$: when $|l - k|$ is even,

$$P(k,l,t) = \Big(\sum_p \cos\left(2t\cos\left(\frac{p\pi}{L+1}\right)\right)\sin\left(\frac{kp\pi}{L+1}\right)\sin\left(\frac{lp\pi}{L+1}\right)\Big)^2,$$

and when $|l - k|$ is odd,

$$P(k,l,t) = \Big(\sum_p \sin\left(2t\cos\left(\frac{p\pi}{L+1}\right)\right)\sin\left(\frac{kp\pi}{L+1}\right)\sin\left(\frac{lp\pi}{L+1}\right)\Big)^2.$$

[18] Consider an extreme case that the distance between two adjacent vertices in space goes to zero and the length L goes to infinity. The system is reduced to free space.

every l at different time t. We see that at time t, the wavefront reaches $l \approx 2t$. Figure 5(b) shows the probability of getting the measurement outcome $|l\rangle$ for a *fixed* l versus time. We see that the probability is extremely small when $t \ll l/2$ and reaches the maximum at $t \approx l/2$. Finally, it behaves like a damped oscillation when $t \gtrsim l/2$. These observations suggest that the wavefront propagates at a constant speed, which gives a hint that the particle reaches the vertex $l = \Theta(t)$ at time t.[19]

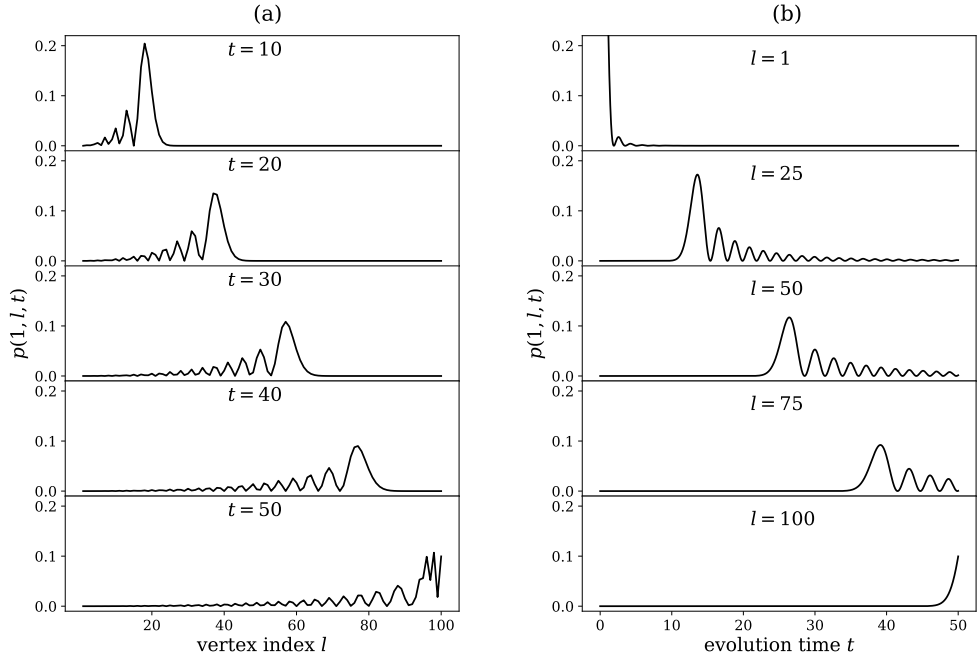


Figure 5 The result of quantum walk on a segment with $L = 100$ for the evolution time $t \in [0, L/2]$. The initial state is $|1\rangle$. (a) The probability of getting the outcomes $|l\rangle$ for every node l at the evolution time $t = 10, 20, 30, 40$ and 50 respectively. (b) The probability of getting the outcomes $l = 1, 25, 50, 75$ and 100 versus evolution time $t \in [0, L/2]$.

Next, we are going to prove Lemma 35. We take another approach instead of diagonalizing H_L directly. We follow the approach in [14]. Similar to solving "the particle in a box model" in quantum mechanics, we first find the homogeneous solution in free space and then find the particular solution that satisfies the boundary conditions and the initial conditions. (See, for example, [34].)

Consider the quantum walk on an infinite line which is illustrated in Figure 4(b). The Hamiltonian of the quantum walk on an infinite line is defined by

$$H_\infty := \sum_{j=-\infty}^{\infty} |j+1\rangle\langle j| + |j\rangle\langle j+1|, \tag{32}$$

[19] This corresponds to the fact that the uncertainty of the position of a free particle is linear in t. See, for example, [34].

and we define the propagator

$$G(k,l,t) := \langle l|e^{-iH_\infty t}|k\rangle. \tag{33}$$

The (sub-normalized) eigenstate of H_∞ is the momentum state $|p\rangle$. The momentum state has the following property

$$\langle j|p\rangle = e^{ipj}, -\pi \le p \le \pi. \tag{34}$$

The corresponding eigenvalue of $|p\rangle$ is $E_p = 2\cos p$. Hence, we have

$$\langle l|e^{-iH_\infty t}|k\rangle = \int_{-\pi}^{\pi} dp e^{-i2t\cos p + ip(l-k)} = i^{(l-k)}J_{l-k}(2t), \tag{35}$$

where $J_n(\cdot)$ is the Bessel function of order n. (See (1).)

Now we are ready to calculate the propagator of the quantum walk on a *finite* segment. We use $\widetilde{G}(k,l,t) := \langle l|e^{-iH_L t}|k\rangle$ to denote the propagator of the quantum walk on a finite segment. The propagator $\widetilde{G}$ is a superposition of G and $\widetilde{G}$ that satisfies the boundary conditions: $\widetilde{G}(k,0,t) = 0 = \widetilde{G}(k,L+1,t)$, and the initial condition $\widetilde{G}(k,l,0) = \delta_{kl}$.

The solution is

$$\widetilde{G}(1,l,t) = \sum_{m=-\infty}^{\infty} G(1,l+2m(L+1),t) - G(1,-l+2m(L+1),t). \tag{36}$$

The above equation (36) can be interpreted as the wave reflecting between the boundaries $j = 0$ and $j = L+1$.

We set the starting point $j = 1$. In the time interval that we are interested in, namely, $t \in [0, L/2]$, we have $G(1, \pm l + 2m(L+1), t) = J_{2m(L+1)\pm l-1}(2t)$ is exponentially small in L for $m \neq 0$. This is because the order $|2m(L+1) \pm l - 1| > L$ for $m \neq 0$ and the argument $2t \le L$ for $t \le L/2$. (See (4).)
Thus,

$$\begin{aligned}
\widetilde{G}(1,l,t) &\approx G(1,l,t) - G(1,-l,t) \\
&= i^{l-1}J_{l-1}(2t) - i^{-(l+1)}J_{-(l+1)}(2t) \\
&= i^{l-1}J_{l-1}(2t) - (i^{-(l+1)})(-1^{l+1})J_{l+1}(2t) \\
&= i^{l-1}J_{l-1}(2t) - (-i)^{l+1})J_{l+1}(2t) \\
&= i^{l-1}(J_{l-1}(2t) - (-i)^2 J_{l+1}(2t)) \\
&= i^{l-1}(J_{l-1}(2t) + J_{l+1}(2t)) \\
&= i^{l-1}\frac{l}{t}J_l(2t).
\end{aligned} \tag{37}$$

The third equation is due to the relation of negative order (2) of the Bessel function, and the last equation uses the recursion property (3) of the Bessel function. Then we have

$$P(1,l,t) = \left|\widetilde{G}(1,l,t)\right|^2 \approx \left(\frac{l}{t}\right)^2 J_l^2(2t). \tag{38}$$

As a remark, the probability $P(1,l,t)$ is almost independent of L when $t \in [L/2]$. It can be interpreted as the following: before the wavefront reaches the boundary, the wave propagates as in free space. Finally, we prove Lemma 35.

Proof of Lemma 35. We directly calculate the probability $\sum_{l=1}^{\lfloor t \rfloor} P(1,l,t)$ as follows:

$$\sum_{l=1}^{\lfloor t \rfloor} P(1,l,t) = \sum_{l=1}^{\lfloor t \rfloor} \left(\frac{l}{t}\right)^2 J_l^2(2t) \leq \sum_{l=1}^{\lfloor t \rfloor} \left(\frac{l}{t}\right)^2 \frac{2}{\pi}\frac{1}{l} = \frac{2}{\pi}\sum_{j=l}^{\lfloor t \rfloor} \frac{l}{t^2} \leq \frac{2}{\pi},$$

where the first equation follows from (38) and the second inequality follows from Lemma 15. As a result, we conclude that

$$\sum_{l=\lceil t \rceil}^{L} P(1,l,t) = 1 - \sum_{l=1}^{\lfloor t \rfloor} P(1,l,t) > \frac{1}{3}. \qquad \blacktriangleleft$$

8 No Fast-forwarding in Oracle Model: Unconditional Result

In this section, we are going to investigate the parallel lower bound of Hamiltonian simulation in the oracle model. In the oracle model, the Hamiltonian is expressed by a Hermitian matrix. There are many algorithms that can efficiently simulate a Hamiltonian in the oracle model if the Hamiltonian matrix is sparse [6–8, 13, 27, 28]. As a result, we are interested in the lower bound of simulating a sparse Hamiltonian. Besides, we normalize the Hamiltonian by setting the absolute value of every element of the Hamiltonian to be at most 1. The sparse Hamiltonian is defined as follows.

▶ **Definition 36** (Sparse Hamiltonian). *Let $H \in \mathbb{C}^{N\times N}$ denote a Hamiltonian acting on the Hilbert space with dimension N. We say H is d-sparse if there are at most d nonzero entries in every row.*

In the oracle setting, the simulation algorithm can only obtain the description of the Hamiltonian via oracle queries. In most of the models of the algorithms, there are two oracles that can be accessed: First, the *entry oracle*, denoted by $\mathcal{O}_H$, answers the value of the matrix element. Second, the *sparse structure oracle*, denoted by $\mathcal{O}_L$, answers the index of the nonzero entry. Let the Hamiltonian H that we want to simulate be acting on an N-dimensional Hilbert space and be d-sparse. When the entry oracle $\mathcal{O}_H$ is queried on the index (j,k) where $j,k \in [N]$, it returns the element value H_{jk}. When the sparse structure oracle is queried on (j,s) where $j \in [N]$ and $s \in [d]$, it returns k where H_{jk} is the s-th nonzero entry of the j-th row.

The algorithm can query these two oracles in superposition respectively. In the standard quantum oracle model, these two oracles are written as:

$$\mathcal{O}_H|j,k,z\rangle = |j,k,z \oplus H_{jk}\rangle, \tag{39}$$

and

$$\mathcal{O}_L|j,s\rangle = |j,k\rangle, \tag{40}$$

where k is the index of the s-th nonzero entry in the j-th row.

We are going to prove that simulating a quantum system for evolution time t requires at least $\Omega(t)$ parallel quantum queries. We have the following result.

▶ **Theorem 37** (Simulation lower bound in the oracle model). *For any integer n, any polynomial $T(\cdot)$ and $p = \mathrm{poly}(n)$, there exists a time-independent Hamiltonian $H \in \mathbb{C}^{(2^nT(n))\times(2^nT(n))}$ satisfies the following. For any quantum algorithm that can make p-parallel queries to the entry oracle $\mathcal{O}_H$ (defined in (39)) and the sparse structure oracle $\mathcal{O}_L$ (defined in (40)), simulating H for an evolution time $t \in [0, T(n)/2]$ within an error $\epsilon \leq 1/4$ needs at least $\Omega(t)$ p-parallel queries to $\mathcal{O}_H$ and $\mathcal{O}_L$ in total. Furthermore, H is 2-sparse and $|H_{jk}| \leq 1$ for every $j,k \in [2^nT(n)]$.*

Theorem 37 can be interpreted as simulating a system with $n + O(\log n)$ qubits for an evolution time $t < \mathrm{poly}(n)$ cannot be fast-forwarded.

Before the formal proof of Theorem 37, we first sketch our proof strategy. We modify the proof of the query lower bound in [6]. In [6], the parity problem is reduced to the Hamiltonian simulation problem. In particular, it is shown that if one can fast-forward the Hamiltonian simulation, then one can find the parity of an N-bit string with $o(N)$ queries. However, this technique cannot be extended to prove the parallel lower bound since finding the parity of a string is not parallel-hard. Instead, we reduce the permutation chain problem, of which the parallel hardness was already proven in Section 5, to the Hamiltonian simulation. We are going to show that there exists a specific Hamiltonian such that simulating the Hamiltonian implies solving the permutation chain problem.

We restate the permutation chain problem and its hardness below.

▶ **Definition 38** (Permutation chain). *Let $n \in \mathbb{N}$ and $p, L = \mathrm{poly}(n)$. For each $j \in [L]$, let $\Pi_j : \{0,1\}^n \to \{0,1\}^n$ be a random permutation and let Π_j^{-1} be the inverse of Π_j. Let $f^{(j)}(\cdot)$ denote $\Pi_j(\Pi_{j-1}(\cdots\Pi_1(\cdot)))$. A quantum algorithm can make p-parallel query to both Π_j and Π_j^{-1} for each $j \in [L]$ respectively and is asked to output $x_q \in \{0,1\}^n$ such that $x_q = f^{(q)}(0^n)$, where $q \in [L]$.*

▶ **Corollary 39** (Hardness of permutation chain). *Let $n \in \mathbb{N}$, and $p, L = \mathrm{poly}(n)$. For each $j \in [L]$, let Π_j and Π_j^{-1} be a random permutation over n-bit strings and its inverse. Let $f^{(j)}(\cdot) := \Pi_j(\Pi_{j-1}(\cdots\Pi_1(\cdot)))$ be the function defined in Definition 38. For any $t, q \in [L]$ and any quantum algorithm $\mathcal{A}$ that makes t p-parallel queries to Π_j and Π_j^{-1}, the probability that $\mathcal{A}$ outputs $x_q \in \{0,1\}^n$ satisfying $x_q = f^{(q)}(0^n)$ and $t < q$ is negligible in n.*

Proof. Let $\bar{x}_q \coloneqq f^{(q)}(0^n)$ for each $q \in [L]$. The probability that $\mathcal{A}$ outputs $\bar{x}_q$ such that $t < q$ is given by

$$\sum_{j=t+1}^{L} \Pr[\mathcal{A} \text{ outputs } \bar{x}_j].$$

By Theorem 18, for any quantum algorithm A that makes t p-parallel queries to Π_j and Π_j^{-1}, the probability that $\mathcal{A}$ outputs x_j such that $j > t$ is $O(t\sqrt{p/2^n})$. Hence, the probability

$$\sum_{j=t+1}^{L} \Pr[\mathcal{A} \text{ outputs } \bar{x}_j] = \mathrm{poly}(n) \cdot O\left(t\sqrt{\frac{p}{2^n}}\right)$$

is negligible in n. ◀

Similar to [6], we use quantum walk on a graph to solve the underlying hard problem. We construct a graph that consists of L columns where there are 2^n vertices in each column. Each vertex in the j-th column is labelled by (j, x), where $j \in \{0, 1, \ldots L\}$ and $x \in \{0,1\}^n$. The label is translated as follows: after j queries, the output string is x. The vertices in the j-th column are only adjacent to the vertices that are in the $(j \pm 1)$-th columns. Furthermore, the vertices (j, x) and $(j+1, x')$ (resp., $(j-1, x')$) are adjacent if and only if $x' = \Pi_{j+1}(x)$ (resp. $\Pi_j^{-1}(x)$). Because each Π_j is a permutation, the graph consists of 2^n disconnected lines of length L. If the vertices (j, x) and $(0, x_0)$ are connected, it holds that $x = f^{(j)}(x_0)$.

In Figure 6, we presents a toy example: let $\Pi_j : \{0,1\}^2 \to \{0,1\}^2$ for each $j \in [L]$ and each $\Pi_j = \Pi$ has the same truth table, i.e.,

$$\Pi(00) = (01),\ \Pi(01) = (10),\ \Pi(10) = (11),\ \text{and } \Pi(11) = (00).$$

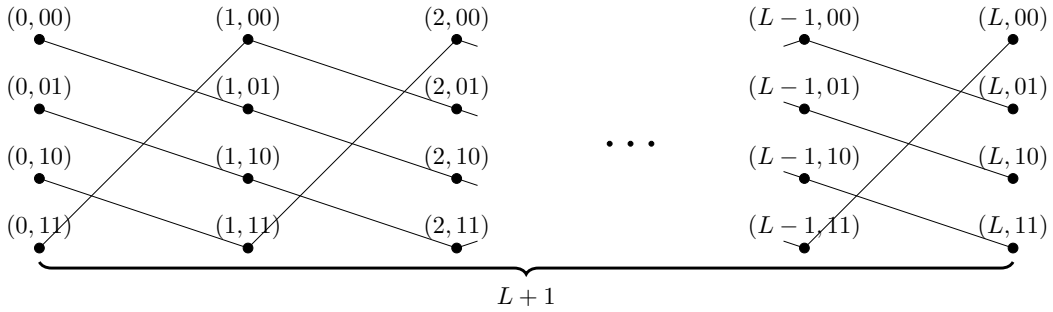


Figure 6 Using quantum walk to solve the permutation chain problem: a toy example.

We let the Hamiltonian H that determines the behavior of the quantum walk be the adjacency matrix of the graph.[20] That is,

$$H = \sum_{j=0}^{L-1} \sum_{x\in\{0,1\}^n} |j+1, \Pi_j(x)\rangle\langle j, x| + |j, x\rangle\langle j+1, \Pi_j(x)|$$
$$= \sum_{x_0\in\{0,1\}^n} \sum_{j=0}^{L-1} |j+1, f^{(j+1)}(x_0)\rangle\langle j, f^{(j)}(x_0)| + |j, f^{(j)}(x_0)\rangle\langle j+1, f^{(j+1)}(x_0)|. \quad (41)$$

Because two vertices on different lines are decoupled, we have the following observation.

▶ **Observation 40.** *If the random walk starts at the vertex $(0, x_0)$, then it always walks on the same line. To be more precise, if a system evolves under the Hamiltonian H described in (41) and the initial state is $|0, x_0\rangle$, then at any time t, the quantum state of the system is in the subspace* $\mathrm{Span}\left(\left\{|j, f^{(j)}(x_0)\rangle\right\}_{j=0}^{L}\right)$.

Observation 40 can be verified by taking the Taylor expansion of the time evolution operator: $e^{-iHt} = \sum_{k=0}^{\infty}(-iHt)^k/k!$.

To solve the permutation chain problem, we use a Hamiltonian simulation algorithm to simulate the quantum walk under the Hamiltonian H with initial state $|0, 0^n\rangle$. When we measure the system at time t and get the outcome (q, x). The string x is a potential solution to the permutation chain problem. We aim to prove the following two statements. First, the oracles $\mathcal{O}_H$ and $\mathcal{O}_L$ can be simulated efficiently by Π_j and Π_j^{-1}. Second, the probability of getting a measurement outcome (q, x) at time t such that $q \geq t$ is high. Combining these two statements, we have the following conclusion. If an algorithm can simulate H for an evolution time t with $o(t)$ queries, then we can solve the permutation chain problem with $o(t)$ queries as well. However, this violates the hardness of the permutation chain problem.

Now we are ready to present the formal proof of Theorem 37.

Proof of Theorem 37. We construct a time-independent Hamiltonian H acting on a $2^n(L+1)$-dimensional Hilbert space where $L + 1 = f(n)$. The basis vector of the $2^n(L+1)$-dimensional Hilbert space is denoted by $|j, x\rangle$ where $j \in \{0, 1, \ldots, L\}$ and $x \in \{0, 1\}^n$. The element of H is defined as follows.

$$\langle j', x'|H|j, x\rangle = \begin{cases} 1, & \text{if } j' = j+1 \text{ and } x' = \Pi_{j+1}(x) \\ 1, & \text{if } j' = j-1 \text{ and } x' = \Pi_j^{-1}(x) \\ 0, & \text{otherwise.} \end{cases} \quad (42)$$

[20] Our Hamiltonian is different from that appears in [6], in which the graph is weighted.

Notice that the Hamiltonian H is 2-sparse and the absolute value of every matrix element is at most 1.

We are going to show the following. Suppose $\mathcal{A}$ can simulate H for an evolution time $t \in [0, (L+1)/2]$ within an error $\epsilon < 1/4$ by making $o(t)$ p-parallel queries to $\mathcal{O}_H$ and $\mathcal{O}_L$. Then we can construct a reduction $\mathcal{R}$ that makes $o(t)$ p-parallel queries to Π_j and Π_j^{-1} and outputs a pair (x_0, x_q) such that $x_q = f^{(q)}(x_0)$ and $q > t$ with constant probability. The reduction $\mathcal{R}$ is described as follows:

1. Run the Hamiltonian simulation algorithm $\mathcal{A}$ on inputs the Hamiltonian H, the evolution time $t \in [0, L/2]$ and the initial state $|0, 0^n\rangle$.
   - When $\mathcal{A}$ queries $\mathcal{O}_H$ on the index $((j, x), (j', x'))$, the reduction $\mathcal{R}$ returns the response by the following rules.
     - If $j' = j + 1$ and $x' = \Pi_{j+1}(x)$, then $\mathcal{R}$ returns 1.
     - If $j' = j - 1$ and $x' = \Pi_j^{-1}(x)$, then $\mathcal{R}$ returns 1.
     - Otherwise, $\mathcal{R}$ returns 0.
   - When $\mathcal{A}$ queries $\mathcal{O}_L$ on $((j, x), s)$, reduction $\mathcal{R}$ returns the response by the following rules.
     - If $j = 0$ and $s = 1$, then $\mathcal{R}$ returns $(1, \Pi_1(x))$.
     - If $j = L$ and $s = 1$, then $\mathcal{R}$ returns $(L - 1, \Pi_L^{-1}(x))$
     - If $j \neq 0, L$ and $s = 1$, then $\mathcal{R}$ returns $(j - 1, \Pi_j^{-1}(x))$.
     - If $j \neq 0, L$ and $s = 2$, then $\mathcal{R}$ returns $(j + 1, \Pi_{j+1}(x))$.
2. Measure the system in the $\{|j, x\rangle\}$ basis and obtain the outcome (q, x_q).
3. Output x_q.

Note that answering a query to the entry oracle $\mathcal{O}_H$ can be implemented by $O(1)$ queries to Π_j and Π_j^{-1}. Similarly, the sparse structure oracle $\mathcal{O}_L$ can be simulated by $O(1)$ queries to Π_j and Π_j^{-1} as well.

Next, we analyze the evolution under H. Let us define another Hamiltonian $H|_0$ restricted to the subspace Span $\left(\left\{|j, f^{(j)}(0)\rangle\right\}_{j=0}^{L}\right)$:

$$H|_0 = \sum_{j=0}^{L-1} |j+1, f^{(j+1)}(0)\rangle\langle j, f^{(j)}(0)| + |j, f^{(j)}(0)\rangle\langle j+1, f^{(j+1)}(0)|.$$

By Observation 40, the time evolution under $H|_0$ is equivalent to the time evolution under H with the initial state $|0, 0^n\rangle$.

We first consider a perfect Hamiltonian simulation algorithm $\widetilde{\mathcal{A}}$ that outputs the state $|\widetilde{\psi}\rangle \coloneqq e^{-iHt}|0, 0^n\rangle = e^{-iH|_0 t}|0, 0^n\rangle$. In Step 2, the measurement outcome (q, x_q) satisfies $x_q = f^{(q)}(0^n)$. Then by Lemma 35, the probability that the measurement outcome satisfies $q > t$ is at least $1/3$.

Next, we consider the general simulation algorithm that outputs a state $|\psi\rangle$ such that $\Delta(|\psi\rangle\langle\psi|, |\widetilde{\psi}\rangle\langle\widetilde{\psi}|) \leq 1/4$. By the property of the trace distance, the difference in probabilities that $\mathcal{R}$ outputs a correct outcome by measuring $|\psi\rangle$ and $|\widetilde{\psi}\rangle$ is at most $1/4$. As a result, $\mathcal{R}$ outputs the accepted string x_q with probability at least $1/3 - 1/4 = 1/12$.

Combining everything together, if $\mathcal{A}$ simulates H for time t within $\epsilon \leq 1/4$ by making $o(t)$ p-parallel queries, then $\mathcal{R}$ will output $x_q = f^{(q)}(0^n)$ such that $q > t$ with constant probability by making $o(t)$ p-parallel queries. This contradicts Corollary 39. ◀

9 No Fast-forwarding in Plain Model

In this section, we are going to investigate the parallel lower bound of Hamiltonian simulation in the plain model. In the plain model, we are interested in the Hamiltonians that have a succinct description. Typically, we consider the *local Hamiltonians*.

▶ **Definition 41** (Local Hamiltonian). *We say a Hamiltonian H that acts on n qubits is* k-local *if H can be written as*

$$H = \sum_j H_j,$$

where each H_j acts non-trivially on at most k qubits.

The *geometrically local* Hamiltonians are another kind of Hamiltonians that often appear in physics models. A geometrically local Hamiltonian is a local Hamiltonian with more constraints. For a geometrically local Hamiltonian written by $H = \sum_j H_j$, each term H_j acts non-trivially on the qubits that are near in space. We are especially interested in one-dimensional geometrically local Hamiltonians.

▶ **Definition 42** (One-dimension geometrical local Hamiltonians). *Let a system consist of n qubits that are aligned in space and each qubit is labeled by an integer $l \in [n]$. Let $H = \sum_j H_j$ be a k-local Hamiltonian that acts on n qubits. We say H is an* one-dimension geometrically local *Hamiltonian if each H_j acts non-trivially on at most k consecutive indices.*

For example, consider Hamiltonians H_1 and H_2 acting on four qubits defined as follows:

$$H_1 := \sigma^X \otimes \sigma^Z \otimes I \otimes I + \sigma^Z \otimes I \otimes I \otimes \sigma^Z + I \otimes I \otimes \sigma^X \otimes I,$$

and

$$H_2 := \sigma^Z \otimes \sigma^Z \otimes I \otimes I + I \otimes \sigma^Z \otimes \sigma^Z \otimes I + I \otimes I \otimes \sigma^Z \otimes \sigma^Z,$$

where σ^X and σ^Z are Pauli operators and I is the identity operator. Hamiltonian H_1 is 2-local but not geometrically local, but Hamiltonian H_2 is geometrically local. We normalize the Hamiltonian by setting the spectral norm $\|H_j\| = O(1)$ for each j.

Having a succinct description gives the simulation algorithm more power than in the oracle model. In this sense, we obtain a stronger lower bound. On the other hand, our lower bound in the plain model relies on computational assumptions, which weakens the result. For our lower bound, we need to assume an iterative parallel-hard function, which is slightly modified from the definition of an iterative sequential function by Boneh et al. [9].

▶ **Definition 43** (Iterative parallel-hard functions/puzzles). *A function $f\colon \mathbb{N} \times \hat{X} \to X$ where $\hat{X} \in X$ and $|\hat{X}| = 2^{\theta(\lambda)}$ is a (post-quantum) (s,d)-iterated parallel-hard function if there exists a function $g\colon X \to X$ such that*

- *g can be computed by a quantum circuit with width λ and size $s(\lambda)$. Without loss of generality, we can let $s(\lambda) = \Omega(\lambda)$*
- *$f(k,x) = g^{(k)}(x)$.*
- *For all sufficiently large $k = 2^{o(\lambda)}$, for any quantum circuit C with depth less than $d(k)$ and size less than $\mathrm{poly}(t, d(k), \lambda)$,*

$$\Pr[C(x) = f(k,x) \mid x \leftarrow \hat{X}] \leq \mathrm{negl}(\lambda)$$

Without loss of generality, we assume that d is non-decreasing.

We say that f forms a (post-quantum) (s, d)-iterated parallel-hard puzzle if it only satisfies a weaker version of the third requirement as follows:

- *For all $k = 2^{o(\lambda)}$, for any uniform quantum circuit C with depth less than $d(k)$ and size less than* $\mathrm{poly}(t, d(k), \lambda)$,

$$\Pr[C(x) = f(k', x) \text{ for some } k' \geq k/2 \mid x \leftarrow \hat{X}] \leq \mathrm{negl}(\lambda).$$

Note that an (s, d)-iterated parallel-hard function is directly an (s, d')-iterated parallel-hard puzzle, where $d'(x) := d(x/2)$.

Under the (quantum) random oracle heuristic [4, 10], such parallel-hard puzzles can be heuristically obtained by instantiating the twisted hash chain with a cryptographic hash function.

▶ **Assumption 44.** *With the random oracle heuristic, we can assume that the standard instantiation of the twisted hash chain is parallel-hard by Corollary 33. Assuming the cryptographic hash function h in the instantiation can be implemented by circuits of size $s(\lambda)$ on λ-bit inputs, the twisted hash chain directly gives an $(s + O(\lambda), d)$ iterative parallel-hard function with $d(x) := x - 1$, which is an $(s + O(\lambda), d)$ iterative parallel-hard puzzle with $d(x) := \lfloor \frac{x}{2} \rfloor - 1$*

We present the simulation lower bound for the local Hamiltonians in the following theorem.

▶ **Theorem 45** (Simulation lower bound for local Hamiltonians in the plain model)**.** *Assuming an (s, d)-iterated parallel-hard puzzle, for any integer n, there exists a time-independent c-local Hamiltonian H acting on $n + (2s(n)T(n))^{1/c}$ qubits such simulating H for an evolution time $t \in [0, s(n)T(n)]$ with error $\epsilon < 1/4$ needs a $(d(\lfloor t/2s(n) \rfloor) - O(s(n)))$-depth circuit, where $T(\cdot)$ is an arbitrary polynomial and c is a constant.*

We also have the lower bound for simulating geometrically local time-dependent Hamiltonians.

▶ **Theorem 46** (Simulation lower bound for geometrically local Hamiltonians in the plain model)**.** *Assuming an (s, d)-iterated parallel-hard function, for any integer n, there exists a piecewise-time-independent 1-D geometrically 2-local Hamiltonian H acting on n qubits such that simulating H for an evolution time $t \in [0, ns(n)T(n)]$ with error $\epsilon(n) \leq 1 - 1/\mathrm{poly}(n)$ needs a $\left(d(\lfloor \frac{t}{ns(n)} \rfloor) - O(ns(n)) - \mathrm{poly}(\log(n), \log\log(1/\epsilon'(n))\right)$-depth circuit, where $T(\cdot)$ is an arbitrary polynomial and $\epsilon'(n) < 1 - \epsilon(n) - 1/\mathrm{poly}(n)$.*

We sketch our proof strategy as follows. The main idea is, again, to reduce the hard problem to the Hamiltonian simulation problem. First, we consider a quantum circuit C that computes an (s, d)-iterated parallel-hard puzzle, which according to the definition, can be written as a sequential composition of λ-qubit $s(\lambda)$-sized circuits. Then we construct a Hamiltonian H_{circuit} to implement the circuit C by the circuit to Hamiltonian reduction technique. The circuit to time-independent reduction is introduced in Section 9.1, and the circuit to time-dependent reduction is introduced in Section 9.3. Finally, we use the Hamiltonian simulation algorithm to simulate the Hamiltonian H_{circuit}. If we can fast-forward the Hamiltonian evolution under H_{circuit}, then we can break the depth guarantee provided by the iterated parallel-hard puzzle.

9.1 Circuit to time-independent Hamiltonian

Feynman suggested that we can implement a quantum circuit (which was called reversible computation at his time) by a time-independent Hamiltonian [21]. About a decade later, Childs and Nagaj provided rigorous analyses for the implementation [15, 30]. The idea is to introduce an extra register, which is called the clock register $\mathcal{H}_{\text{clock}}$, associated with the circuit register $\mathcal{H}_{\text{circuit}}$ to record the progress of the quantum circuit. After introducing the clock register, we define a state $|\psi_j\rangle = |\phi_j\rangle_{\text{circuit}} \otimes |\gamma_j\rangle_{\text{clock}}$ to indicate that the j-th steps outcomes is $|\phi_j\rangle$. We can construct a Hamiltonian acting on $\mathcal{H}_{\text{circuit}} \otimes \mathcal{H}_{\text{clock}}$ such that during the evolution, the system is in Span $(\{|\psi_j\rangle\})$. When we measure the clock register $\mathcal{H}_{\text{clock}}$ and get the outcome γ_k, the quantum state in the circuit register $\mathcal{H}_{\text{circuit}}$ collapses to $|\phi_k\rangle$. And then we obtain the k-th step outcome of the quantum circuit. Similar techniques appear in the proof of QMA completeness [25] and universality of adiabatic computation [1].

In [30], Nagaj proved that for any quantum circuit C with L quantum gates, there is a Hamiltonian H_{circuit} such that evolving the system under the Hamiltonian H_{circuit} for time $O(L)$ and then measuring the system, we can get the final state of the quantum circuit C with high probability. We extend Nagaj's result. In this section, we are going to prove that if the system evolves under H_{circuit} for time $t \in [0, L/2]$, we can get $|\psi_j\rangle$ where $j \geq t$ with high probability. Our method is slightly different from Nagai's. In [30], the evolution time is uniformly sampled, while we have an explicit evolution time. Another difference is that in [30] it needs to pad $O(L)$ dummy identity gates at the end of the quantum circuit to amplify the probability of getting the output state. In our construction, padding is not required.

Let a quantum circuit C which acts on the register $\mathcal{H}_{\text{circuit}}$ consist of a sequence of g quantum gates $U_1, U_2, \ldots, U_L$. Namely,

$$C = U_L U_{L-1} \cdots U_1.$$

After introducing the clock register $\mathcal{H}_{\text{clock}}$, and the clock state, which is a family of orthonormal states $\{|\gamma_j\rangle\}_{j=0}^{L}$ where each $|\gamma_j\rangle \in \mathcal{H}_{\text{clock}}$, we construct the following Hamiltonian

$$H_{\text{circuit}} := \sum_{j=1}^{L} H_j, \tag{43}$$

where

$$H_j := U_j \otimes |j\rangle\langle j-1| + U_j^\dagger \otimes |j-1\rangle\langle j|. \tag{44}$$

Let the input state of the circuit be $|\phi_0^{(0)}\rangle$, and let $|\phi_j^{(0)}\rangle$ denote the quantum state of j-th step. That is, $|\phi_j^{(0)}\rangle = U_j U_{j-1} \cdots U_1 |\phi_0^{(0)}\rangle$. Define the state

$$|\psi_j^{(0)}\rangle := |\phi_j^{(0)}\rangle \otimes |\gamma_j\rangle, \tag{45}$$

which is the state after j steps of C.

Let $\{|\phi_0^{(1)}\rangle, \ldots, |\phi_0^{(N-1)}\rangle\}$ be the states that are orthogonal to $|\phi_0^{(0)}\rangle$ where N denotes the dimension of $\mathcal{H}_{circuit}$. Besides, let $|\phi_j^{(m)}\rangle := U_j U_{j-1} \cdots U_1 |\phi_0^{(m)}\rangle$ and $|\psi_j^{(m)}\rangle := |\phi_j^{(m)}\rangle \otimes |\gamma_j\rangle$ where $m \in [N-1]$. We have

$$H_{\text{circuit}}|\psi_j^{(m)}\rangle = \begin{cases} |\psi_{j+1}^{(m)}\rangle & , \text{ if } j = 0, \\ |\psi_{j-1}^{(m)}\rangle + |\psi_{j+1}^{m}\rangle & , \text{ if } j = 1, \ldots, L-1, \\ |\psi_{j-1}^{(m)}\rangle & , \text{ if } j = L, \end{cases}$$

for any $m \in \{0, 1, \ldots, N-1\}$.

Again, the evolution under H_{circuit} can be viewed as quantum walks on a graph. We construct a graph illustrated in Figure 7 whose adjacency matrix is H_{circuit}.

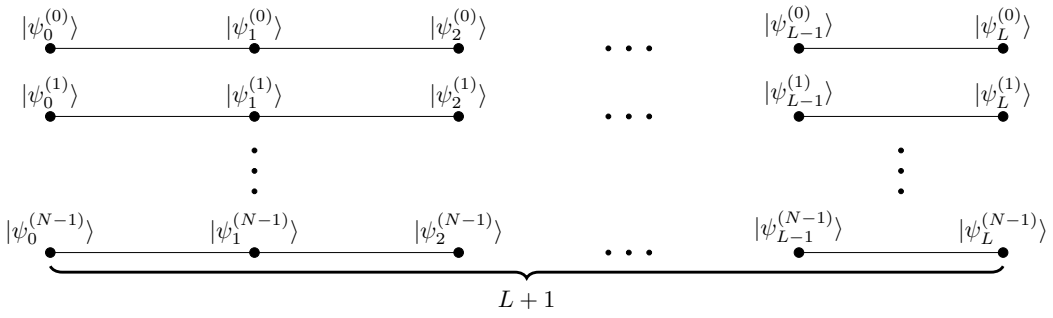


Figure 7 The quantum walk on a graph for the circuit to Hamiltonian reduction.

We have the following lemma.

▶ **Lemma 47.** *A system evolves under the Hamiltonian H_{circuit} described in (43) with the initial state $|\psi_0^{(0)}\rangle$ described in (45). If the clock register is measured at time $t \in [0, L/2]$ in the $\{\gamma_j\}$ basis and get the outcome l, the probability that the circuit register collapses to $|\phi_l^{(m)}\rangle$ where $m = 0$ and $l > t$ is at least $1/3$.*

Proof. By the similar argument of Observation 40, we define another Hamiltonian $H|_{\psi^{(0)}}$ restricted to the subspace Span $\left(\{|\psi_j^{(0)}\rangle\}_{j=0}^{L}\right)$:

$$H|_{\psi^{(0)}} := \sum_{j=0}^{L-1} |\psi_{j+1}^{(0)}\rangle\langle\psi_j^{(0)}| + |\psi_j^{(0)}\rangle\langle\psi_{j+1}^{(0)}|. \tag{46}$$

If the system is initially at $|\psi_0^{(0)}\rangle$, the time evolution under H_{circuit} is the same as the time evolution under $H|_{\psi^{(0)}}$. As a result, we have $m = 0$ for any time t.

Because $|\psi_j^{(0)}\rangle = |\phi_j^{(0)}\rangle \otimes |\gamma_j\rangle$, the measurement results of measuring clock register in $\{|\gamma_j\rangle\}$ basis is the same as measuring the entire system in $\{|\psi_j^{(0)}\rangle\}$ basis. The probability that $l > t$ can be obtained directly from Lemma 35. This finishes the proof. ◀

Next, we present our construction of the clock state.

▶ **Lemma 48.** *For all $c, T \in \mathbb{N}$, there exists a construction that implements the clock state for time T with locality c and at most $O(T^{1/(c-1)})$ qubits.*

Proof. Let n be the smallest integer such that $\binom{n}{c-1} \geq T$. Thus, $n = O(T^{1/(c-1)})$. Consider the system that consists of n qubits indexed from 1 to n. Consider the Johnson graph $J_{n,c-1}$ (see Definition 16), for each node $S \subseteq [n]$, define the n-qubit state $|S\rangle$ as

$$|S\rangle := \bigotimes_{i=1}^{n} |I_S(i)\rangle_i,$$

where the subscript i denotes the register of the i-th qubit; $I_S : [n] \to \{0,1\}$ is the indicator function that equals 1 if $i \in S$ and 0 otherwise.

Choose an arbitrary Hamiltonian path of $J_{n,c-1}$, denoted by $(S_1, S_2, \ldots, S_{\binom{n}{c-1}})$. Now, each time $j \in [\binom{n}{c-1}]$ corresponds to the n-qubit state $|S_j\rangle$.

For every $j \in \left[\binom{n}{c-1} - 1\right]$, the time transition $|j+1\rangle\langle j|$ is implemented by

$$E_{j\to j+1} := \bigotimes_{i=1}^{n} P_i,$$

where

$$P_i := \begin{cases} |1\rangle\langle 1|_i, & \text{if } i \in S_j \cap S_{j+1} \\ |1\rangle\langle 0|_i, & \text{if } i \in S_{j+1} \setminus S_j \\ |0\rangle\langle 1|_i, & \text{if } i \in S_j \setminus S_{j+1} \\ I, & \text{otherwise.} \end{cases}$$

It is easy to see that the transitions are c-local due to the definition of $J_{n,c-1}$. It remains to check the correctness of the transitions. That is, for every $j, j' \in \left[\binom{n}{c-1} - 1\right]$, they should satisfy

$$E_{j\to j+1}|S_{j'}\rangle = \begin{cases} |S_{j'+1}\rangle, & \text{if } j = j' \\ 0, & \text{otherwise.} \end{cases}$$

When $j = j'$, the equality holds since there is an edge between S_j and S_{j+1}. When $j \neq j'$, there must exist an $i^* \in S_j$ such that $i^* \notin S_{j'}$. Therefore, $P_{i^*} = |0\rangle\langle 1|_{i^*}$ will vanish $|S_{j'}\rangle$ because the i^*-th qubit of $|S_{j'}\rangle$ is in the state $|0\rangle_i$. This verifies the correctness. ◀

9.2 Proof of the lower bound for local Hamiltonians

Proof of Theorem 45. From an (s,d)-iterated parallel-hard puzzle $f(k,x) = g^{(k)}(x)$, we here show how to construct a time-independent $(c+2)$-local Hamiltonian H acting on $n + (2s(n)T(n))^{1/c}$ qubits.

The construction is direct. By definition, g can be implemented by an $s(n)$-sized quantum circuit C^g. We can thus construct a circuit C concatenating $T(n)$ copies of C^g, which computes $f(T(n), x)$ with size $s(n)T(n)$. Note that, if we denote $C(i,x)$ to be the intermediate output of $C(x)$ after applying the i-th gate, then we additionally have $C(ks(n), x) = |f(k,x)\rangle$ for all $k \leq T(n)$.

Given such a circuit C, we can construct a Hamiltonian H by the circuit-to-Hamiltonian reduction introduced in Section 9.1. We use the construction in Lemma 48 with locality c and time-bound $2s(n)T(n)$ to implement the clock state. Hence $H = \sum_j H_j$, where $H_j := U_j \otimes |j\rangle\langle j-1| + U_j^\dagger \otimes |j-1\rangle\langle j|$, and U_j is a unitary corresponding to the j-th gate of C. Note that U_j is always an one- or two-qubit gate, and $|j\rangle\langle j-1|$ is c-local. Hence H is $c+2$ local over $n + (2s(n)T(n))^{1/c}$ qubits. It is also direct to see that $\|H_j\| = O(1)$ for each j.

For such H and any $t \in [0, s(n)T(n)]$, if there is a quantum algorithm $\mathcal{A}$ that computes e^{-iHt} within depth $d_{\mathcal{A}}$, we can indeed construct a quantum algorithm $\mathcal{R}$ with depth $d_{\mathcal{A}}+O(n)$ that computes the underlying parallel-hard puzzle. The algorithm $\mathcal{R}$ is defined as follows.

1. For an input $x \in \{0,1\}^n$, run the algorithm $\mathcal{A}$ with input Hamiltonian H and input state $|x\rangle$, obtain the output state of $\mathcal{A}$, denoted by $|\psi\rangle$.
2. Measure the clock register $\mathcal{H}_{\text{clock}}$ in $\{|\gamma_j\rangle\}$ basis and obtain some $l \in [s(n)f(n)]$. The residual state in the circuit register $\mathcal{H}_{\text{circuit}}$ is denoted by $|\phi_l\rangle$.
3. Let $m = \lceil l/s(n) \rceil$. Apply $U_{ms(n)} \cdots U_{l+1}$ on $|\phi_l\rangle$. Let the final state be $|\phi_m\rangle$.
4. Measure $|\phi_m\rangle$ on the computational basis and obtain the outcome x_m.
5. Output x_m

In this construction, Step 2 and Step 4 can be done within depth $O(n)$ and Step 3 can be done within depth $s(n)$. It is easy to see that $\mathcal{R}$ can be implemented with depth $d_{\mathcal{A}}+O(s(n))$.

We claim that, with constant probability, $m > \frac{t}{s(n)}$ and $x_m = f(m, x)$. This implies that $\mathcal{R}$ can break the underlying parallel-hard puzzle on $k = 2\lfloor \frac{t}{s(n)} \rfloor$

To prove the claim, we first consider a simplified case, where there exists an ideal $\widetilde{\mathcal{A}}$ that perfectly simulates H for time t and plugs it into our construction. We use $|\widetilde{\psi}\rangle$ to denote the output of $\widetilde{\mathcal{A}}$ in Step 1, and similarly $|\widetilde{\psi}_l\rangle$ for output in Step 2. By Lemma 47, we have $|\widetilde{\phi}_l\rangle = U_l \cdots U_1 |x\rangle$ for all l and the probability that we get some $l > t$ in Step 2 is at least $1/3$. By the definition of C, the output state of Step 3 satisfies $|\widetilde{\phi}_m\rangle = U_{ms(n)} \cdots U_1 |x\rangle = |f(m, x)\rangle$. Moreover, $m \geq \lfloor \frac{t}{s(n)} \rfloor$ with probability at least $1/3$. This matches the required condition of our claim.

Now we return to the general $\mathcal{A}$ with simulation error $\epsilon < 1/4$. Let $|\psi\rangle$ be the output of $\mathcal{A}$ with error ϵ. Observe that we have $\Delta(|\psi\rangle\langle\psi|, |\widetilde{\psi}\rangle\langle\widetilde{\psi}|) < 1/4$. Thus, by the definition of the trace distance, the difference in probabilities of obtaining any outcome by applying the same procedure to two states should be at most $1/4$. Hence, with probability at least $1/3 - 1/4 = 1/12$, measuring $|\phi_m\rangle$ gives $f(m, x)$ with some $m \geq \lfloor \frac{t}{s(n)} \rfloor$. This completes the proof of our claim.

Finally, by the security guarantee of the (s, d)-iterated parallel-hard puzzle, any circuit computing the puzzle for $k = 2\lfloor \frac{t}{s(n)} \rfloor$ should have depth at least $d(2\lfloor \frac{t}{s(n)} \rfloor)$. This gives an lower bound that $d_{\mathcal{A}} + O(s(n)) > d(2\lfloor \frac{t}{s(n)} \rfloor)$, which completes the proof. ◀

9.3 Circuit to time-dependent Hamiltonian

In this section, we will show how to encode a circuit into a time-dependent geometrically local circuit.

▶ **Lemma 49.** *A quantum circuit C (of 2-qubit gates) over n qubits of size s can be transformed into a circuit C' over n qubits of size ns, such that every gate in C' acts only on consecutive qubits, and $C'(x) = \pi C(x)$ for some permutation π on n elements. We call such C' a geometrically local circuit.*

Proof. The proof of this small lemma is very direct. Given a circuit C with (sequential) gates $G_1, G_2, \ldots, G_s$, where gate G_i acts on qubits α_i, β_i, then we can rewrite C as $S_{1,\alpha_1}, S_{1,\alpha_1+1}, \ldots, S_{1,\beta_1-2}, G'_1, G'_2, \ldots, G'_s$, where S_i is a swap gate acting on the i and $(i+1)$-th qubits, and G'_i is G_i acting on the permuted qubits. Note that now G'_1 is acting on two consecutive qubits $\beta_1 - 1, \beta_1$. It is easy to see that applying G_1 and applying $S_{1,\alpha_1}, S_{1,\alpha_1+1}, \ldots, S_{1,\beta_1-2}, G'_1$ generate output states that differ up to a permutation. Through repeating such process s times, we can obtain a circuit C' that consists of at most $(n-1)s$ swap gates and s permuted gates from C. Furthermore, every gate in C' is acting only on consecutive qubits. ◀

▶ **Theorem 50.** *Given a quantum circuit C over n qubits that consists of s gates $U_1 \dots U_s$, we can define a time-dependent Hamiltonian H such that $H(t) := -i \log U_i$ for all $t \in [i-1, i)$ and $U_i := I_n$ for $i > s$. Note that H obviously satisfies $e^{iHt}|\phi\rangle = C|\phi\rangle$ for $t > s$.*

▶ Remark 51. A Hamiltonian H obtained from a geometrically local circuit is 1-D geometrically 2-local.

▶ Remark 52. Such time-independent Hamiltonians that remain constant in each time segment are also called *piecewise constant Hamiltonians* in [23].

9.4 Proof of the lower bound for geometrically local Hamiltonians

Before proving the theorem, we will need the existence of a Hamiltonian simulation algorithm for 2-local *time-independent* Hamiltonians. While there are plenty of proposals in the literature, we use the one in [39] which provides a good dependency on ϵ. The choice of the simulation algorithm will only affect an additive term of our bound.

▶ **Theorem 53** ([39], with some parameter specified)**.** *A 2-local Hamiltonian acting on n qubits can be simulated for time t within precision ϵ by a quantum circuit of depth* $\mathrm{poly}(\log\log(1/\epsilon), \log(n), t)$.

Proof of Theorem 46. The proof is very similar to the proof of Theorem 45. From an (s, d)-iterated parallel-hard function $f(k, x) = g^{(k)}(x)$, we will construct a geometrically 2-local Hamiltonian H over n qubits.

Again, since g can be implemented by an $s(n)$ sized circuit C^g, we can construct C^f by concatenating $T(n)$ copies of C^g, which computes $f(T(n), x)$ with size $s(n)T(n)$. Then, by applying Lemma 49, we can obtain a geometrically local circuit C of depth at most $ns(n)T(n)$. Note that Lemma 49 basically constructs C by adding at most $n-1$ swap gates before each gate of C^f. For notation simplicity, we add extra dummy gate before each gate so that there are exactly $(n-1)$ gates added before each gate of C^f. Thus the depth of C is exactly $ns(n)T(n)$, and if we denote $C(i, x)$ to be the intermediate output of $C(x)$ after applying the i-th gate, we have $C(kns(n), x) = \pi_k|f(k, x)\rangle$ for all $k \le T(n)$, where π_k is some (known) permutation on n bits.

With such a geometrically local circuit C, we can apply Theorem 50 to obtain a time-dependent geometrically 2-local Hamiltonian H over n qubits. For all $t \in [0, ns(n)T(n)]$, if there is a quantum algorithm $\mathcal{A}$ that computes e^{-iHt} with precision ϵ of depth $d_{\mathcal{A}}$, we can indeed construct a quantum algorithm $\mathcal{R}$ of depth $d_{\mathcal{A}} + O(s(n)) + \mathrm{poly}(\log\log(1/\epsilon'), \log n)$ with $\epsilon' < 1 - \epsilon - 1/\mathrm{poly}(n)$. The algorithm $\mathcal{R}$ is defined as follows.

1. For an input $x \in \{0,1\}^n$, run $\mathcal{A}$ on inputs the Hamiltonian H and the initial state $|x\rangle$ to obtain the output state $|\phi\rangle$.
2. Run the Hamiltonian simulation algorithm $\mathcal{S}$ in Theorem 53 with inputs the Hamiltonian $H(t)$ (which is time-independent) , the initial state $|\phi\rangle$, the evolution time $\lceil t \rceil - t$, and the precision parameter ϵ' to obtain the output state $|\phi_{\lceil t \rceil}\rangle$.
3. Let $m = \lceil \frac{t}{ns(n)} \rceil$. Apply $U_{mns(n)} \cdot U_{\lceil t \rceil + 1}$ to $|\phi_{\lceil t \rceil}\rangle$. Let the final state be $|\phi_m\rangle$.
4. Measure the permuted state $\pi_m|\phi_m\rangle$ on the computational basis and obtain the outcome x_m.
5. Output x_m.

In the construction, Step 3 can be done within depth $O(ns(n))$ and Step 4 can be done within depth $O(1)$. Thus, R can be instantiated within depth $d_{\mathcal{A}} + O(ns(n)) + \mathrm{poly}(\log\log(1/\epsilon'), \log n)$.

Now we show that if $\epsilon + \epsilon' \leq 1 - 1/\operatorname{poly}(n)$, then $x_m = f(k, m)$ holds with non-negligible probability.

We consider the case in which the algorithms $\mathcal{A}$ and $\mathcal{S}$ both simulate the evolution perfectly. Denote the output state in each step of this experiment as $|\tilde{\phi}\rangle, |\tilde{\phi}_{\lceil t \rceil}\rangle$ and $|\tilde{\phi}_m\rangle$. In particular,

$$|\tilde{\phi}\rangle = \exp_{\mathcal{T}}\left(-i\int_0^t H(t')dt'\right)|x\rangle$$

and

$$\begin{aligned}|\tilde{\phi}_{\lceil t \rceil}\rangle &= e^{-iH(t)(\lceil t \rceil - t)} \times \exp_{\mathcal{T}}\left(-i\int_0^t H(t')dt'\right)|x\rangle \\ &= \exp_{\mathcal{T}}\left(-i\int_0^{\lceil t \rceil} H(t')dt'\right)|x\rangle = |C(\lceil t \rceil, x)\rangle,\end{aligned}$$

where the last equation follows from the definition of H, while the second last equation follows from the fact that H is constant on the interval $(t, \lceil t \rceil)$. Hence, $|\tilde{\phi}_m\rangle = |C(mns(n), x)\rangle = |f(m, x)\rangle$ with probability 1.

Back to the actual construction, by the precision guarantee of $\mathcal{A}$ and $\mathcal{S}$, there exists some polynomial p such that $\Delta(|\psi_{\lceil t \rceil}\rangle\langle\psi_{\lceil t \rceil}|, |\tilde{\psi}_{\lceil t \rceil}\rangle\langle\tilde{\psi}_{\lceil t \rceil}|) < 1 - 1/p(n)$. Thus, $x_m = f(m, x)$ holds with probability at least $1/p(n)$.

Finally, by the security guarantee of the (s, d)-iterated parallel-hard function, any quantum circuit that computes $f(m, x)$ should have depth at least $d(m) = d(\lceil \frac{t}{ns(n)} \rceil)$. This gives an lower bound that $d_{\mathcal{A}} + O(ns(n)) + \operatorname{poly}(\log\log(1/\epsilon), \log(n)) < d(\lceil \frac{t}{ns(n)} \rceil)$, which completes the proof. ◀

10 No Fast-forwarding with Natural Simulators

In previous sections, we have shown no-go theorems for using quantum circuits to parallelly fast-forward (geometrically) local Hamiltonian simulation. Here, we are going to generalize Theorem 46 and Theorem 45 by showing that simulators that are geometrically local Hamiltonians cannot do much better than quantum circuits.

▶ **Corollary 54.** *Assuming an (s, d)-iterated parallel-hard function, for any integer n, there exists a piecewise-time-independent 1-D geometrically 2-local Hamiltonian H_A acting on n qubits satisfying the following: For any geometrically constant-local Hamiltonian H_B acting on $\operatorname{poly}(n)$ qubits, using H_B to simulate H_A for evolution time $t \in [0, ns(n)T(n)]$ with error $\epsilon(n) \leq 1 - 1/\operatorname{poly}(n)$ needs $(d(\lfloor \frac{t}{ns(n)} \rfloor) - O(ns(n)) - \operatorname{poly}(\log(n), \log\log(1/\epsilon'(n))))/\operatorname{polylog}(tn/\epsilon)$ evolution time, where $T(\cdot)$ is an arbitrary polynomial and $\epsilon'(n) < 1 - \epsilon(n) - 1/\operatorname{poly}(n)$.*

Proof. Let H_A be the Hamiltonian we considered in the proof of Theorem 46. Suppose there exists H_B that can simulate H_A for evolution time $t \in [0, ns(n)T(n)]$ and error $\epsilon(n)/2 \leq 1 - 1/\operatorname{poly}(n)$ with $t' < \frac{d(\lfloor \frac{t}{ns(n)} \rfloor) - O(ns(n)) - \operatorname{poly}(\log(n), \log\log(1/\epsilon'(n)))}{\operatorname{polylog}(tn/\epsilon)}$ evolution time, where $\epsilon'(n) < 1 - \epsilon(n)/2 - 1/\operatorname{poly}(n)$.

Recall that the algorithm in [23] can simulate a geometrically constant-local Hamiltonian with quantum circuit depth $t \cdot \operatorname{polylog}(tn/\epsilon)$, where n is the number of qubits and ϵ is the precision parameter.

Then, we apply the algorithm in [23] to simulate H_B with evolution time t' and precision $\epsilon/2$. This leads to a simulation algorithm for H_A with error ϵ and circuit depth strictly less than $d(\lfloor \frac{t}{ns(n)} \rfloor) - O(ns(n)) - \operatorname{poly}(\log(n), \log\log(1/\epsilon'(n)))$, which contradicts Theorem 46. ◀

▶ **Corollary 55.** *Assuming an (s, d)-iterated parallel-hard function, for any integer n, there exists a time-independent c-local Hamiltonian H acting on $n+(2s(n)T(n))^{1/c}$ qubits satisfying the following: For any geometrically constant-local Hamiltonian H_B acting on $\mathrm{poly}(n)$ qubits, using H_B to simulate H_A for evolution time $t \in [0, s(n)T(n)]$ with error $\epsilon(n) < 1/4$ needs $d(\lfloor t/2s(n) \rfloor) - O(s(n))/\mathrm{polylog}(tn/\epsilon)$ evolution time, where $T(\cdot)$ is an arbitrary polynomial and c is a constant.*

The proof for Corollary 55 is similar to the proof for Corollary 54.

References

1 Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic Quantum Computation Is Equivalent to Standard Quantum Computation. *SIAM Review*, 50(4):755–787, 2008.

2 Brian Alspach. Johnson graphs are Hamilton-connected. *Ars Mathematica Contemporanea*, 6(1):21–23, 2012.

3 Yosi Atia and Dorit Aharonov. Fast-forwarding of hamiltonians and exponentially precise measurements. *Nature Communications*, 8(1), November 2017. doi:10.1038/s41467-017-01637-7.

4 Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security*, pages 62–73, 1993.

5 Charles H Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM journal on Computing*, 26(5):1510–1523, 1997.

6 Dominic W Berry, Graeme Ahokas, Richard Cleve, and Barry C Sanders. Efficient quantum algorithms for simulating sparse Hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371, 2007.

7 Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. Exponential improvement in precision for simulating sparse Hamiltonians. *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 283–292, May 2014. arXiv:1312.1414. doi:10.1145/2591796.2591854.

8 Dominic W. Berry, Andrew M. Childs, and Robin Kothari. Hamiltonian Simulation with Nearly Optimal Dependence on all Parameters. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 792–809, October 2015. ISSN: 0272-5428. doi:10.1109/FOCS.2015.54.

9 Dan Boneh, Joseph Bonneau, Benedikt Bünz, and Ben Fisch. Verifiable delay functions. In *CRYPTO (1)*, volume 10991 of *Lecture Notes in Computer Science*, pages 757–788. Springer, 2018.

10 Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings 17*, pages 41–69. Springer, 2011.

11 P Oscar Boykin, Tal Mor, Matthew Pulver, Vwani Roychowdhury, and Farrokh Vatan. On universal and fault-tolerant quantum computing: a novel basis and a new constructive proof of universality for Shor's basis. In *40th Annual Symposium on Foundations of Computer Science (Cat. No. 99CB37039)*, pages 486–494. IEEE, 1999.

12 Jorge Chávez-Saab, Francisco Rodríguez-Henríquez, and Mehdi Tibouchi. Verifiable isogeny walks: Towards an isogeny-based postquantum VDF. In *SAC*, volume 13203 of *Lecture Notes in Computer Science*, pages 441–460. Springer, 2021.

13 Andrew M. Childs and Dominic W. Berry. Black-box Hamiltonian simulation and unitary implementation. *Quantum Information and Computation*, 12(1&2):29–62, January 2012. doi:10.26421/QIC12.1-2-4.

14 Andrew M. Childs, Richard Cleve, Enrico Deotto, Edward Farhi, Sam Gutmann, and Daniel A. Spielman. Exponential algorithmic speedup by a quantum walk. In *Proceedings of the thirty-fifth ACM symposium on Theory of computing - STOC '03*. ACM Press, 2003. doi:10.1145/780542.780552.

15 Andrew MacGregor Childs. *Quantum Information Processing in Continuous Time*. Thesis, Massachusetts Institute of Technology, 2004.

16 Kai-Min Chung, Serge Fehr, Yu-Hsuan Huang, and Tai-Ning Liao. On the compressed-oracle technique, and post-quantum security of proofs of sequential work. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 598–629. Springer, 2021.

17 J. Ignacio Cirac and Peter Zoller. Goals and opportunities in quantum simulation. *Nature Physics*, 8(4):264–266, April 2012. doi:10.1038/nphys2275.

18 Toby Cubitt, Ashley Montanaro, and Stephen Piddock. Universal Quantum Hamiltonians. *Proceedings of the National Academy of Sciences*, 115(38):9497–9502, September 2018. doi:10.1073/pnas.1804949115.

19 Naomi Ephraim, Cody Freitag, Ilan Komargodski, and Rafael Pass. Continuous verifiable delay functions. In *EUROCRYPT (3)*, volume 12107 of *Lecture Notes in Computer Science*, pages 125–154. Springer, 2020.

20 Luca De Feo, Simon Masson, Christophe Petit, and Antonio Sanso. Verifiable delay functions from supersingular isogenies and pairings. In *ASIACRYPT (1)*, volume 11921 of *Lecture Notes in Computer Science*, pages 248–277. Springer, 2019.

21 Richard P. Feynman. Quantum Mechanical Computers. *Optics News*, 11(2):11–20, February 1985. doi:10.1364/ON.11.2.000011.

22 Shouzhen Gu, Rolando D. Somma, and Burak Ş ahinoğlu. Fast-forwarding quantum evolution. *Quantum*, 5:577, November 2021. doi:10.22331/q-2021-11-15-577.

23 Jeongwan Haah, Matthew B. Hastings, Robin Kothari, and Guang Hao Low. Quantum algorithm for simulating real time evolution of lattice Hamiltonians. *SIAM Journal on Computing*, pages FOCS18–250–FOCS18–284, January 2021. doi:10.1137/18m1231511.

24 Stacey Jeffery, Frederic Magniez, and Ronald de Wolf. Optimal parallel quantum query algorithms, 2013. doi:10.48550/ARXIV.1309.6116.

25 A. Kitaev, A. Shen, and M. Vyalyi. *Classical and Quantum Computation*, volume 47 of *Graduate Studies in Mathematics*. American Mathematical Society, Providence, Rhode Island, May 2002. doi:10.1090/gsm/047.

26 Ilia Krasikov. Uniform bounds for Bessel functions. *Journal of Applied Analysis*, 12(1):83–91, 2006.

27 Guang Hao Low and Isaac L. Chuang. Optimal Hamiltonian simulation by quantum signal processing. *Phys. Rev. Lett.*, 118:010501, January 2017. doi:10.1103/PhysRevLett.118.010501.

28 Guang Hao Low and Isaac L. Chuang. Hamiltonian Simulation by Qubitization. *Quantum*, 3:163, July 2019. doi:10.22331/q-2019-07-12-163.

29 Michael Luby and Charles Rackoff. How to construct pseudorandom permutations from pseudorandom functions. *SIAM Journal on Computing*, 17(2):373–386, 1988.

30 Daniel Nagaj. Fast universal quantum computation with railroad-switch local Hamiltonians. *Journal of Mathematical Physics*, 51(6):062201, 2010.

31 Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. doi:10.1017/CBO9780511976667.

32 Krzysztof Pietrzak. Simple verifiable delay functions. In *ITCS*, volume 124 of *LIPIcs*, pages 60:1–60:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

33 Ronald L Rivest, Adi Shamir, and David A Wagner. Time-lock puzzles and timed-release crypto. *Massachusetts Institute of Technology. Laboratory for Computer Science*, 1996.

34 J. J. Sakurai and Jim Napolitano. *Modern Quantum Mechanics*. Cambridge University Press, third edition, September 2020. doi:10.1017/9781108587280.

35 Benjamin Wesolowski. Efficient verifiable delay functions. *J. Cryptol.*, 33(4):2113–2147, 2020.

36 Christof Zalka. Grover's quantum searching algorithm is optimal. *Physical Review A*, 60(4):2746, 1999.

37 Mark Zhandry. A note on the quantum collision and set equality problems, 2013. doi:10.48550/arXiv.1312.1027.

38 Mark Zhandry. How to record quantum queries, and applications to quantum indifferentiability. In *Annual International Cryptology Conference*, pages 239–268. Springer, 2019.

39 Zhicheng Zhang, Qisheng Wang, and Mingsheng Ying. Parallel quantum algorithm for Hamiltonian simulation. *arXiv preprint*, 2021. arXiv:2105.11889.

The Optimal Depth of Variational Quantum Algorithms Is QCMA-Hard to Approximate

Lennart Bittel
Institute for Theoretical Physics, Heinrich Heine University Düsseldorf, Germany

Sevag Gharibian
Department of Computer Science, and Institute for Photonic Quantum Systems, Universität Paderborn, Germany

Martin Kliesch
Institute for Theoretical Physics, Heinrich Heine University Düsseldorf, Germany
Institute for Quantum-Inspired and Quantum Optimization, Technische Universtiät Hamburg, Germany

Abstract

Variational Quantum Algorithms (VQAs), such as the Quantum Approximate Optimization Algorithm (QAOA) of [Farhi, Goldstone, Gutmann, 2014], have seen intense study towards near-term applications on quantum hardware. A crucial parameter for VQAs is the *depth* of the variational ansatz used – the smaller the depth, the more amenable the ansatz is to near-term quantum hardware in that it gives the circuit a chance to be fully executed before the system decoheres. In this work, we show that approximating the optimal depth for a given VQA ansatz is intractable. Formally, we show that for any constant $\epsilon > 0$, it is QCMA-hard to approximate the optimal depth of a VQA ansatz within multiplicative factor $N^{1-\epsilon}$, for N denoting the encoding size of the VQA instance. (Here, Quantum Classical Merlin-Arthur (QCMA) is a quantum generalization of NP.) We then show that this hardness persists in the even "simpler" QAOA-type settings. To our knowledge, this yields the first natural QCMA-hard-to-approximate problems.

2012 ACM Subject Classification Theory of computation → Quantum complexity theory

Keywords and phrases Variational quantum algorithms (VQA), Quantum Approximate Optimization Algorithm (QAOA), circuit depth minimization, Quantum-Classical Merlin-Arthur (QCMA), hardness of approximation, hybrid quantum algorithms

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.34

Related Version *Full Version*: `https://arxiv.org/abs/2211.12519`

Funding *Lennart Bittel and Martin Kliesch*: German Federal Ministry of Education and Research (BMBF), funding program "Quantum Technologies – from Basic Research to Market" via the joint project MANIQU (grant number 13N15578); Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under the grant number 441423094 within the Emmy Noether Program.
Sevag Gharibian: DFG under grant numbers 450041824 and 432788384; the BMBF within the funding program "Quantum Technologies – from Basic Research to Market" via project PhoQuant (grant number 13N16103); project "PhoQC" from the programme "Profilbildung 2020", an initiative of the Ministry of Culture and Science of the State of North Rhine-Westphalia.

Acknowledgements We thank Ashley Montanaro for helpful discussions.

1 Introduction

In the current era of Noisy Intermediate Scale Quantum (NISQ) devices, quantum hardware is (as the name suggests) limited in size and ability. Thus, NISQ-era quantum algorithm design has largely focused on *hybrid* classical-quantum setups, which ask: What types of computational problems can a classical supercomputer, paired with a *low-depth* quantum

computer, solve? This approach, typically called Variational Quantum Algorithms (VQA), has been studied intensively in recent years (see, e.g. [13, 8] for reviews), with Farhi, Goldstone and Gutmann's Quantum Approximate Optimization Algorithm (QAOA) being a prominent example [14].

More formally, VQAs roughly work as follows. One first chooses a variational ansatz (i.e. parameterization) over a family of quantum circuits. Then, one iterates the following two steps until a "suitably good" parameter setting is found:

1. Use a classical computer to optimize the ansatz parameters variationally[1].
2. Run the resulting parameterized quantum algorithm on a NISQ device to evaluate the "quality" of the chosen parameters (relative to the computational problem of interest).

The essential advantage of this setup over more traditional quantum algorithm design techniques (such as full Trotterization of a desired Hamiltonian evolution) is that one can attempt to minimize the *depth* of the ansatz used. (A formal definition of "depth" is given in Problem 1; briefly, it is the number of Hamiltonian evolutions the ansatz utilizes.) This possibility gives VQAs a potentially crucial advantage on near-term quantum hardware (i.e. noisy hardware without quantum error correction), because a NISQ device can, in principle, execute a low-depth ansatz before the system decoheres, i.e. before environmental noise destroys the "quantumness" of the computation. From an analytic perspective, low-depth ansatzes also have an important secondary benefit – VQAs of superlogarithmic depth are exceedingly difficult to analyze via worst-case complexity. Sufficiently low-depth setups, however, sometimes *can* be rigorously analyzed, with the groundbreaking QAOA work of [14] for MAX-CUT being a well-known example. Thus, estimating the optimal depth for a VQA appears central to its use in near-term applications.

1.1 Our results

In this work, we show that it is intractable to approximate the optimal depth for a given VQA ansatz, even within large multiplicative factors. Moreover, this hardness also holds for the restricted "simpler" case of the QAOA. To make our claim rigorous, we first define the VQA optimization problem we study. (Intuition to follow.)

▶ **Problem 1** (VQA minimization (MIN-VQA(k,l))). *For an n-qubit system:*

- *Input:*
  1. *Set $H = \{H_i\}$ of Hamiltonians[2], where H_i acts non-trivially only on a subset $S_i \subseteq [n]$ of size $|S_i| = k$.*
  2. *An l-local observable M acting on a subset of l qubits.*
  3. *Integers $0 \leq m \leq m'$ representing circuit depth thresholds.*
- *Output:*
  1. *YES if there exists a list of at most m angles $(\theta_1, \ldots, \theta_m) \in \mathbb{R}^m$ and a list $(G_1, \ldots, G_m)$ of Hamiltonians from H (repetitions permitted) such that $|\psi\rangle := e^{i\theta_m G_m} \cdots e^{i\theta_1 G_1}|0\cdots 0\rangle$ satisfies $\langle\psi|M|\psi\rangle \leq 1/3$.*
  2. *NO if for all lists of at most m' angles $(\theta_1, \ldots, \theta_{m'}) \in \mathbb{R}^{m'}$ and all lists $(G_1, \ldots, G_{m'})$ of Hamiltonians from H (repetitions permitted), $|\psi\rangle := e^{i\theta_{m'} G_{m'}} \cdots e^{i\theta_1 G_1}|0\cdots 0\rangle$ satisfies $\langle\psi|M|\psi\rangle \geq 2/3$.*

[1] In practice, this typically means heuristic optimization.

[2] An n-qubit Hamiltonian H is a $2^n \times 2^n$ Hermitian matrix. Any unitary operation U on a quantum computer can be generated via some Hamiltonian H and evolution time $t \geq 0$, i.e. $U = e^{iHt}$.

For intuition, recall that a VQA ansatz is a parameterization over a family of quantum circuits. Above, the ansatz is parameterized by angles θ_j, and the family of quantum circuits is generated by Hamiltonians H_j. The aim is to pick a *minimum-length* sequence of Hamiltonian evolutions $e^{i\theta_j G_j}$, so that the generated state $|\psi\rangle$ has (say) low overlap with the target observable, M. For clarity, throughout this work, by "depth" of a VQA ansatz, we are referring to the standard VQA notion of the number of Hamiltonian evolutions m applied[3]. (In the setting of QAOA, the "depth" is often referred to as the "level", up to a factor of 2.)

We remark for Problem 1 that we do not restrict the order in which Hamiltonians H_i are applied, and any H_i may be applied multiple times. Moreover, our results also hold if one defines the YES case to maximize overlap with M (as opposed to minimize overlap).

Our first result is the following.

▶ **Theorem 1.** MIN-VQA(k, l) *is* QCMA*-complete for* $k \geq 4$*,* $l = 2$*, and* $m \leq \mathsf{poly}(n)$*. Moreover, for any* $\epsilon > 0$*, it is* QCMA*-hard to distinguish between the YES and NO cases of* MIN-VQA *even if* $m'/m \geq N^{1-\epsilon}$*, where* N *is the encoding size of the instance.*

Here, Quantum-Classical Merlin-Arthur (QCMA) is a quantum generalization of NP with a classical proof and quantum verifier (formal definition in Definition 5). For clarity, the *encoding size* of the instance is the number of bits required to write down a MIN-VQA instance, i.e. to encode $H = \{H_i\}$, M, m, m' (see Problem 1). Note the encoding size is typically dominated by the encoding size of H, which may be assumed to scale as $|H|$, i.e. with the number of *interaction terms* H_i, which can be asymptotically larger than the number of qubits, n. Thus, simple gap amplification strategies such as taking many parallel copies of all interaction terms do *not* suffice to achieve our hardness ratio of $N^{1-\epsilon}$.

A direct consequence of Theorem 1 is that it is intractable (modulo the standard conjecture that BQP $\neq$ QCMA, which also implies P $\neq$ QCMA) to compute the optimum circuit depth within relative precision $N^{1-\epsilon}$:

▶ **Corollary 2** (Depth minimization)**.** *In Problem 1, let* m_{opt} *denote the minimum depth* m *such that* $\langle\psi|M|\psi\rangle \leq 1/3$*. Then, for any constant* $\epsilon > 0$*, computing estimate* $m_{\text{est}} \in [m_{\text{opt}}, N^{1-\epsilon} m_{\text{opt}}]$ *is* QCMA*-hard.*

On the other hand, even if a desired depth $m = m'$ is specified in advance, it is also QCMA-hard to find the minimizing angle and Hamiltonian sequences $(\theta_1, \ldots, \theta_m)$ and $(G_1, \ldots, G_m)$, respectively, which follows directly from Theorem 1:

▶ **Corollary 3** (Parameter optimization)**.** *Consider Problem 1 with input* $m = m'$*. Then the problem of finding angles* $(\theta_1, \ldots, \theta_m)$ *that minimize expectation* $\langle\psi|M|\psi\rangle$ *is* QCMA*-hard.*

We next turn to the special case of QAOAs. As detailed shortly under "Previous work", the study of QAOA ansatzes was initiated by [14] in the context of *quantum* approximation algorithms for MAX CUT. In that work, a QAOA is analogous to a VQA, except there are only *two* Hamiltonians $H = \{H_b, H_c\}$ given as input and M is one of those two observables (see Problem 3 for a formal definition). For clarity, here we work with a more general definition of QAOA than [14], in which neither H_b nor H_c need be diagonal in the standard

[3] Alternatively, one could consider the *circuit depth* of any simulation of the desired Hamiltonian sequence in Problem 1. The downside of this is that it would be much more difficult to analyze – one would presumably first need to convert each $e^{i\theta_j G_j}$ to a circuit U_j via a fixed choice of Hamiltonian simulation algorithm. One would then need to characterize the depth of the concatenated circuit $U_m \cdots U_1$.

basis. (In this sense, our definition is closer to the more general Quantum Alternating Operator Ansatz, also with acronym QAOA [23].) For our hardness results, it will suffice for H_b and H_c to be *k-local* Hamiltonians[4]. For QAOA, we show a matching hardness result:

▶ **Theorem 4.** MIN-QAOA(k) *is* QCMA*-complete for* $k \geq 4$ *and* $m \leq \mathsf{poly}(n)$*. Moreover, for any* $\epsilon > 0$*, it is* QCMA*-hard to distinguish between the YES and NO cases of* MIN-QAOA *even if* $m'/m \geq N^{1-\epsilon}$*, for* N *the number of strictly* k*-local terms comprising* H_b *and* H_c*.*

Note that in contrast to MIN-VQA, which is parameterized by k (the Hamiltonians' locality) and l (the observable's locality), MIN-QAOA is only parameterized by k. This is because in QAOA, the "cost" Hamiltonian H_c *itself* acts as the observable (in addition to helping drive the computation), which will be one of the obstacles we will need to overcome. For context, typically in applications of QAOA, H_c encodes (for example [14]) a MAX CUT instance.

To the best of our knowledge, Theorem 1 and Theorem 4 yield the first natural QCMA-hard to approximate problems.

1.2 Previous work

Generally speaking, it is well-known that VQA parameters are "hard to optimize", both numerically and from a theoretical perspective. We now discuss selected works from the (vast) VQA literature, and clarify how these differ from our work.

1. Theoretical studies. As previously mentioned, in 2014, Farhi, Goldstone and Gutmann proposed the Quantum Approximate Optimization Algorithm (QAOA), a special case of VQA with only two local Hamiltonians $H = \{H_b, H_c\}$ (acting on n qubits each). They showed that level-1 of the QAOA (what we call "depth 2" in Problem 1) achieves a 0.6924-factor approximation for the NP-complete MAX CUT problem. Unfortunately, worst-case analysis of higher levels has in general proven difficult, but Bravyi, Kliesch, Koenig and Tang [12] have shown an interesting negative result – QAOA to any *constant* level/depth cannot outperform the classical Goemans-Williams algorithm for MAX CUT [19]. Thus, superconstant depth is *necessary* if QAOA is to have a hope of outperforming the best classical algorithms for MAX CUT. In terms of complexity theoretic hardness, Farhi and Harrow [15] showed that even level-1 QAOA's output distribution cannot be efficiently simulated by a classical computer.

Most relevant to this paper, however, is the work of Bittel and Kliesch [9], which roughly shows that finding the optimal set of rotation angles (the θ_j in Problem 1 and Problem 3) is NP-hard. Let us clearly state how the present work differs from [9]:

1. [9] fixes both the depth of the VQA and the precise sequence of Hamiltonians H_i to be applied as part of the input. It then asks: What is the complexity of computing the optimal rotation angles θ_i so as to minimize overlap with a given observable?
   In contrast, our aim here is to study the complexity of optimizing the *depth* itself. Thus, Problem 1 does not fix the depth m, nor the order/multiplicity of application of any of the Hamiltonian terms.

[4] A k-local n-qubit Hamiltonian H is a quantum analogue of a MAX-k-SAT instance, and can be written $H = \sum_i H_i$, with each "quantum clause" H_i acting non-trivially on some subset of k qubits. Strictly speaking, each H_i is tensored with the identity matrix on $n - k$ qubits to ensure all operators in the sum have the correct dimension.

2. [9] shows that optimizing the rotation angles in QAOA is NP-hard, *even if* one is allowed to work in time polynomial in the *dimension* of the system. (Formally, this is obtained by reducing a MAX CUT instance of encoding size N to QAOA acting on $\log(N)$ qubits.) In contrast, we work in the standard setting of allowing only poly-time computations in the number of qubits, n, not the dimension. In return, we obtain stronger hardness results, both in that NP $\subseteq$ QCMA (and thus QCMA-hardness is a stronger statement than NP-hardness[5]), and in that we show hardness of approximation up to any multiplicative factor $N^{1-\epsilon}$.

2. Practical/numerical studies. For clarity, numerical studies are not directly related to our work. However, due to the intense practical interest in VQA for the NISQ era, for completeness we next survey some of the difficulties encountered when optimizing VQAs on the numerical side. For this, note that VQAs are typically used to solve problems which can be phrased as energy optimization problems (such as NP-complete problems like MAX CUT [14]).

In this direction, two crucial problems can arise in the classical optimization part of the standard VQA setup: (i) barren plateaus [30], which lead to vanishing gradients, and (ii) local minima [9], many of which can be highly non-optimal. Such unwanted local minima are also called *traps*. In order to counterbalance these challenges, heuristic optimization strategies have led to promising results in relevant cases but with not too many qubits. Initialization-dependent barren plateaus [30] can be avoided by tailored initialization [40], and there are indications that barren plateaus are a less significant challenge than traps [3]. In general, the optimization can be improved using natural gradients [36], multitask learning type approach [39], optimization based on trigonometric model functions [26], neural network-based optimization methods [31], brick-layer structures of generic unitaries [32], and operator pool-based methods [22, 11]. ADAPT-VQEs [22] iteratively grow the VQA's parametrized quantum circuit (PQC) by adding operators from a pool that have led to the largest derivative in the previous step. This strategy allows one to avoid barren plateaus and even "burrow" out of some traps [21]. CoVar [11] is based on similar ideas complemented with estimating several properties of the variational state in parallel using classical shadows [24]. The optimization strategies are of a heuristic nature, and analytic results are scarce. Finally, it has been numerically observed [33, 37] and analytically shown [27] that VQA-type ansätze become almost free from traps when the ansatz is overparameterized. Our work implies that these practical approaches cannot work for all instances and, therefore, provides a justification to resort to such heuristics.

1.3 Techniques

We focus on techniques for showing QCMA-hardness of approximation, as containment in QCMA is straightforward[6] for both MIN-VQA and MIN-QAOA.

To begin, recall that in a QCMA proof system (Definition 5), given a YES input, there exists a poly-length *classical* proof y causing a quantum poly-size circuit V to accept, and for a NO input, all poly-length proofs y cause V to reject. Our goal is to embed such proof systems into instances of Problem 1 and Problem 3, while maintaining a large promise

[5] Note that for $\log(N)$-size instances of QAOA as in [9], one cannot hope for more than NP-hardness, since both Hamiltonians H_b and H_c have polynomial dimension, and thus can be classically simulated efficiently. Thus, such instances are verifiable in NP.

[6] The prover sends angles θ_j, and the verifier simulates each $e^{i\theta_j H_j}$ via known Hamiltonian simulation algorithms [29].

gap ratio m'/m. To do so, we face three main challenges: (1) Where will hardness of approximation come from? Typically, one requires a PCP theorem [5, 6] for such results, which remains a notorious open question for both QCMA and QMA[7] [1]. (2) Problem 1 places no restrictions on which Hamiltonians are applied, in which order, and with which rotation angles. How can one enforce computational structure given such flexibility? In addition, MIN-QAOA presents a third challenge: (3) How to overcome the previous two challenges when we are only permitted two Hamiltonians, H_b and H_c, the latter of which must also act as the observable?

To address the first challenge, we appeal to the hardness of approximation work of Umans [34]. The latter showed how to use a graph-theoretical construct, known as a *disperser*, to obtain strong hardness of approximation results for Σ_2^p (the second level of the Polynomial-Time Hierarchy). Hiding at the end of that paper is Theorem 9, which showed that the techniques therein also apply to yield hardness of approximation within factor $N^{1/5-\epsilon}$ for a rather artificial NP-complete problem. Gharibian and Kempe [18] then showed that [34] can be extended to obtain hardness of approximation results for a quantum analogue of Σ_2^p, and also obtained QCMA-hardness of approximation within $N^{1-\epsilon}$ for an even more artificial problem, Quantum Monotone Minimum Satisfying Assignment (QMSA, Problem 2). Roughly, QMSA asks – given a quantum circuit V accepting a monotone set (Definition 6) of strings, what is the smallest Hamming weight string accepted by V? Here, our approach will be to construct many-one reductions from QMSA to MIN-VQA and MIN-QAOA, where we remark that maintaining the $N^{1-\epsilon}$ hardness ratio (i.e. making the reduction approximation-ratio-preserving) will require special attention.

1. The reduction for MIN-VQA. To reduce a given QMSA circuit $V = V_L \cdots V_1$ to a VQA instance $(\{H_i\}, M, m, m')$, we utilize a "hybrid Cook-Levin + Kitaev" circuit-to-Hamiltonian construction, coupled with a *pair* of clocks (whereas Kitaev [25] requires only one clock). Here, a *non-hybrid* (i.e. standard) circuit-to-Hamiltonian construction is a quantum analogue of the Cook-Levin theorem, i.e. a map from quantum circuits V to local Hamiltonians H_V, so that there exists a proof $|\psi\rangle$ accepted by V if and only if H_V has a low-energy[8] "history state", $|\psi_{\text{hist}}\rangle$. A history state, in turn, is a quantum analogue of a Cook-Levin tableau, except that each time step of the computation is encoded in superposition via a clock construction of Feynman [16]. In contrast, our construction is "hybrid" in that it uses a clock register like Kitaev, but does not produce a history state in superposition over all time steps, like Cook-Levin. A bit more formally, the Hamiltonians $\{H_i\}$ of our VQA instance act on four registers, $ABCD$, denoting proof (A), workspace (B), clock 1 (C), and clock 2 (D). To an honest prover, these Hamiltonians $\{H_i\}$ may be viewed as being partitioned into two sets: Hamiltonians for "setting proof bits", denoted P, and Hamiltonians for simulating gates from V, denoted Q. An example of a Hamiltonian in P is $P_j := X_{A_j} \otimes |1\rangle\langle 1|_{C_j} \otimes |1\rangle\langle 1|_{D_{|D|}}$ which says: If clock 1 (register C) is at time j *and* clock 2 (register D) is at time $|D|$ (more on clock 2 shortly), then flip the jth qubit of register A via a Pauli X gate. An example of a Hamiltonian in Q is

$$Q_j := (V_j)_{AB} \otimes |01\rangle\langle 10|_{C_{|A|+j},C_{|A|+j+1}} + (V_j^\dagger)_{AB} \otimes |10\rangle\langle 01|_{C_{|A|+j},C_{|A|+j+1}}, \tag{1}$$

which allows the prover to apply gate V_j of V to registers AB, while updating clock 1 from time $|A| + j$ to $|A| + j + 1$. In this first (insufficient) attempt at a reduction, the honest prover for MIN-VQA acts as follows: First, apply a subset of the P Hamiltonians to prepare

[7] Quantum Merlin-Arthur (QMA) is QCMA but with a quantum proof.

[8] By "energy" of a state $|\psi\rangle$ against Hamiltonian H, one means the expectation $\langle\psi|H|\psi\rangle$, whose minimum possible value is precisely $\lambda_{\min}(H)$, i.e. the smallest eigenvalue of H.

the desired input y to the QMSA verifier V in register A, and then evolve Hamiltonians Q_1 through Q_L to simulate gates V_1 through V_L on registers A and B. The observable M is then defined to measure the designated output qubit of B in the standard basis, conditioned on C being at time T.

The crux of this (honest prover) setup is that if we start with a YES (respectively, NO) instance of QMSA, then the Hamming weight of the optimal y is at most g (respectively, at least g'), for $g'/g \geq N_{\mathsf{QMSA}}^{1-\epsilon}$ and N_{QMSA} the encoding size of the QMSA instance. This, in turn, means that the VQA prover applies at most g Hamiltonians from P (YES case), or at least g' Hamiltonians from P (NO case). The problem is that the prover must *also* apply Hamiltonians Q_1 through Q_L in order to simulate the verifier, V, and so we have hardness ratio $m'/m = (g'+L)/(g+L) \to 1$ if $L \in \omega(g)$, as opposed to $N^{1-\epsilon}$!

To overcome this, we make flipping each bit of P "more costly" by utilizing a *2D clock setup*. This, in turn, will ensure the hardness ratio $(g'+L)/(g+L)$ becomes (roughly) $\frac{g'|D|+L}{g|D|+L} \approx \frac{g'}{g}$ for $|D| \in \omega(L)$, as desired. Specifically, to flip bit A_j for any j, we force the prover to first sequentially increment the *second* clock, D, from 1 to $|D|$. By definition, P_j can now flip the value of A_j – but it cannot increment time in C (i.e. we remain in time step j on clock 1). This next forces the prover to decrement D from $|D|$ back to 1, at which point a separate Hamiltonian (not displayed here) can increment clock C from j to $j+1$. The entire process then repeats itself to flip bit A_{j+1}. What is crucial for our desired approximation ratio is that we only have a single copy of register D, i.e. we re-use it to flip each bit A_j, thus effectively making CD act as a 2D clock. This ensures the added overhead to the encoding size of the VQA instance scales as $|D|$, not $|A|\,|D|$, which is what one would obtain if CD encoded a 1D clock (i.e. if each A_j had a *separate* copy of D).

Finally, to show soundness against provers deviating from the honest strategy above, we first establish that any sequence of evolutions from $\{H_i\}$ keeps us in a desired logical computation space, i.e. the span of vectors of form

$$S \coloneqq \Big\{ V_{s-|A|} \cdots V_1 |y\rangle_A |0\cdots 0\rangle_B |\widetilde{s}\rangle_C |\widetilde{t}\rangle_D \mid y \in \{0,1\}^{|A|}, s \in \{1,\ldots,|C|\}, t \in \{1,\ldots,|D|\} \Big\},$$

for $|y\rangle_A$ the "proof string" prepared via P-gates and $\widetilde{s}$ and $\widetilde{t}$ the unary representations of time steps s and t in clocks 1 and 2, respectively. We then show that applying too few Hamiltonian evolutions from $\{H_i\}$ results in a state with either no support on large Hamming weight strings y (meaning the verifier V must reject in the NO case), or no support on states with a fully executed verification circuit $V = V_L \cdots V_1$ (in which case we design V to reject).

2. The reduction for MIN-QAOA. At a high level, our goal is to mimic the reduction to MIN-VQA above. However, the fact that we have only two Hamiltonians at our disposal, H_b (driving Hamiltonian) and H_c (cost Hamiltonian), and no separate observable M, complicates matters. Very roughly, our aim is to *alternate* even and odd steps of the honest prover's actions from MIN-VQA, so that H_b simulates the even steps, and H_c the odd ones. To achieve this requires several steps:

1. First, we modify the MIN-VQA setup so that all the odd (respectively, even) local terms H_i pairwise commute. This ensures that the actions of $\exp(i\theta H_b)$ and $\exp(i\theta H_c)$ can be analyzed, since H_b and H_c will consist of sums of (now commuting) H_i terms.
2. In MIN-VQA, all Hamiltonians satisfied $H_i^2 = I$, which intuitively means an honest prover could use H_i to either act trivially ($\theta_i = 0$) or perform some desired action ($\theta_i = \pi$). For MIN-QAOA, we instead require a trick inspired by [9] – we introduce certain local terms G_j (Equation (27)) with 3-cyclic behavior. In words, the honest prover can induce *three* logical actions from such G_j, obtained via angles $\theta_j \in \{0, \pi/3, 2\pi/3\}$, respectively.

3. We next add additional constraints to H_b to ensure its unique ground state encodes the correct start state (see Equation (23) of Problem 3). This is in contrast to MIN-VQA, where the initial state $|0\cdots 0\rangle$ is fixed and independent of the H_i.
4. Finally, the observable M is added as a local term to H_c, but scaled larger than all other terms in H_c. This ensures that for any state $|\psi\rangle$, $|\langle\psi|H_c - M|\psi\rangle|$ is "small", so that measuring cost Hamiltonian H_c once the QAOA circuit finishes executing is "close" to measuring M.

As for soundness, the high-level approach is similar to MIN-VQA, in that we analyze a logical space of computation steps, akin to the definition of S, and track Hamming weights of prepared proofs in this space. The analysis, however, is more involved, as the construction itself is more intricate than for MIN-VQA. For example, a new challenge for our MIN-QAOA construction is that evolving by a Hamiltonian (specifically, H_c) does *not* necessarily preserve the logical computation space. We thus need to prove that we may "round" each intermediate state in the analysis back to the logical computation space, in which we can then track the Hamming weight of the proof y (Lemma 20).

1.4 Open questions

We have shown that the optimal depth of a VQA or QAOA ansatz is hard to approximate, even up to large multiplicative factors. A natural question is whether similar NP-*hardness* of approximation results for depth can be shown when (e.g.) the cost Hamiltonian in QAOA is classical, such as in [14]? Since we aimed here to capture the strongest possible hardness result, i.e. for QCMA, our Hamiltonians were necessarily not classical/diagonal. Second, although our results are theoretical worst-case results, VQAs are of immense practical interest in the NISQ community. Can one design good heuristics for optimal depth approximation which often work well in practice? Third, can one approximate the optimal depth for QAOA on *random* instances of a computational problem? Here, for example, recent progress has been made by Basso, Gamarnik, Mei and Zhou [7], Boulebnane and Montanaro [10], and Anshu and Metger [4], which give analytical bounds on the success probability of QAOA at various levels and on random instances of various constraint satisfaction problems, for instance size n going to infinity. The bounds of [4], for example, show that even *superconstant* depth (i.e. scaling as $o(\log\log n)$) is insufficient for QAOA to succeed with non-negligible probability for a random spin model. On a positive note, we remark that [10] give numerical evidence (based on their underlying analytical bounds) that at around level 14, QAOA begins to surpass existing classical SAT solvers for the case of random 8-SAT. Fourth, we have given the first natural QCMA-hard to approximate problems. What other QCMA-complete problems can be shown hard to approximate? A natural candidate here is the *Ground State Connectivity* problem [17, 20, 35], whose hardness of approximation we leave as an open question. Finally, along these lines, can a PCP theorem for QCMA be shown as a first stepping stone towards a PCP theorem for QMA?

Organization. In Section 2, we show Theorem 1. Section 3 shows Theorem 4. All omitted proofs are in the full version.

2 QCMA-hardness of approximation for VQAs

In this section, we show Theorem 1. We begin in Section 2.1 with relevant definitions and lemmas. Section 2.2 proves Theorem 1.

2.1 Definitions and required facts

Throughout, the relation $:=$ denotes a definition, and $[n] := \{1, 2, \ldots, n\}$. We use $|x|$ to specify the length of a vector or string or the cardinality of set x. The term I_A denotes the identity operator/matrix on qubits with indices in register A. By $\|H\|_\infty$ we denote the spectral norm of an operator H acting on $\mathbb{C}^d$, i.e. $\max_{|\psi\rangle\in\mathbb{C}^d} \frac{\|H|\psi\rangle\|_2}{\||\psi\rangle\|_2}$, for $\|\cdot\|_2$ the standard Euclidean norm. The trace norm of an operator is denoted by $\|\cdot\|_{\mathrm{tr}}$. e_i refers to a computational basis state.

▶ **Definition 5** (Quantum-classical Merlin-Arthur (QCMA)). *Let $\Pi = (\Pi_{\mathrm{yes}}, \Pi_{\mathrm{no}})$ be a promise problem. Then $\Pi \in$ QCMA if and only if there is a polynomial p such that for any $x \in \Pi$ there exists a quantum circuit V_x of size $p(|x|)$ with one designated output qubit satisfying:*

(i) *If $x \in \Pi_{\mathrm{yes}}$ there exists a string $y \in \{0,1\}^{p(|x|)}$ such that $\Pr[V_x \text{ accepts } y] \geq 2/3$ and*

(ii) *if $x \in \Pi_{\mathrm{no}}$ and all strings $y \in \{0,1\}^{p(|x|)}$ it holds that $\Pr[V_x \text{ accepts } y] \leq 1/3$.*

Often, it is helpful to separate the qubits into an a *proof register* A, which contains the classical proof $|y\rangle$, and an *ancilla/work register* B, which is initialized in the $|0\rangle$ state. Then the acceptance probability can be expressed as $\Pr[V_x \text{ accepts } (x,y)] = \langle y;0|V_x^{(n)\dagger} M^{(B_1)} V_x^{(n)}|y;0\rangle$, where the measurement is given by an operator $M^{(B_1)}$ acting on the first qubit of the work register B.

QCMA was first defined in [2], and satisfies NP $\subseteq$ QCMA $\subseteq$ QMA. QCMA-complete problems include Identity Check on Basis States (i.e. "does a quantum circuit act almost as the identity on all computational basis states?") [38] and Ground State Connectivity (GSCON) (i.e. is the ground space of a local Hamiltonian "connected"?) [17]. The latter remains hard (specifically, $\mathsf{QCMA}_{\mathrm{EXP}}$-hard) in the 1D translation-invariant setting [35]. Next, we will introduce a QCMA-complete problem related to monotone sets.

▶ **Definition 6** (Monotone set). *A set $S \subseteq \{0,1\}^n$ is called* monotone *if for any $x \in S$, any string obtained from x by flipping one or more zeroes in x to one is also in S.*

▶ **Definition 7** (Quantum circuit accepting monotone set). *Let V be a quantum circuit consisting of 1- and 2-qubit gates, which takes in an n-bit classical input register, m-qubit ancilla register initialized to all zeroes, and outputs a single qubit, q. For any input $x \in \{0,1\}^n$, we say V* accepts *(respectively,* rejects*) x if measuring q in the standard basis yields 1 (respectively, 0) with probability at least $1 - \epsilon_Q$ (If not specified, $\epsilon_Q = 1/3$). We say V accepts a* monotone set *if the set $S \subseteq \{0,1\}^n$ of all strings accepted by V is a monotone.*

▶ **Problem 2** (QUANTUM MONOTONE MINIMUM SATISFYING ASSIGNMENT (QMSA)). *Given a quantum circuit V accepting a non-empty monotone set $S \subseteq \{0,1\}^n$, and integer thresholds $0 \leq g \leq g' \leq n$, output:*

- *YES if there exists an $x \in \{0,1\}^n$ of Hamming weight at most g accepted by V.*
- *NO if all $x \in \{0,1\}^n$ of Hamming weight at most g' are rejected by V.*

▶ **Theorem 8** (Gharibian and Kempe [18]). *QMSA is QCMA-complete, and moreover it is QCMA-hard to decide whether, given an instance of QMSA, the minimum Hamming weight string accepted by V is at most g or at least g' for $g'/g \in O(N^{1-\epsilon})$ (where $g' \geq g$).*

In words, QMSA is QCMA-hard to approximate within $N^{1-\epsilon}$ for any constant $\epsilon > 0$, where N is the encoding size of the QMSA instance.

2.2 QCMA-completeness

▶ **Theorem 1.** MIN-VQA(k,l) *is* QCMA*-complete for $k \geq 4$, $l = 2$, and $m \leq$ poly(n). Moreover, for any $\epsilon > 0$, it is* QCMA*-hard to distinguish between the YES and NO cases of* MIN-VQA *even if $m'/m \geq N^{1-\epsilon}$, where N is the encoding size of the instance.*

In words, it is QCMA-hard to decide whether, given an instance of MIN-VQA, the variational circuit can prepare a "good" ansatz state with at most m evolutions, or if all sequences of m' evolutions fail to prepare a "good" ansatz state, for $m'/m \in O(N^{1-\epsilon})$ (where $m' \geq m$).

Proof. Containment in QCMA is straightforward; the prover sends the angles θ_i and indices of Hamiltonians H_i to evolve, which the verifier then completes using standard Hamiltonian simulation techniques [28, 29]. We now show QCMA-hardness of approximation. Let $\Pi' = (V', g, g')$ be an instance of QMSA, for $V' = V'_{L'} \cdots V'_1$ a sequence of L' 2-qubit gates taking in n'_V input bits and m'_V ancilla qubits.

Preprocessing V'. Suppose V' takes in n'_V input qubits in register A' and m'_V ancilla qubits in register B'. To ease our soundness analysis, we make two assumptions about V' without loss of generality:

▶ **Assumption 9.** *V' only* reads *register A', but does not write to it. To achieve this, add n'_V ancilla qubits (initialized to $|0\rangle$) to B', and prepend V' with n'_V CNOT gates applied transversally to copy input x from A' to the added ancilla qubits in B'. Update any subsequent gate which acts on the original input x to instead act on its copied version in B'.*

▶ **Assumption 10.** *The output qubit of V' is set to $|0\rangle$ until V'_L is applied. For this, add a single ancilla qubit to B' initialized to $|0\rangle$, and treat this as the new designated output qubit. Append to the end of V' a CNOT gate from its original output wire to the new output wire.*

Call the new circuit with all modifications V. V acts on $n_V := n'_V$ input qubits, $m_V := m'_V + n'_V + 1$ ancilla qubits, and consists of $L := L' + n'_V + 1$ gates.

Proof organization. The remainder of the proof is organized as follows. Section 2.2.1 constructs the MIN-VQA instance. Section 2.2.2 proves observations and lemmas required for the completeness and soundness analyses. Sections 2.2.3 and 2.2.4 show completeness and soundness, respectively. Finally, Section 2.2.5 analyzes the hardness ratio achieved. All omitted proofs are in the full version.

2.2.1 The MIN-VQA instance

We now construct our instance Π of MIN-VQA as follows. Π acts on a total of n qubits, which we partition into 4 registers: A (proof), B (workspace), C (clock 1), and D (clock 2). Register A consists of n_V qubits, B of m_V qubits, C of $L + n_V + 1$ qubits, and D of $\lceil L^{1+\delta} \rceil$ qubits for some fixed $0 < \delta < 1$ chosen at the end of the proof in Section 2.2.5.

Our construction will ensure that C (respectively, D) always remains in the span of logical time steps, $\mathcal{T}_C := \{|\widetilde{s}\rangle\}_{s=1}^{|C|}$ (respectively, $\mathcal{T}_D := \{|\widetilde{t}\rangle\}_{t=1}^{|D|}$), defined as:

$$|\widetilde{s}\rangle := |0\rangle^{\otimes s-1}|1\rangle|0\rangle^{\otimes |C|-s} \quad \text{for } 1 \leq s \leq |C| \tag{2}$$

$$|\widetilde{t}\rangle = |0\rangle^{\otimes t-1}|1\rangle|0\rangle^{\otimes |D|-t} \quad \text{for } 1 \leq t \leq |D|\,. \tag{3}$$

For example for C, $|\widetilde{1}\rangle = |1\rangle|0\rangle^{\otimes |C|-1}$, $|\widetilde{2}\rangle = |0\rangle|1\rangle|0\rangle^{\otimes |C|-2}$, $|\widetilde{3}\rangle = |00\rangle|1\rangle|0\rangle^{\otimes |C|-3}$, and so forth. Note this differs from the usual Kitaev unary clock construction, which encodes time t via $|1\rangle^{\otimes t}|0\rangle^{\otimes N-t}$ [25]. This allows us to reduce the locality of our Hamiltonian.

Throughout, we use (e.g.) C_j to refer to qubit j and $C_{i,j}$ and qubits i and j of register C. All qubits not explicitly mentioned are assumed to be acted on by the identity. Define four families of Hamiltonians as follows:

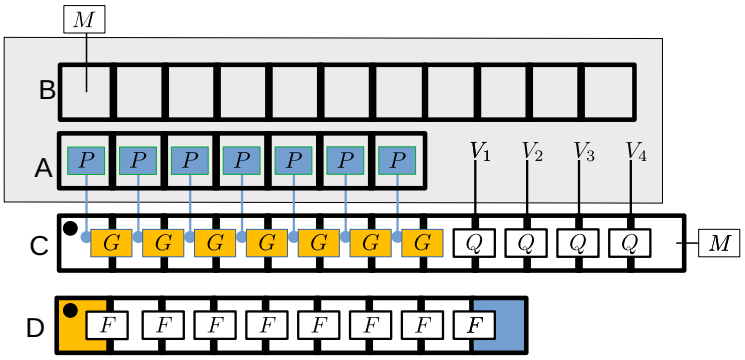


Figure 1 Sketch describing the VQA instance. A colored square (say, blue) at index j of a register means that register's jth qubit must be in $|1\rangle$ for any blue gates to act non-trivially. So, for example, the G gates increment the first clock register C, but only if the D register is in the state $|1\rangle_{D_1}$. For the initial state, C_1 and D_1 are in the $|1\rangle$ state, marked by a black dot. The gates F increment the second clock register D. The P gates are controlled operations on the C register, which perform X operations on the A register, but only if D is in the state $|1\rangle_{D_{|D|}}$. The Q gates increment the clock register C, while also applying the circuit $V_1, \ldots, V_L$ on the AB registers. The measurement operator M acts on the B_1 and $C_{|C|}$ qubit.

- (F) For propagation of the second clock, D, define 2-local Hamiltonians as

$$F_j \coloneqq |01\rangle\langle 10|_{D_{j,j+1}} + |10\rangle\langle 01|_{D_{j,j+1}} \text{ for all } j \in \{1, \ldots, |D| - 1\}. \tag{4}$$

- (G) For propagation of the first clock, C, define 3-local Hamiltonians as

$$G_j \coloneqq \left(|01\rangle\langle 10|_{C_{j,j+1}} + |10\rangle\langle 01|_{C_{j,j+1}}\right) \otimes |1\rangle\langle 1|_{D_1} \text{ for all } j \in \{1, \ldots, |A|\}. \tag{5}$$

- (P) For each qubit $j \in \{1, \ldots, |A|\}$ of A, define 3-local Hamiltonian as

$$P_j \coloneqq X_{A_j} \otimes |1\rangle\langle 1|_{C_j} \otimes |1\rangle\langle 1|_{D_{|D|}}. \tag{6}$$

- (Q) For each gate V_k for $k \in \{1, \ldots, L\}$, let R_k denote the two qubits of AB which V_k acts on. Define 4-local Hamiltonians as

$$Q_k \coloneqq (V_k)_{R_k} \otimes |01\rangle\langle 10|_{C_{|A|+k,|A|+k+1}} + (V_k^\dagger)_{R_k} \otimes |10\rangle\langle 01|_{C_{|A|+k,|A|+k+1}}. \tag{7}$$

Denote the union of these four sets of Hamiltonians as $S_{FGPQ} \coloneqq F \cup G \cup P \cup Q$. Set a 2-local observable

$$M \coloneqq I - |1\rangle\langle 1|_{B_1} \otimes |1\rangle\langle 1|_{C_{|C|}} \tag{8}$$

where we assume without loss of generality that V outputs its answer on qubit B_1. Set $m = g \cdot (2\,|D| - 1) + |A| + L$, $m' = g' \cdot (2\,|D| - 1) + |A| + L$. To aid the reader in the remainder of the proof, all definitions above are summarized in Table 1.

It remains to choose our initial state. Strictly speaking, Problem 1 mandates initial state $|0 \cdots 0\rangle_{ABCD}$. However, to keep notation simple, it will be convenient to instead choose

$$|\phi\rangle \coloneqq |0 \cdots 0\rangle_{AB} |10^{|C|-1}\rangle_C |10^{|D|-1}\rangle_D = |0 \cdots 0\rangle_{AB} |\widetilde{1}\rangle_C |\widetilde{1}\rangle_D, \tag{9}$$

Table 1 Terms used in the proof of Theorem 1.

Term	Description	Properties
V'	Input QMSA instance's verification circuit	$V' = V'_{L'} \cdots V'_1$
L'	Number of 1- and 2-qubit gates in V'	
n'_V	Number of proof qubits taken in by V'	
m'_V	Number of ancilla qubits taken in by V'	
g,g'	YES/NO thresholds for QMSA instance, resp.	
V	QMSA verifier obtained from V' via Assump. 9 and 10	$V = V_L \cdots V_1$
L	Number of 1- and 2-qubit gates in V	$L = L' + n'_V + 1$
n_V	Number of proof qubits taken in by V	$n_V = n'_V$
m_V	Number of ancilla qubits taken in by V	$m_V = m'_V + n'_V + 1$
A	Proof register	$\lvert A\rvert = n_V$
B	Workspace register	$\lvert B\rvert = m_V$
C	Clock 1 register	$\lvert C\rvert = L + n_V + 1$
D	Clock 2 register	$\lvert D\rvert = \lceil L^{1+\delta} \rceil$, see Section 2.2.5 for δ
F	Propagation terms for clock 2	Act on register D, $\lvert F\rvert = \lvert D\rvert - 1$
G	Propagation terms for clock 1	Act on registers C, D, $\lvert g\rvert = \lvert A\rvert$
P	Hamiltonian terms for setting proof bits	Act on registers A, C, D, $\lvert P\rvert = \lvert A\rvert$
Q	Hamiltonian terms for simulating verifier gates, V_k	Act on registers A, B, C, $\lvert Q\rvert = L$
M	Observable for MIN-VQA instance	$M \coloneqq I - \lvert 1\rangle\langle 1\rvert_{B_1} \otimes \lvert 1\rangle\langle 1\rvert_{C_{\lvert C\rvert}}$
m,m'	YES/NO thresholds for MIN-VQA instance, resp.	$m = g \cdot (2\lvert D\rvert - 1) + \lvert A\rvert + L$, $m' = g' \cdot (2\lvert D\rvert - 1) + \lvert A\rvert + L$.

i.e. with the two clock registers C and D initialized to their starting clock state, $|\widetilde{1}\rangle$. This is without loss of generality – we may, in fact, start with *any* standard basis state as our initial state without requiring major structural changes to our construction, as the following observation states.

▶ **Observation 11.** *Fix any standard basis state* $|x\rangle_{ABCD} = \overline{X}|0\cdots 0\rangle_{ABCD}$*, for* $\overline{X} \coloneqq X_1^{x_1} \otimes \cdots \otimes X_N^{x_N}$ *with* $N \coloneqq |A| + |B| + |C| + |D|$*. Consider the updated set* $S'_{FGPQ} \coloneqq \left\{\overline{X}H\overline{X} \mid H \in S_{FGPQ}\right\}$*, where for simplicity we match* $H \in S_{FGPQ}$ *with* $H' \coloneqq \overline{X}H\overline{X} \in S'_{FGPQ}$*. Then, for any* $m \in \mathbb{N}$*, and any sequence* $(H_t)_{t=1}^m$ *of Hamiltonians from* S_{FGPQ}*,*

$$e^{i\theta_m H_m} \cdots e^{i\theta_2 H_2} e^{i\theta_1 H_1} |x\rangle_{ABCD} = e^{i\theta_m H'_m} \cdots e^{i\theta_2 H'_2} e^{i\theta_1 H'_1} |0\cdots 0\rangle_{ABCD}. \tag{10}$$

Moreover, each H *and* H' *have the same locality.*

2.2.2 Helpful observations and lemmas

We next state all observations and technical lemmas for the later correctness analysis of our construction. All omitted proofs are in the full version.

▶ **Observation 12.** *For all $\theta \in \mathbb{R}$, and all $F_j \in F$, $G_j \in G$, $P_j \in P$ and $Q_k \in Q$,*

$$e^{i\theta F_j} = \cos(\theta)(|01\rangle\langle 01| + |10\rangle\langle 10|)_{D_{j,j+1}} + i\sin(\theta)F_j + (I - |01\rangle\langle 01| - |10\rangle\langle 10|)_{D_{j,j+1}} \tag{11}$$

$$e^{i\theta G_j} = \cos(\theta)\left(|01\rangle\langle 10|_{C_{j,j+1}} + |10\rangle\langle 01|_{C_{j,j+1}}\right) \otimes |1\rangle\langle 1|_{D_1} + i\sin(\theta)G_j + \left(I - \left(|01\rangle\langle 10|_{C_{j,j+1}} + |10\rangle\langle 01|_{C_{j,j+1}}\right) \otimes |1\rangle\langle 1|_{D_1}\right) \tag{12}$$

$$e^{i\theta P_j} = (\cos(\theta)I + i\sin(\theta)X)_{A_j} \otimes |1\rangle\langle 1|_{C_j} \otimes |1\rangle\langle 1|_{D_{|D|}} + (I - |1\rangle\langle 1|_{C_j} \otimes |1\rangle\langle 1|_{D_{|D|}}) \tag{13}$$

$$e^{i\theta Q_k} = \cos(\theta)I_{AB} \otimes (|01\rangle\langle 01| + |10\rangle\langle 10|)_{C_{|A|+k,|A|+k+1}} + i\sin(\theta)Q_k + I_{AB} \otimes (I - |01\rangle\langle 01| - |10\rangle\langle 10|)_{C_{|A|+k,|A|+k+1}}. \tag{14}$$

Any register not explicitly listed in equations above is assumed to be acted on by identity.

▶ **Definition 13** (Support only on logical time steps)**.** *We say state $|\psi\rangle_{ABCD}$ is* supported only on logical time steps *if it can be written $|\psi\rangle_{ABCD} = \sum_{s=1}^{|C|}\sum_{t=1}^{|D|}\alpha_{st}|\eta_{st}\rangle_{AB}|\widetilde{s}\rangle_C|\widetilde{t}\rangle_D$ for unit vectors $|\eta_{st}\rangle$ and $\sum_{st}|\alpha_{st}|^2 = 1$, and $|\widetilde{s}\rangle \in \mathcal{T}_C$ and $|\widetilde{t}\rangle \in \mathcal{T}_D$ defined as in Equation* (2) *and Equation* (3)*, respectively.*

▶ **Observation 14.** *Recall that the initial state $|\phi\rangle = |0\cdots 0\rangle_{AB}|\widetilde{1}\rangle_C|\widetilde{1}\rangle_D$ is supported only on logical time steps. Then, for any $m \in \mathbb{N}$ and sequence of evolutions $\exp(i\theta_j H_j)$ for $\theta_j \in \mathbb{R}$ and $H_j \in S_{FGPQ}$, $e^{i\theta_m H_m}\cdots e^{i\theta_2 H_2}e^{i\theta_1 H_1}|\phi\rangle$ is supported only on logical time steps.*

The following lemma tells us that any sequence of Hamiltonian evolutions $\exp(i\theta_u H_u)$ on initial state $|\phi\rangle$ remains in a certain logical computation space.

▶ **Lemma 15.** *Define*

$$S := \left\{V_{s-|A|}\cdots V_1|y\rangle_A|0\cdots 0\rangle_B|\widetilde{s}\rangle_C|\widetilde{t}\rangle_D \mid y \in \{0,1\}^{|A|}, s \in \{1,\ldots,|C|\}, t \in \{1,\ldots,|D|\}\right\}, \tag{15}$$

where we adopt the convention that the V gates are present only when $s > |A|$. Then, for any $m \in \mathbb{N}$, $\Pi_{u=1}^m e^{i\theta_u H_u}|\phi\rangle \in \mathrm{Span}(S)$ for any angles $\theta_u \in \mathbb{R}$ and sequence of Hamiltonians $H_u \in S_{FGPQ}$.

Next, we relate the circuit depth of a state generated by our VQA to the Hamming weight of the proof string y.

▶ **Lemma 16.** *Let $(H_u)_{u=1}^m$ be a sequence of Hamiltonians drawn from S_{FGPQ} which maps the initial state* (9) *to $|\phi_m\rangle := \Pi_{u=1}^m e^{i\theta_u H_u}|\phi\rangle$. Suppose $|\phi_m\rangle$ has non-zero overlap with some $|\eta_{y,s,t}\rangle$ with y of Hamming weight at least w and $s = |A| + 1$. Then, $m \geq w(2|D| - 1) + |A|$ with at least $w(2|D| - 1) + |A|$ of the H_u drawn from $F \cup G \cup P$.*

Finally, the next lemma ensures that any prover applying fewer than L Hamiltonians from Q cannot satisfy the YES case's requirements for MIN-VQA.

▶ **Lemma 17.** *For any $m \in \mathbb{N}$, let $(H_u)_{u=1}^m$ be any sequence of Hamiltonians drawn from S_{FGPQ} and containing strictly fewer than L Hamiltonians from Q. Then, for observable $M = I - |1\rangle\langle 1|_{B_1} \otimes |1\rangle\langle 1|_{C_{|C|}}$, the state $|\phi_m\rangle := \Pi_{u=1}^m e^{i\theta_u H_u}|0\cdots 0\rangle_{ABC}$ satisfies*

$$\langle\phi_m|M|\phi_m\rangle = 1. \tag{16}$$

Proof. By Lemma 15, $|\phi_m\rangle \in S$ for S from Equation (15). Next, by Observation 12, Hamiltonians from $F \cup P$ act invariantly on clock C, and Hamiltonians from G can only increment C from 1 (i.e. its initial value in $|\phi\rangle$) to $|A|+1$. The observable M, however, acts non-trivially only when C is set to $|C| = |A| + L + 1$. The only Hamiltonians which can increment C from $|A|+1$ to $|A|+L+1$ are those from Q. Each such $H_s \in Q$ can map C from time $|A|+s$ to $|A|+s+1$ or vice versa, for $s \in \{1,\ldots,L\}$. Thus, since strictly fewer than L of the H_u chosen are from Q, it follows that $|\phi_m\rangle$ has no support on time step $|C| = |A| + L + 1$, i.e. $(I_{AB} \otimes |1\rangle\langle 1|_{C_{|C|}})|\phi_m\rangle = 0$. The claim now follows since we Assumption 10 says verifier $V = V_L \cdots V_1$ has its output qubit, denoted B_1, set to $|0\rangle$ until its final gate V_L is applied. ◀

2.2.3 Completeness

With all observations and lemmas of Section 2.2.2 in hand, we are ready to prove completeness of the construction. Specifically, in the YES case, there exists an input $y \in \{0,1\}^{|A|}$ of Hamming weight at most g accepted with probability at least 2/3 by V. The honest prover proceeds as follows.

- (Prepare classical proof) Prepare state (up to global phase) $|\psi_0\rangle \coloneqq |y\rangle_A|0\rangle_B|\widetilde{|A|+1}\rangle_C|\widetilde{1}\rangle_D$ as follows. Starting with $|\phi\rangle = |0\cdots 0\rangle_{AB}|\widetilde{1}\rangle_C|\widetilde{1}\rangle_D$:
  1. Set $j = 1$.
  2. If $y_j = 1$ then
     - Apply, in order, unitaries $\exp(i(\pi/2)F_1)$, $\exp(i(\pi/2)F_2)$,…, $\exp(i(\pi/2)F_{|D|-1})$. This maps registers C and D to 1 and $|D|$, respectively.
     - Apply $\exp(i(\pi/2)P_j)$, which maps A_j from 0 to 1.
     - Apply, in order, unitaries $\exp(i(\pi/2)F_{|D|})$, $\exp(i(\pi/2)F_{|D|-1})$,…, $\exp(i(\pi/2)F_1)$. This maps registers C and D back to 1 and 1, respectively.
  3. Apply unitary $\exp(i(\pi/2)G_j)$, which maps C from j to $j+1$.
  4. Set $j = j+1$.
  5. If $j < |A|+1$, return to line 2 above.

  This process applies $g(2|D|-1)+|A|$ gates.
- (Simulate verifier) Prepare the sequence of states $|\psi_j\rangle = e^{i\frac{\pi}{2}Q_j}\cdots e^{i\frac{\pi}{2}Q_1}|\psi_0\rangle$ by applying, in order, unitaries $\exp(i(\pi/2)Q_1)$, $\exp(i(\pi/2)Q_2)$,…, $\exp(i(\pi/2)Q_L)$. Since the jth step of this process applies $\exp(i(\pi/2)Q_j)$, and since the state $|\psi_0\rangle$ has clock C set to $|A|+1$, Observation 12 and Equation (7) imply that

$$e^{i\frac{\pi}{2}Q_j}|\psi_{j-1}\rangle = \left((V_j)_{R_j} \otimes |\widetilde{|A|+j}+1\rangle\langle\widetilde{|A|+j}|_C\right)|\psi_{j-1}\rangle, \tag{17}$$

  i.e. we increment the clock from $|A|+j$ to $|A|+j+1$ and apply the jth gate V_j. The final state obtained is thus $|\psi_L\rangle = (V_L\cdots V_1|y\rangle_A|0\rangle_B)\otimes|\widetilde{|A|+L+1}\rangle_C|\widetilde{1}\rangle_D$. This process applies L gates.

Since V accepts y with probability at least 2/3, we conclude $\langle\psi_L|M|\psi_L\rangle \leq 1/3$, as desired. The number of Hamiltonians from S_{FGPQ} we needed to simulate in this case is $m = g(2|D|-1)+|A|+L$, as desired.

2.2.4 Soundness

We next show soundness. Specifically, in the NO case, for all inputs $y \in \{0,1\}^{|A|}$ of Hamming weight at most g', V accepts with probability at most 1/3. So, consider any sequence of $m' = g'(2|D|-1)+|A|+L$ Hamiltonian evolutions producing state $|\phi_{m'}\rangle \coloneqq \Pi_{t=1}^{m'} e^{i\theta_u H_u}|0\cdots 0\rangle_{AB}|\widetilde{1}\rangle_C|\widetilde{1}\rangle_D$ for arbitrary $\theta_u \in \mathbb{R}$ and Hamiltonians $H_u \in S_{FGPQ}$. Lemma 15 says we may write

$$|\phi_{m'}\rangle = \sum_{y\in\{0,1\}^{|A|}} \sum_{s=1}^{|C|} \sum_{t=1}^{|D|} \alpha_{y,s,t} |\eta_{y,s,t}\rangle \in \mathrm{Span}(S) \tag{18}$$

with $\sum_{y,s,t} |\alpha_{y,s,t}|^2 = 1$. Now, for the observable (8) it follows that

$$\langle\phi_{m'}|M|\phi_{m'}\rangle = 1 - \langle\phi_{m'}| \left(|1\rangle\langle 1|_{B_1} \otimes |1\rangle\langle 1|_{C_{|C|}}\right) |\phi_{m'}\rangle = 1 - \langle\eta||1\rangle\langle 1|_{B_1}|\eta\rangle \text{ for} \tag{19}$$

$$|\eta\rangle \coloneqq \sum_{y\in\{0,1\}^{|A|}} \sum_{t=1}^{|D|} \alpha_{y,|A|+L+1,t} V_L \cdots V_1 |y\rangle_A |0\rangle_B ||A|\widetilde{+L}+1\rangle_C |\widetilde{t}\rangle_D, \tag{20}$$

where we have used Equation (18) and the fact that M projects onto time step $|C|$ in register C. Now, if we applied strictly less than L evolutions from Q, Lemma 17 says we have no weight on time step $|C|$, so that $\langle\phi_{m'}|M|\phi_{m'}\rangle = 1 \geq 2/3$, as required in the NO case. If, on the other hand, we applied at least L evolutions from Q, then we must have applied at most $g'(2\,|D|-1)+|A|$ evolutions from $F\cup G\cup P$ (otherwise, we have a contradiction since $m' = g'(2\,|D|-1)+|A|+L$). Lemma 16 hence implies the right hand side of Equation (19) equals $1-\langle\eta_{g'}||1\rangle\langle 1|_{B_1}|\eta_{g'}\rangle$ for[9]

$$|\eta_{g'}\rangle \coloneqq \sum_{y \text{ s.t. } \mathrm{HW}(y)\leq g'} \sum_{t=1}^{|D|} \alpha_{y,|A|+L+1,t} V_L \cdots V_1 |y\rangle_A |0\rangle_B ||A|\widetilde{+L}+1\rangle_C |\widetilde{t}\rangle_D, \tag{21}$$

where $\mathrm{HW}(y)$ denotes the Hamming weight of the bitstring y. But since any input y of Hamming weight at most g' is accepted with probability at most $1/3$, we conclude $\langle\phi_{m'}|M|\phi_{m'}\rangle \geq 2/3$, as claimed.

2.2.5 Hardness ratio

Finally, we show our reduction has the desired approximation ratio. Observe

$$\frac{m'}{m} = \frac{g'(2\,|D|-1)+|A|+L}{g(2\,|D|-1)+|A|+L} = \frac{g'(2\lceil L^{1+\delta}\rceil-1)+|A|+L}{g(2\lceil L^{1+\delta}\rceil-1)+|A|+L}. \tag{22}$$

Since $|A|\leq L$ by definition, and since we will choose $\delta>0$ as a small constant, this ratio scales asymptotically as g'/g. Recall now that Theorem 8 says that for any constant $\epsilon'>0$, the QMSA instance $\Pi' = (V', g, g')$ we are reducing from is QCMA-hard to approximate within $g'/g \in O((N')^{1-\epsilon'})$, for N' the encoding size of Π'. By appropriately comparing N' to the encoding size N of our MIN-VQA instance Π, one can in fact show that for any $\epsilon>0$, $g'/g \geq N^{1-\epsilon}$ for large enough V', as desired. The proof is in the full version. ◀

3 Extension of the hardness results to QAOAs

In this section, we prove Theorem 4, which is restated for convenience shortly. First, we define the optimization problem MIN-QAOA covered by the theorem.

A k-local Hamiltonian is a sum of strictly k-local terms, i.e. Hermitian operators each of which acts non-trivially on at most k qubtis. As mentioned previously, our definition of MIN-QAOA is more general than that of [14], and closer to that of [23].

[9] Below, $\mathrm{HW}(y)$ denotes the Hamming weight of string y.

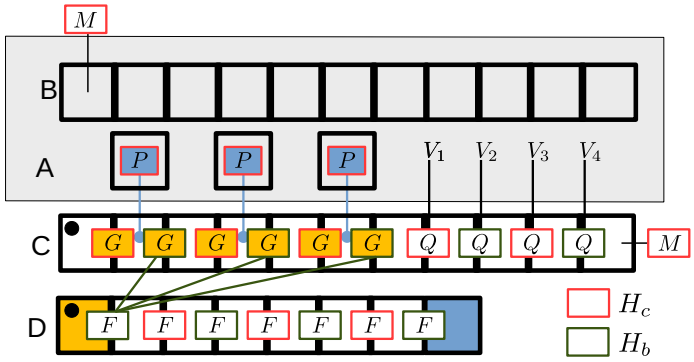


Figure 2 Figure describing the QAOA instance (see Figure 1 for further details). The border color of each gate indicates if the generator belongs to H_b or H_c. Compared to the previous VQA instance, the P now only act at even time steps in C and the even-indexed G_j and the F_1 generator are combined into one generator, denoted by the red and dark green edges.

▶ **Problem 3** (QAOA minimization (MIN-QAOA(k))). *For an n-qubit system:*

- *Input:*
  1. *A set $H = \{H_b, H_c\}$ of k-local Hamiltonians.*
  2. *A $\mathrm{poly}(n)$-size quantum circuit U_b preparing the ground state of H_b, denoted $|\mathrm{gs}_b\rangle$.*
  3. *Integers $0 \leq m \leq m'$ representing thresholds for depth.*
- *Output:*
  1. *YES if there exists a sequence of angles[10] $(\theta_i)_{i=1}^{m} \in \mathbb{R}^m$, such that*

$$|\psi\rangle := e^{i\theta_m H_b} e^{i\theta_{m-1} H_c} \cdots e^{i\theta_2 H_b} e^{i\theta_1 H_c} |\mathrm{gs}_b\rangle \tag{23}$$

  satisfies $\langle\psi|H_c|\psi\rangle \leq \frac{1}{3}$.
  2. *NO if for all sequences of angles $(\theta_i)_{i=1}^{m'} \in \mathbb{R}^{m'}$*

$$|\psi\rangle := e^{i\theta_{m'} H_b} e^{i\theta_{m'-1} H_c} \cdots e^{i\theta_2 H_b} e^{i\theta_1 H_c} |\mathrm{gs}_b\rangle, \tag{24}$$

  satisfies $\langle\psi|H_c|\psi\rangle \geq \frac{2}{3}$.

Just as for MIN-VQA, by "optimal depth" of a QAOA, we mean the minimum number of Hamiltonian evolutions m required above. The expectation value thresholds $\frac{1}{3}$ and $\frac{2}{3}$ are arbitrary and can be changed by rescaling and shifting H_c.

▶ **Theorem 4.** MIN-QAOA(k) *is* QCMA*-complete for $k \geq 4$ and $m \leq \mathrm{poly}(n)$. Moreover, for any $\epsilon > 0$, it is* QCMA*-hard to distinguish between the YES and NO cases of* MIN-QAOA *even if $m'/m \geq N^{1-\epsilon}$, for N the number of strictly k-local terms comprising H_b and H_c.*

Proof. Containment in QCMA is again straightforward and thus omitted. For QCMA-hardness of approximation, we again use a reduction from an instance $\Pi = (V, g, g')$ of QMSA with $V = V_L \cdots V_1$ being a sequence of L two-qubit gates taking in n_V input bits and m_V ancilla qubits. All those terms are defined as in the proof of Theorem 1.

[10] Throughout Problem 3, for clarity we assume all angles are specified to $\mathrm{poly}(n)$ bits.

Proof organization. The proof is organized as follows. In Section 3.1 we explain the modifications of the VQA instances to obtain the QAOA instances of our construction. Section 3.1.2 and Section 3.1.3 explain how we recover the desired initial state and cost function. Section 3.1.4 provides notation preliminary technical results needed for the QCMA-completeness proof. Then, completeness is shown in Section 3.1.5 and soundness in Section 3.1.6. Finally, in Section 3.1.7, we analyze the hardness ratio achieved by the reduction. All omitted proofs are in the full version.

3.1 QCMA completeness for QAOAs

To specify our QAOA instance, we modify the set S_{FGPQ} from the proof of Theorem 1 to suit our reduction here as follows. The structural changes are illustrated in Figure 2. Briefly recapping the proof techniques outline in Section 1.3, we:

- **(i)** implement the reduction with only two generators by alternating even and odd steps of the honest prover's actions, so that H_b simulates the even steps, and H_c the odd ones,
- **(ii)** introduce terms G_j from Equation (27) with 3-cyclic behavior, i.e. allowing three logical actions,
- **(iii)** add new constraints to H_b to ensure its unique ground state encodes the correct start state (see Equation (23) of Problem 3), and
- **(iv)** add the observable O to H_c (scaled larger than other terms in H_c) to obtain the correct cost function.

An undesired side effect of this is that evolution by H_c allows one to leave the desired logical computation space, S. We will show via Lemma 20 that the states obtained are still close to the set, which suffices for our soundess analysis.

To begin, we use registers composed of $|A| = n_V$, $|B| = m_V$, $|C| = L + 2n_V + 1$, and $|D| = \lceil L^{1+\delta} \rceil$ qubits, respectively, where $0 < \delta < 1$ is fixed by specified later. Without loss of generality, we assume $|D|$ and L to be even. Additionally to the changes we outline, we also add diagonal terms additional diagonal terms. This will be relevant for defining the initial state later on.

- (F) We remove F_1,

$$F_j \coloneqq |01\rangle\langle 10|_{D_{j,j+1}} + |10\rangle\langle 01|_{D_{j,j+1}} - 2|00\rangle\langle 00|_{D_{j,j+1}} \text{ for all } j \in \{2, \dots, |D| - 1\}. \quad (25)$$

- (G) We double the number of qubits G acts on,

$$G_j \coloneqq \left(|01\rangle\langle 10|_{C_{j,j+1}} + |10\rangle\langle 01|_{C_{j,j+1}}\right) \otimes |1\rangle\langle 1|_{D_1} - 2|001\rangle\langle 001|_{C_{j,j+1},D_1} \quad \text{for all } j \in \{1, 3, \dots, 2|A| - 1\}, \quad (26)$$

$$G_j \coloneqq \frac{i}{\sqrt{3}}\big(|0110\rangle\langle 1010| + |1001\rangle\langle 0110| + |1010\rangle\langle 1001| - |1010\rangle\langle 0110| - |0110\rangle\langle 1001| - |1001\rangle\langle 1010|\big)_{C_{j,j+1},D_{1,2}} - 2|0010\rangle\langle 0010|_{C_{j,j+1},D_{1,2}} \quad \text{for all } j \in \{2, 4, \dots, 2|A|\}. \quad (27)$$

While odd numbered gates can only change the clock, even numbered ones can still increment C, but also have the option of moving $|\widetilde{1}\rangle_D \to |\widetilde{2}\rangle_D$, which is the operation performed by $F_1^{(o)}$ in the proof of Theorem 1 on MIN-VQA. The superscript (o) refers to the gates of the previous VQA proof. The following relations hold:

$$e^{i\frac{\pi}{3}G_i}|\widetilde{i},\widetilde{1}\rangle_{C,D} = e^{i\frac{\pi}{2}G_i^{(o)}}|\widetilde{i},\widetilde{1}\rangle_{C,D} \propto |\widetilde{i+1},\widetilde{1}\rangle_{C,D}, \quad (28)$$

$$e^{i\frac{2\pi}{3}G_i}|\widetilde{i},\widetilde{1}\rangle_{C,D} = e^{i\frac{\pi}{2}F_1^{(o)}}|\widetilde{i},\widetilde{1}\rangle_{C,D} \propto |\widetilde{i},\widetilde{2}\rangle_{C,D}, \quad (29)$$

where, in this case, "$\propto$" means equality up to a phase.

- (P) For each qubit $j \in \{1, \ldots, |A|\}$ of A, we define the X-operators, but now they only act on even values in the clock register,

$$P_j \coloneqq X_{A_j} \otimes |1\rangle\langle 1|_{C_{2j}} \otimes |1\rangle\langle 1|_{D_{|D|}} - 2|00\rangle\langle 00|_{C_{2j}, D_{|D|}} \text{ for all } j \in \{1, \ldots, |A|\}. \tag{30}$$

- (Q) We shift the C-indices of the Q-gates because reading in the proof takes longer time now,

$$Q_k \coloneqq (V_k)_{R_k} \otimes |01\rangle\langle 10|_{C_{2|A|+k, 2|A|+k+1}} + (V_k^\dagger)_{R_k} \otimes |10\rangle\langle 01|_{C_{2|A|+k, 2|A|+k+1}} \tag{31}$$

$$- 2|00\rangle\langle 00|_{C_{2|A|+k, 2|A|+k+1}} \text{ for all } k \in \{1, \ldots, L\} \tag{32}$$

- (M), (H_0) We add the two operators

$$H_0 = -\left(\sum_{i \in [|A|]} |0\rangle\langle 0|_{A_i} + \sum_{i \in [|B|]} |0\rangle\langle 0|_{B_i} \right) \otimes |1\rangle\langle 1|_{C_1}, \tag{33}$$

$$M = I - |1\rangle\langle 1|_{B_1} \otimes |1\rangle\langle 1|_{C_{|C|}} \tag{34}$$

to the set of generators.

To construct our desired QAOA instance, we define a partition of all gates into two groups:

$$\mathcal{G}_1 = \{G_i\}_{i \in \{2,4,\ldots,2|A|\}} \cup \{F_i\}_{i \in \{3,5,\ldots,|D|-1\}} \cup \{Q_i\}_{i \in \{2,4,\ldots,L\}}, \tag{35}$$

$$\mathcal{G}_2 = \{G_i\}_{i \in \{1,3,\ldots,2|A|-1\}} \cup \{F_i\}_{i \in \{2,4,\ldots,|D|-2\}} \cup \{Q_i\}_{i \in \{1,3,\ldots,L-1\}} \cup \{P_i\}_{i \in [|A|]}. \tag{36}$$

Intuitively, $\mathcal{G}_1$ (respectively, $\mathcal{G}_2$) will be part of our Hamiltonian H_b (respectively, H_c). Note also that all operators in $\mathcal{G}_1$ (respectively, $\mathcal{G}_2$) pairwise commute, a fact we will use in our analysis. Finally, in addition to Assumption 9 and Assumption 10 from the VQA section, we shall also use the following.

▶ **Assumption 18.** *The acceptance probability of V in the YES (respectively, NO) case is at least $1 - \epsilon_Q$ (respectively, at most ϵ_Q), where $\epsilon_Q = O(N^{-1})$. This is achieved via standard parallel k times repetition of the circuit V, followed by a majority vote. This increases the encoding size of V – for k repetitions, the new gate sequence length scales with $L' = k(L + O(1))$, and yields $\epsilon'_Q = \epsilon_Q^{O(k)}$. For the precision we require, it suffices to set $k = O(\log(N))$.*

Due to this assumption, our encoding size increases by a multiplicative log factor, which does not affect our final approximation ratio calculation.

3.1.1 The Min-QAOA instance

The QAOA instance we use to prove Theorem 4 takes the generators

$$H_b = \sum_{\Gamma \in \mathcal{G}_1} \Gamma + H_0\,, \tag{37}$$

$$H_c = \kappa \sum_{\Gamma \in \mathcal{G}_2} \Gamma + M \tag{38}$$

with $m = g(2\,|D| - 2) + |C| - 1$ and $m' = g'(2\,|D| - 4) + |C| - 1$. Crucially, the generators/-operators comprising H_b (respectively, H_c) pairwise commute. The Q gates are taken from a QMSA circuit where using Assumption 18, we set the acceptance threshold of the circuit to $\sqrt{\epsilon_Q} = \frac{1}{48m'}$. Also, we set $\kappa = \frac{1}{24|\mathcal{G}|}$.

We first characterize the initial state and cost function as defined in Problem 3.

3.1.2 Initial state

Recall that in Problem 3 the initial state has to be a ground state of H_b (given as input via a preparation circuit U_b). We want this initial state to be

$$|\mathrm{gs}_b\rangle = |0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD}, \tag{39}$$

which can trivially be prepared by a constant-sized circuit U_b. To see that we indeed obtain this ground state, note below that for all generators except G_1, M, which are not included in H_b, $|\mathrm{gs}_b\rangle$ is a ground state of the generator. Moreover, the groundstate turns out to be unique because for each qubit, the state is uniquely determined by one of the generators, which implies that the entire state is unique. Specifically, we have

$$\begin{aligned}
F_i|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} &= -2|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} \quad \forall i \in \{3,5,\dots,|D|-1\}, \quad &\|F_i\|_\infty &= 2,\\
G_i|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} &= -2|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} \quad \forall i \in \{2,4\dots,2\,|A|\}, &\|G_i\|_\infty &= 2,\\
Q_i|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} &= -2|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} \quad \forall i \in \{2,4\dots,L\}, &\|Q_i\|_\infty &= 2,\\
H_0|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD} &= -(|A|+|B|)|0,0,\widetilde{1},\widetilde{1}\rangle_{ABCD}, &\|H\|_\infty &= |A|+|B|\,.
\end{aligned}$$

Indeed, since the state (39) is a ground state of all the generators of H_b and the terms of H_b mutually commute, it is also a ground state of H_b. Moreover, since every qubit is non-trivially supported by at least one generator of H_b, it is also the unique ground state for the entire Hilbert space, i.e., $|\mathrm{gs}_b\rangle$ represents the unique one-dimensional subspaces where each gate acts non-trivially.

3.1.3 Cost function

In the QAOA setup, the measured observable is H_c. For our construction we wish to use the observable M. Fortunately, we can find an upper bound for the difference of these operators. Namely, for every $|\Psi\rangle \in \mathcal{H}$

$$|\langle\Psi|(H_c - M)|\Psi\rangle| = \kappa|\langle\Psi|\textstyle\sum_{\Gamma\in\mathcal{G}_2}\Gamma|\Psi\rangle| \leq 2\kappa|\mathcal{G}_2| \leq \frac{1}{12} \tag{40}$$

where we used (1) that $\|g\|_\infty \leq 2$ for all $\Gamma \in \mathcal{G}_2$, and (2) the definition of κ.

3.1.4 Preliminaries for the completeness proof

We first define the set of states comprising our logical computation space,

$$S \coloneqq \{V_{t-2|A|-1}\cdots V_1|y\rangle_A|0\cdots 0\rangle_B|\widetilde{t}\rangle_C|\widetilde{s}\rangle_D \,|\, \forall (y,t,s) \in I_S\} \tag{41}$$

with

$$I_S = \left\{(y,t,s)\middle| y\in\{0,1\}^{|A|}, t\in\{1,\dots,|C|\}, s\in\begin{cases}\{1,\dots,|D|\} & \text{if } t\in\{2,4,\cdots 2\,|A|\}\\ \{1\} & \text{otherwise}\end{cases}\right\}$$

being the allowed index set. Here, the notation means that V_1 is only applied if $t > 2\,|A|+1$. Below, we often write a state $|\Psi_S\rangle \in \mathrm{span}(S)$ as

$$|\Psi_S\rangle = \sum_{(y,t,s)\in I_S} a_{y,t,s} V_{t-2|A|-1}\cdots V_1|y\rangle_A|0\cdots 0\rangle_B|\widetilde{t}\rangle_C|\widetilde{s}\rangle_D =: \sum_{(y,t,s)\in I_S} a_{y,t,s}|\Psi_{y,t,s}\rangle\,.$$

We also define the function W, which is intended to capture a lower bound on the number of Hamiltonian evolutions required to prepare a given logical state $|\Psi_{y,t,s}\rangle$:

$$W(y,t,s) \coloneqq (2\,|D|-4)\mathrm{HW}(y) + t + (-1)^{\delta_{y_{\lceil t/2\rceil},1}}(s+\delta_{s,1}-2), \tag{42}$$

where $\mathrm{HW}(y)$ denotes the Hamming weight of y.

We next show a helpful lemma regarding the action of each Hamiltonian on our logical computation space, S.

▶ **Lemma 19.** *The following two statements hold:*

- *For every $|\Psi_{y,t,s}\rangle \in S$ and $H_i \in \{H_b, H_c\}$, $e^{\mathrm{i}H_i\theta}|\Psi_{y,t,s}\rangle = e^{\mathrm{i}\alpha^{(i)}_{y,t,s}\theta}e^{\mathrm{i}\Gamma^{(i)}_{y,t,s}\theta}|\Psi_{y,t,s}\rangle$ for some phase $\alpha \in \mathbb{R}$. In words, applying H_i simulates application of a single gate $\Gamma^{(b)}_{y,t,s} \in \mathcal{G}_1$, $\Gamma^{(c)}_{y,t,s} \in \kappa\mathcal{G}_2 \cup \{M\}$ up to global phase $\alpha_{y,t,s}$, where $\kappa\mathcal{G}_2 = \{\kappa\Gamma \mid \Gamma \in \mathcal{G}_2\}$.*
- *For every $|\Psi_{y,t,s}\rangle \in S$ and every gate $\Gamma \in \mathcal{G}_1 \cup \mathcal{G}_2 \cup \{H_0\}$, $\exists$ amplitudes $\{a_{y,t,s}\}$ such that*

$$e^{\mathrm{i}\Gamma\theta}|\Psi_{y,t,s}\rangle = \sum_{\substack{(y',t',s')\in I_S \\ W(y',t',s')\le W(y,t,s)+1}} a_{y,t,s}|\Psi_{y',t',s'}\rangle \tag{43}$$

In words, the application of Γ can only increase value of the W-function by at most 1.

3.1.5 Completeness

In the YES case, there exists a sequence of gates with proof $y \in \{0,1\}^{|A|}$ of Hamming weight at most g accepted with probability at least $1-\epsilon_Q$ by the verifier circuit V. We use shorthand notation $(y)_j = (y_1, \ldots y_{j-1}, 0, \ldots, 0)$ to indicate the partially written proof string. Also, $\exp(\mathrm{i}\theta H_i) \sim \exp(\mathrm{i}\theta\Gamma)$ indicates which generator Γ in H_i performs the non-trivial operation (as per Lemma 19, claim 1). The honest prover proceeds as follows:

- (Prepare classical proof) Prepare state (up to global phase) $|\psi_0\rangle \coloneqq |y\rangle_A|0\rangle_B|\widetilde{2\,|A|+1}\rangle_C|\widetilde{1}\rangle_D$ as follows. Starting with $|\mathrm{gs}_b\rangle = |(y)_0, 0, \widetilde{1}, \widetilde{1}\rangle_{ABCD}$:
  1. Set $j = 1$.
  2. Apply $\exp(\mathrm{i}\frac{\pi}{2\kappa}H_c) \sim \exp(\mathrm{i}\frac{\pi}{2}G_{2j-1})$ to map $|\widetilde{2j-1}\rangle_C \to |\widetilde{2j}\rangle_C$. This maps
  $$|(y)_{j-1}, 0, \widetilde{2j-1}, \widetilde{1}\rangle_{ABCD} \quad\mapsto\quad |(y)_{j-1}, 0, \widetilde{2j}, \widetilde{1}\rangle_{ABCD}. \tag{44}$$
  3. If $y_j = 1$ then
     - Apply $\exp(\mathrm{i}\frac{2\pi}{3}H_b) \sim \exp(\mathrm{i}\frac{2\pi}{3}G_{2j})$, to map $|\widetilde{1}\rangle_D \to |\widetilde{2}\rangle_D$, i.e.
     $$|(y)_{j-1}, 0, \widetilde{2j}, \widetilde{1}\rangle_{ABCD} \quad\mapsto\quad |(y)_{j-1}, 0, \widetilde{2j}, \widetilde{2}\rangle_{ABCD}. \tag{45}$$
     - Apply, in order, $\exp(\mathrm{i}\frac{\pi}{2\kappa}H_c) \sim \exp(\mathrm{i}\frac{\pi}{2}F_2)$, $\exp(\mathrm{i}\frac{\pi}{2}H_b) \sim \exp(\mathrm{i}\frac{\pi}{2}F_3)$, $\ldots$, $\exp(\mathrm{i}\frac{\pi}{2}H_b) \sim \exp(\mathrm{i}\frac{\pi}{2}F_{|D|-1})$, in total $|D|-2$ operations . This maps $|\widetilde{2}\rangle_D \to ||\widetilde{D}|\rangle_D$, i.e.
     $$|(y)_{j-1}, 0, \widetilde{2j}, \widetilde{2}\rangle_{ABCD} \quad\mapsto\quad |(y)_{j-1}, 0, \widetilde{2j}, |\widetilde{D}|\rangle_{ABCD}. \tag{46}$$
     - Apply $\exp(\mathrm{i}\frac{\pi}{2\kappa}H_c) \sim \exp(\mathrm{i}\frac{\pi}{2}P_j)$, to map $|0\rangle_{A_j} \to |1\rangle_{A_j}$, i.e.
     $$|(y)_{j-1}, 0, \widetilde{2j}, |\widetilde{D}|\rangle_{ABCD} \quad\to\quad |(y)_{j}, 0, \widetilde{2j}, |\widetilde{D}|\rangle_{ABCD}. \tag{47}$$
     - Apply, in order, $\exp(\mathrm{i}\frac{\pi}{2}H_b) \sim \exp(\mathrm{i}\frac{\pi}{2}F_{|D|-1})$,$\exp(\mathrm{i}\frac{\pi}{2\kappa}H_c) \sim \exp(\mathrm{i}\frac{\pi}{2}F_{|D|-2})$ $\ldots$, $\exp(\mathrm{i}\frac{\pi}{2\kappa}H_c) \sim \exp(\mathrm{i}\frac{\pi}{2}F_2)$, in total $|D|-2$ operations. This maps $||\widetilde{D}|\rangle_D \to |\widetilde{2}\rangle_D$, i.e.
     $$|(y)_{j}, 0, \widetilde{2j}, |\widetilde{D}|\rangle_{ABCD} \quad\mapsto\quad |(y)_{j}, 0, \widetilde{2j}, \widetilde{2}\rangle_{ABCD}. \tag{48}$$

- Apply $\exp(\mathrm{i}\frac{2\pi}{3}H_b) \sim \exp(\mathrm{i}\frac{2\pi}{3}G_{2j})$, to map $|\widetilde{2}\rangle_D \to |\widetilde{1}\rangle_D$ and $|\widetilde{2j}\rangle_C \to |\widetilde{2j+1}\rangle_C$, i.e.

$$|(y)_j, 0, \widetilde{2j}, \widetilde{2}\rangle_{ABCD} \quad \mapsto \quad |(y)_j, 0, \widetilde{2j+1}, \widetilde{1}\rangle_{ABCD} \tag{49}$$

4. else
   - Apply $\exp(\mathrm{i}\frac{\pi}{3}H_b) \sim \exp(\mathrm{i}\frac{\pi}{3}G_{2j})$, to map $|\widetilde{2j}\rangle_C \mapsto |\widetilde{2j+1}\rangle_C$, i.e.

$$|(y)_{j-1}, 0, \widetilde{2j}, \widetilde{1}\rangle_{ABCD} \quad \mapsto \quad |(y)_j, 0, \widetilde{2j+1}, \widetilde{1}\rangle_{ABCD}. \tag{50}$$

5. Set $j = j + 1$.
6. If $j < |A|$, return to line 2 above.

This process applies $2g(|D| - 1) + 2|A|$ gates.

- (Simulate verifier) Apply in order, $\exp(\mathrm{i}\frac{\pi}{2\kappa}H_c) \sim \exp(\mathrm{i}\frac{\pi}{2}Q_1)$, $\exp(\mathrm{i}\frac{\pi}{2}H_b) \sim \exp(\mathrm{i}\frac{\pi}{2}Q_2)$, $\ldots$, $\exp(\mathrm{i}\frac{\pi}{2}H_b) \sim \exp(\mathrm{i}\frac{\pi}{2}Q_L)$ for a total L gates. This implements the verifier, i.e.

$$|y, 0, 2\widetilde{|A|+1}, \widetilde{1}\rangle_{ABCD} \quad \to \quad |\Psi_L\rangle \coloneqq V_L \cdots V_1 |y, 0, \widetilde{|C|}, \widetilde{1}\rangle_{ABCD}. \tag{51}$$

Since V accepts proof y of the QMSA instance with probability at least $1 - \epsilon_Q$, we conclude using Equation (40) that

$$\langle \Psi_L | H_c | \psi_L \rangle \leq \langle \Psi_L | M | \Psi_L \rangle + \frac{1}{12} \leq 1 - (1 - \epsilon_Q) + \frac{1}{12} \leq \frac{1}{3} \tag{52}$$

as desired. The number of Hamiltonians applied in this case is $m = g(2|D| - 2) + 2|A| + L = g(2|D| - 2) + |C| - 1$, also as desired.

3.1.6 Soundness

In the proof of Theorem 1 for MIN-VQA, we showed that all Hamiltonian evolutions keep us in our desired logical computation space S. In contrast, for our MIN-QAOA construction, the M operator (embedded in H_c) does *not* necessarily preserve the space $\mathrm{span}(S)$ (see Claim 2 of Lemma 19). We thus first require the following lemma, which allows us to "round" our intermediate state back to one in S for our analysis and also establishes $W(y, t, s)$ as a proper lower bound for the number of gate applications required to reach the states in S.

▶ **Lemma 20** (Rounding lemma). *In the NO case, after $\zeta \geq 1$ applications of H_c and H_b, $|\Psi_\zeta\rangle \in \Gamma_\zeta \coloneqq \left\{ \prod_{i=1}^{\zeta} e^{iH_i\theta_i} |\mathrm{gs}_b\rangle \mid H_i \in \{H_b, H_c\}, \theta \in \mathbb{R}^\zeta \right\}$ will be $\epsilon \leq 4\zeta\sqrt{\epsilon_Q}$ close to the span of S i.e.*

$$\forall |\Psi_\zeta\rangle \in \Gamma_\zeta,\ \exists |\Psi'_\zeta\rangle \in \mathrm{span}(S) : \left\| |\Psi_\zeta\rangle\langle\Psi_\zeta| - |\Psi'_\zeta\rangle\langle\Psi'_\zeta| \right\|_{\mathrm{tr}} \leq 4\zeta\sqrt{\epsilon_Q} \tag{53}$$

and it additionally holds that $|\Psi'_\zeta\rangle = \sum_{\substack{(y,t,s)\in I_S \\ W(y,t,s)\leq\zeta+1}} a_{y,t,s} |\Psi_{y,t,s}\rangle$.

The proof is in the full version. This lemma is needed because the time evolution of the observable M (in H_c) may leave the sub-space $\mathrm{Span}(S)$. The rounding step is possible, because in the NO case, the state in the B_1 register, after applying the circuit V ($\tilde{s} = |D|$), is always close to $|0\rangle_{B_1}$ (using Assumption 18), meaning the evolution in M only adds to a global phase.

We are finally ready to prove soundness. For this, we need to show that in the NO case, all sequences of $\zeta \leq m' = g'(2\,|D| - 4) + |C| - 1$ gates produce cost function value $\langle \Psi_\zeta | H_c | \Psi_\zeta \rangle \geq \frac{2}{3}$. This follows since for all $\zeta \leq m'$,

$$\langle \Psi_\zeta | H_c | \Psi_\zeta \rangle \geq \langle \Psi_\zeta | M | \Psi_\zeta \rangle - \frac{1}{12} \tag{54}$$

$$\geq \langle \Psi'_\zeta | M | \Psi'_\zeta \rangle - |\mathrm{Tr}[M(|\Psi_\zeta\rangle\langle\Psi_\zeta| - |\Psi'_\zeta\rangle\langle\Psi'_\zeta|)]| - \frac{1}{12} \tag{55}$$

$$\geq \langle \Psi'_\zeta | M | \Psi'_\zeta \rangle - \|M\|_\infty \left\| |\Psi_\zeta\rangle\langle\Psi_\zeta| - |\Psi'_\zeta\rangle\langle\Psi'_\zeta| \right\|_{\mathrm{tr}} - \frac{1}{12} \tag{56}$$

$$\geq \langle \Psi'_\zeta | M | \Psi'_\zeta \rangle - 4m' \sqrt{\epsilon_Q} - \frac{1}{12} \tag{57}$$

$$\geq \langle \Psi'_\zeta | M | \Psi'_\zeta \rangle - \frac{1}{6} \tag{58}$$

where the first statement follows from Equation (40), the third by Hölder's inequality, the fourth by Lemma 20, and the last since $\sqrt{\epsilon_Q} \leq \frac{1}{48m'}$. By Lemma 19, we can expand $|\Psi'_\zeta\rangle$ in the basis $|\Psi'_\zeta\rangle = \sum_{\substack{(y,t,s)\in I_S \\ W(y,t,s)\leq m'+1}} a_{y,t,s} |\Psi_{y,t,s}\rangle$ which gives

$$\langle \Psi'_\zeta | M | \Psi'_\zeta \rangle = 1 - \sum_{y\in\{0,1\}^{|A|} | \mathrm{HW}(y)\leq g'} |a_{y,|C|,1}|^2 \langle \Psi_{y,|C|,1} | |1\rangle\langle 1|_{B_1} | \Psi_{y,|C|,1} \rangle \geq 1 - \epsilon_Q \tag{59}$$

as M only acts non-trivial on $t = |C|$ and $W(y, |C|, 1) \leq m' + 1$ reduces to $\mathrm{HW}(y) \leq g'$, and in the NO case QMSA accepts such a y with at most ϵ_Q probability. Combining the two results we get

$$\langle \Psi_\zeta | H_c | \Psi_\zeta \rangle \geq 1 - \epsilon_Q - \frac{1}{6} > \frac{2}{3} \tag{60}$$

which shows soundness for all gates-sequences of length $\zeta \leq m'$.

3.1.7 Hardness ratio

The analysis is essentially identical to that for MIN-VQA, and is in the full version. ◀

References

1 Dorit Aharonov, Itai Arad, and Thomas Vidick. Guest column: The quantum PCP conjecture. *SIGACT News*, 44(2):47–79, June 2013. doi:10.1145/2491533.2491549.

2 Dorit Aharonov and Tomer Naveh. Quantum NP - a survey, 2002. arXiv:quant-ph/0210077.

3 Eric R. Anschuetz and Bobak T. Kiani. Beyond barren plateaus: Quantum variational algorithms are swamped with traps, 2022. arXiv:2205.05786.

4 Anurag Anshu and Tony Metger. Concentration bounds for quantum states and limitations on the QAOA from polynomial approximations, 2022. arXiv:2209.02715.

5 S. Arora and S. Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. Prelim. version FOCS '92. doi:10.1145/273865.273901.

6 Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. Prelim. version FOCS '92. doi:10.1145/278298.278306.

7 Joao Basso, David Gamarnik, Song Mei, and Leo Zhou. Performance and limitations of the qaoa at constant levels on large sparse hypergraphs and spin glass models, 2022. doi:10.48550/arXiv.2204.10306.

8 Kishor Bharti, Alba Cervera-Lierta, Thi Ha Kyaw, Tobias Haug, Sumner Alperin-Lea, Abhinav Anand, Matthias Degroote, Hermanni Heimonen, Jakob S. Kottmann, Tim Menke, Wai-Keong Mok, Sukin Sim, Leong-Chuan Kwek, and Alán Aspuru-Guzik. Noisy intermediate-scale quantum (NISQ) algorithms. *Rev. Mod. Phys.*, 94(1):015004, January 2022. doi:10.1103/RevModPhys.94.015004.

9 Lennart Bittel and Martin Kliesch. Training variational quantum algorithms is NP-hard. *Phys. Rev. Lett.*, 127:120502, September 2021. doi:10.1103/PhysRevLett.127.120502.

10 Sami Boulebnane and Ashley Montanaro. Solving boolean satisfiability problems with the quantum approximate optimization algorithm, 2022. doi:10.48550/arXiv.2208.06909.

11 Gregory Boyd and Bálint Koczor. Training variational quantum circuits with covar: covariance root finding with classical shadows, 2022. arXiv:2204.08494.

12 Sergey Bravyi, Alexander Kliesch, Robert Koenig, and Eugene Tang. Obstacles to variational quantum optimization from symmetry protection. *Phys. Rev. Lett.*, 125:260505, December 2020. doi:10.1103/PhysRevLett.125.260505.

13 M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and Patrick J. Coles. Variational quantum algorithms. *Nat. Rev. Phys.*, 3:625–644, 2021. doi:10.1038/s42254-021-00348-9.

14 Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014. arXiv:1411.4028.

15 Edward Farhi and Aram W Harrow. Quantum supremacy through the quantum approximate optimization algorithm, 2016. arXiv:1602.07674.

16 Richard P Feynman. Quantum mechanical computers. *Found. Phys.*, 16(6):507–531, 1986. URL: http://www.cs.princeton.edu/courses/archive/fall05/frs119/papers/feynman85_optics_letters.pdf.

17 S. Gharibian and J. Sikora. Ground state connectivity of local Hamiltonians. In *42nd International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 617–628, 2015. doi:10.1007/978-3-662-47672-7_50.

18 Sevag Gharibian and Julia Kempe. Hardness of approximation for quantum problems. In *39th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 387–398, 2012. doi:10.1007/978-3-642-31594-7_33.

19 M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42:1115–1145, 1995. doi:10.1145/227683.227684.

20 David Gosset, Jenish C. Mehta, and Thomas Vidick. QCMA hardness of ground space connectivity for commuting Hamiltonians. *Quantum*, 1:16, July 2017. doi:10.22331/q-2017-07-14-16.

21 Harper R. Grimsley, George S. Barron, Edwin Barnes, Sophia E. Economou, and Nicholas J. Mayhall. ADAPT-VQE is insensitive to rough parameter landscapes and barren plateaus, 2022. arXiv:2204.07179.

22 Harper R. Grimsley, Sophia E. Economou, Edwin Barnes, and Nicholas J. Mayhall. An adaptive variational algorithm for exact molecular simulations on a quantum computer. *Nat. Commun.*, 10:3007, July 2019. doi:10.1038/s41467-019-10988-2.

23 Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G. Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2):34, 2019. doi:https://www.mdpi.com/1999-4893/12/2/34.

24 Hsin-Yuan Huang, Richard Kueng, and John Preskill. Predicting many properties of a quantum system from very few measurements. *Nature Physics*, 16:1050–1057, June 2020. doi:10.1038/s41567-020-0932-7.

25 Alexei Yu Kitaev, Alexander Shen, and Mikhail N Vyalyi. *Classical and quantum computation*, volume 47. American Mathematical Society, 2002. URL: https://bookstore.ams.org/gsm-47.

26 Bálint Koczor and Simon C. Benjamin. Quantum analytic descent. *Phys. Rev. Research*, 4(2):023017, April 2022. doi:10.1103/PhysRevResearch.4.023017.

27 Martin Larocca, Nathan Ju, Diego García-Martín, Patrick J. Coles, and M. Cerezo. Theory of overparametrization in quantum neural networks, 2021. arXiv:2109.11676.

28 Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996. doi:10.1126/science.273.5278.1073.

29 Guang Hao Low and Isaac L. Chuang. Optimal Hamiltonian simulation by quantum signal processing. *Phys. Rev. Lett.*, 118:010501, January 2017. doi:10.1103/PhysRevLett.118.010501.

30 Jarrod R. McClean, Sergio Boixo, Vadim N. Smelyanskiy, Ryan Babbush, and Hartmut Neven. Barren plateaus in quantum neural network training landscapes. *Nat. Commun.*, 9:4812, November 2018. doi:10.1038/s41467-018-07090-4.

31 Javier Rivera-Dean, Patrick Huembeli, Antonio Acín, and Joseph Bowles. Avoiding local minima in variational quantum algorithms with neural networks, 2021. arXiv:2104.02955.

32 Lucas Slattery, Benjamin Villalonga, and Bryan K. Clark. Unitary block optimization for variational quantum algorithms. *Phys. Rev. Research*, 4(2):023072, April 2022. doi:10.1103/PhysRevResearch.4.023072.

33 Bobak Toussi Kiani, Seth Lloyd, and Reevu Maity. Learning unitaries by gradient descent, 2020. arXiv:2001.11897.

34 C. Umans. Hardness of approximating Σ_2^p minimization problems. In *40th Symposium on Foundations of Computer Science*, pages 465–474, 1999.

35 J. D. Watson, J. Bausch, and S. Gharibian. The Complexity of Translationally Invariant Problems beyond Ground State Energies. In *40th Symposium on Theoretical Aspects of Computer Science (STACS 2023)*, 2023.

36 David Wierichs, Christian Gogolin, and Michael Kastoryano. Avoiding local minima in variational quantum eigensolvers with the natural gradient optimizer. *Phys. Rev. Research*, 2(4):043246, November 2020. doi:10.1103/PhysRevResearch.2.043246.

37 Roeland Wiersema, Cunlu Zhou, Yvette de Sereville, Juan Felipe Carrasquilla, Yong Baek Kim, and Henry Yuen. Exploring entanglement and optimization within the Hamiltonian variational ansatz. *PRX Quantum*, 1:020319, 2020. doi:10.1103/PRXQuantum.1.020319.

38 P. Wocjan, D. Janzing, and T. Beth. Two QCMA-complete problems. *Quantum Information & Computation*, 3(6):635–643, 2003. doi:10.5555/2011556.2011563.

39 Dan-Bo Zhang and Tao Yin. Collective optimization for variational quantum eigensolvers. *Phys. Rev. A*, 101(3):032311, March 2020. doi:10.1103/PhysRevA.101.032311.

40 Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D. Lukin. Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices. *Phys. Rev. X*, 10:021067, 2020. doi:10.1103/PhysRevX.10.021067.

An Algorithmic Approach to Uniform Lower Bounds

Rahul Santhanam
Department of Computer Science, University of Oxford, UK

Abstract

We propose a new family of circuit-based sampling tasks, such that non-trivial algorithmic solutions to certain tasks from this family imply frontier uniform lower bounds such as "NP is not in uniform ACC^0" and "NP does not have uniform polynomial-size depth-two threshold circuits". Indeed, the most general versions of our sampling tasks have implications for central open problems such as NP vs P and PSPACE vs P.

We argue the soundness of our approach by showing that the non-trivial algorithmic solutions we require do follow from standard cryptographic assumptions. In addition, we give evidence that a version of our approach for uniform circuits is *necessary* in order to separate NP from P or PSPACE from P. We give an *algorithmic characterization* for the PSPACE vs P question: PSPACE $\neq$ P iff either E has sub-exponential time non-uniform algorithms infinitely often or there are non-trivial space-efficient solutions to our sampling tasks for uniform Boolean circuits.

We show how to use our framework to capture uniform versions of known non-uniform lower bounds, as well as classical uniform lower bounds such as the space hierarchy theorem and Allender's uniform lower bound for the Permanent. We also apply our framework to prove new lower bounds: NP does not have polynomial-size uniform AC^0 circuits with a bottom layer of MOD 6 gates, nor does it have polynomial-size uniform AC^0 circuits with a bottom layer of threshold gates.

Our proofs exploit recently defined probabilistic time-bounded variants of Kolmogorov complexity [36, 24, 34].

2012 ACM Subject Classification Theory of computation → Circuit complexity; Theory of computation → Complexity classes

Keywords and phrases Probabilistic Kolmogorov complexity, sampling algorithms, uniform lower bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.35

Related Version *Extended Version*: https://eccc.weizmann.ac.il/report/2023/028/

Funding This work was partly funded by the ESPRC New Horizons grant EP/V048201/1 on "Structure vs Randomness in Algorithms and Computation".

Acknowledgements We are grateful to Eric Allender, Arkadev Chattopadhyay, Hanlin Ren, Zhenjian Lu, Igor Oliveira, Ronen Shaltiel and Srikanth Srinivasan for useful discussions.

1 Introduction

1.1 Background and Motivation

The NP vs P problem [18, 20] is the central problem in theoretical computer science. Despite much effort over the years, we seem to be quite far from a solution. Theoretical computer science has had many successes over the years, but as far as NP vs P is concerned, it has been hard even to come up with viable approaches to the problem.

When the problem first received attention in the 1970s, a natural approach to it was to explore analogies with computability theory, and use simulation and diagonalization techniques to achieve a separation. For example, the uncomputability of the Halting Problem is a foundational result in computability theory proved using diagonalization. A *time-bounded*

38th Computational Complexity Conference (CCC 2023).
Editor: Amnon Ta-Shma; Article No. 35; pp. 35:1–35:26

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


version of the Halting Problem for non-deterministic machines is NP-complete, so it makes sense to try resource-bounded variants of diagonalization to separate NP and P. However, all approaches using simulation and diagonalization have been fruitless so far, and a major reason for this was identified by Baker, Gill and Solovay [11] in their paper on the relativization barrier. Classical techniques in computability theory *relativize*, meaning that they continue to hold relative to an arbitrary oracle, but no solution to the NP vs P can relativize - there is an oracle A such that $\mathsf{NP} = \mathsf{P}$ relative to A, and another oracle B such that $\mathsf{NP} \neq \mathsf{P}$ relative to B [11].

After the relativization barrier was identified, attention shifted to the *non-uniform* version of the NP vs P problem. The non-uniform version asks if NP has polynomial-size Boolean circuits. A negative answer implies $\mathsf{NP} \neq \mathsf{P}$, since all problems in P have polynomial-size Boolean circuits. The hope in studying Boolean circuits was that they might be easier to analyze and understand using combinatorial and algebraic techniques than is the case for uniform algorithms. Indeed, the hope was fed by a spate of results in the 1980s showing super-polynomial lower bounds against weak circuit classes, including AC^0 circuits [2, 22, 60, 25] , $\mathsf{AC}^0[p]$ circuits for prime p [47, 50] and monotone circuits [46].

This spate of lower bound results slowed down to a trickle in the 1990s, and even the question of proving super-polynomial lower bounds against constant-depth circuits with composite modular gates remained unsolved. In an attempt to explain the stalled progress, Razborov and Rudich [48] identified a further barrier: the Natural Proofs barrier. While the relativization barrier applies to traditional simulation and diagonalization approaches, the Natural Proofs barrier applies to combinatorial and algebraic techniques that were the main source of hope for showing non-uniform lower bounds. The Natural Proofs barrier rules out *constructive* approaches to circuit lower bounds that involve identifying a complexity measure that is easy to compute, low for functions with small circuits and high for random functions, assuming standard cryptographic conjectures. Essentially all known non-uniform lower bound techniques at the time involved identifying such complexity measures, and so the Natural Proofs barrier did help to explain why progress was stalled.

Since the Natural Proofs barrier was identified, there has been pessimism about the prospect of proving lower bounds in the near future, and few promising lower bound approaches have been identified. The ambitious Geometric Complexity Theorey program of Mulmuley and Sohoni [38, 37] seeks to solve the Permanent vs Determinant problem, which is an algebraic analogue of NP vs P, by using representation theory and algebraic geometry to analyze symmetries of the Determinant and Permanent. A version of their approach also applies to the NP vs P problem. While Geometric Complexity Theory has led to significant new insights in algebraic complexity theory, the original approach has also faced some obstacles [14], and Mulmuley himself believes that it is likely to take a very long time for NP vs P to be solved using this approach [20].

A rare success in the theory of lower bounds over the past couple of decades is the algorithmic method [57, 58] of Ryan Williams. Somewhat paradoxically, Williams proposed to attack the question of circuit lower bounds, i.e., proving that no efficient non-uniform algorithms exist for some task, by finding improved algorithms for a different task. To be more specific, if we wish to prove that NEXP does not have polynomial-size $\mathfrak{C}$-circuits for some circuit class $\mathfrak{C}$ with natural closure properties, then all we need to do is to solve the Satisfiability problem for $\mathfrak{C}$-circuits in barely non-trivial time, i.e., in time $2^n\mathsf{poly}(m)/n^{\omega(1)}$, where m is the circuit size and n is the number of variables. Note that the trivial brute force search algorithm for Satisfiability takes time $2^n\mathsf{poly}(m)$, so what is required is just a *super-polynomial* improvement over this trivial algorithm.

One might wonder whether an algorithmic approach via improved algorithms for Satisfiability is feasible if we are interested in lower bounds for general Boolean circuits. Perhaps no non-trivial improvement over brute force search is possible for Circuit Satisfiability? This potential objection is addressed in [57] by showing that circuit lower bounds hold even if we can estimate the acceptance probability of a $\mathfrak{C}$-circuit in barely non-trivial time. This acceptance probability estimation task is known to be doable in sub-exponential time under circuit lower bound assumptions, by using ideas from the theory of derandomization [42, 10]. Thus, in a sense, the algorithmic approach is without loss of generality when it comes to circuit lower bounds for NEXP.

In a breakthrough, Williams showed how to solve a frontier question in circuit lower bounds using the algorithmic method. He showed that $\mathsf{NEXP} \not\subseteq \mathsf{ACC}^0$ [58], by designing algorithms for ACC^0 Satisfiability that beat brute-force search. Since then, several other lower bounds for restricted classes of circuits have been shown using the algorithmic method [59, 56, 7, 15, 16, 40, 17, 13]. What is particularly appealing about the algorithmic method is that humans seem more suited to constructive algorithmic thinking than to proving impossibility results, and so the reformulation of a lower bound task as an algorithmic task is likely to stimulate progress.

The original formulation of the algorithmic method [57] gave a connection between non-trivial Satisfiability algorithms and circuit lower bounds for NEXP. While NEXP lower bounds are interesting from a derandomization perspective, what we desire most in complexity theory is super-polynomial size lower bounds for NP. Murray and Williams [40] show how to scale down the algorithmic method to derive lower bounds for NQP (non-deterministic quasi-polynomial time) against polynomial-size $\mathfrak{C}$-circuits and lower bounds for NP against fixed polynomial size $\mathfrak{C}$-circuits (where the running time of the NP algorithm depends on the size lower bound) from circuit analysis algorithms for $\mathfrak{C}$. However, their techniques do not seem to be useful in deriving super-polynomial size lower bounds for NP - it is unclear what a corresponding circuit analysis task would be.

It seems very challenging even to prove that NP does not have polynomial-size ACC^0 circuits, and no approaches to this question are known. But what if we weaken our goal to lower bounds against polynomial-size *uniform* ACC^0 circuits? Note that from the perspective of making progress toward $\mathsf{NP} \neq \mathsf{P}$, uniform lower bounds are just as good as non-uniform ones. However, we do not have any cases so far of super-polynomial uniform lower bounds for NP against a class $\mathfrak{C}$ of circuits where a corresponding non-uniform lower bound is unknown. We also don't have any plausible approaches toward showing such uniform lower bounds.

This raises the following question, which is at the core of our work.

▶ **Question.** *Is there an algorithmic approach*[1] *to long-standing open questions about uniform lower bounds?*

To make this question more precise, we state a couple of criteria that we require from an "algorithmic approach". First, we would like our algorithmic task to be as close to a conventional algorithmic task as possible - a task that takes an input and produces output that satisfies some desired property. For example, cryptographic pseudo-random generators imply $\mathsf{NP} \neq \mathsf{P}$, but we do not consider the construction of such generators as a standard algorithmic task, just as the construction of *complexity-theoretic* pseudo-random generators

[1] Note that we are interested in this paper only in algorithmic approaches to *lower bounds*. We hold the conventional belief that $\mathsf{NP} \neq \mathsf{P}$ and wish to show this using an algorithmic approach. Of course, if one wishes to show that $\mathsf{NP} = \mathsf{P}$, an algorithmic approach is completely natural, i.e., designing and analyzing a polynomial-time algorithm for SAT.

does not count as an algorithmic approach toward circuit lower bounds for EXP. Second, we would like our approach to be sound, meaning that there should be some evidence that the algorithmic task is indeed feasible. Note that an infeasible algorithmic task, such as designing polynomial-time algorithms for an EXP-complete problem, would imply anything at all, and in particular would imply $\mathsf{NP} \neq \mathsf{P}$.

When trying to design an algorithmic approach to uniform lower bounds, it is useful to keep in mind an informal but fundamental distinction between two regimes of lower bounds - the *complexity-theoretic* regime and the *cryptographic* regime. The complexity-theoretic regime refers to situations where we are trying to show a lower bound $\mathfrak{C} \not\subseteq \mathfrak{D}$ for complexity classes $\mathfrak{C}$ and $\mathfrak{D}$, and $\mathfrak{C}$ has enough computational resources to simulate $\mathfrak{D}$. For example, the case of $\mathsf{NEXP} \not\subseteq \mathsf{SIZE}(\mathsf{poly})$ falls into the complexity-theoretic regime, since exponential-time machines can simulate polynomial-size circuits. The cryptographic regime refers to situations where $\mathfrak{C}$ does not have the resources to simulate $\mathfrak{D}$. The case of super-polynomial size uniform circuit lower bounds for NP falls into the cryptographic regime, since we are trying to show that a *fixed* problem in NP does not have uniform circuits of *arbitrary* polynomial size, and so the NP machine does not have enough resources to simulate the circuits against which a lower bound is sought.

The common feature to all known applications of the algorithmic method [57, 58, 40] is that the corresponding lower bounds fall in the complexity-theoretic regime. The reason is that the proof technique for establishing the connection between algorithms and lower bounds involves indirect diagonalization, culminating in an application of *hierarchy theorems*. Since hierarchy theorems require the class for which we are showing a lower bound to have more resources than the class against which we are showing a lower bound, it seems unlikely that the algorithmic method can be directly adapted to the cryptographic regime of lower bounds.

In this work, we present a new family of circuit-based sampling tasks such that solutions to tasks in this family imply uniform circuit lower bounds in the cryptographic regime for classes such as NP and PSPACE. Since these algorithmic tasks have not been considered before, there isn't a clear path yet to solving them for general Boolean formulas or circuits. As such, this is not yet a full-fledged approach to strong uniform circuit lower bounds such as $\mathsf{NP} \neq \mathsf{P}$ or $\mathsf{PSPACE} \neq \mathsf{P}$. Nevertheless, our approach does allow us to recover state-of-the-art uniform lower bounds and to prove a couple of new ones, and we believe it might be useful to attack frontier uniform lower bound questions such as separating NP from uniform ACC^0 or uniform TC^0_2. Moreover, we believe that the connection from algorithms to lower bounds is interesting in itself, and hope that it will stimulate further research on the sampling tasks we define.

We now proceed to describe our approach.

1.2 The Approach

We describe our approach to lower bounds for PSPACE first. Suppose we seek to prove uniform lower bounds for PSPACE against some class $\mathfrak{C}$ of circuits. The only requirement we will make of $\mathfrak{C}$ is that it can be simulated by general Boolean circuits. In order to describe the algorithmic tasks in our approach, we first introduce some terminology.

Given a $\mathfrak{C}$-circuit C on n variables of size $\mathsf{poly}(n)$, we say that C is *dense* if C accepts at least a $2/3$ fraction of all inputs of length n. We are interested in the problem of sampling satisfying assignments of $\mathfrak{C}$. Satisfying assignments of C are plentiful, so a trivial randomized algorithm works with high probability, but it might not be easy for a deterministic algorithm to find satisfying assignments. In our setting, we will *allow* the use of randomness in the algorithm.

In order not to make the task trivial, we will require that the algorithm outputs a *fixed* satisfying assignment with probability at least $n^k/2^n$ for some large enough constant k. This requirement is related to the notion of pseudo-deterministic search defined by Gat and Goldwasser [23]. A pseudo-deterministic algorithm is a randomized algorithm for a search problem that outputs a fixed solution with high probability, say 2/3. In contrast, we only require that a fixed assignment is output with probability $\mathsf{poly}(n)/2^n$, which is barely non-trivial.

It is not hard to see that this task is easy if the algorithm is given full access to C and is able to simulate C. The key restriction we impose is that the algorithm does not have enough resources to simulate C, though it does have full access to C. The algorithm is given random access to the representation of C, and must run in some fixed polynomial space bound n^d, where d is independent of the size of C. This restriction turns out to be enough to imply lower bounds for PSPACE against uniform $\mathfrak{C}$-circuits.

We say that there is *space-efficient non-trivial sampling for dense $\mathfrak{C}$-satisfiability* if the task described above is feasible, namely if for some large enough constant[2] k, there is a constant d and a probabilistic algorithm A running in space n^d such that, given an input $\mathfrak{C}$-circuit C on n variables of size $\mathsf{poly}(n)$ accepting at least a 2/3 fraction of all inputs, A outputs a fixed satisfying assignment y of C with probability at least $n^k/2^n$. Our main result for PSPACE shows that feasibility of this task implies lower bounds for PSPACE against uniform $\mathfrak{C}$-circuits, where the notion of uniformity is LOGSPACE-uniformity.

▶ **Theorem 1** (Informal Statement). *Let $\mathfrak{C}$ be any class of circuits that can be simulated by Boolean circuits of polynomial size. If there is space-efficient non-trivial sampling for dense $\mathfrak{C}$-satisfiability, then* PSPACE *does not have uniform $\mathfrak{C}$-circuits of polynomial size.*

As a corollary, when $\mathfrak{C}$ is the class of general Boolean circuits, space-efficient non-trivial sampling for dense $\mathfrak{C}-\mathsf{SAT}$ implies that $\mathsf{PSPACE} \neq \mathsf{P}$. Thus a solution to a purely algorithmic task separates PSPACE from P. We find this connection surprising, even if it's not apparent what sorts of algorithmic ideas might be useful in solving our task.

We note that the requirements of our algorithmic task are fairly relaxed in many ways. We are interested in randomized algorithms, while it is still open to derive non-uniform circuit lower bounds for NEXP from randomized algorithms for CircuitSAT. A trivial linear-time randomized algorithm for our task simply samples a random $y \in \{0,1\}^n$ and outputs it - each satisfying assignment is output with probability 2^{-n} by this algorithm. We only require a *fixed polynomial advantage* in sampling probability over this trivial algorithm.

However, the restriction that the algorithm must operate in space a fixed polynomial independent of the circuit size of C is indeed a fairly strong requirement. Essentially, what this restriction means is that we can't simulate the circuit C when trying to solve our algorithmic task. Thus, in order to solve our algorithmic task, we need to have a rich structural understanding of $\mathfrak{C}$-circuits accepting at least a 2/3 fraction of their inputs.

We argue heuristically that some such restriction on white-box access is necessary to derive lower bounds for PSPACE. Consider the case where $\mathfrak{C}$ is just the class of general Boolean circuits. Suppose we were able to derive $\mathsf{PSPACE} \neq \mathsf{P}$ from the success of some algorithmic task T that is defined with a Boolean circuit C as input, and suppose we had full white-box access to C. If T is solvable in PSPACE, then if $\mathsf{PSPACE} = \mathsf{P}$, T should be solvable efficiently. Since the efficient solvability of T implies $\mathsf{NP} \neq \mathsf{P}$, we get that $\mathsf{PSPACE} \neq \mathsf{P}$ unconditionally!

[2] Our proof shows that $k > 3$ suffices, and we suspect that $k > 1$ might suffice if we optimise our parameters.

Of course it is possible that T is not solvable in PSPACE, but rather (say) in EXP. However, in this case, there might not be sufficient reason to believe T is solvable efficiently, and using T to derive a lower bound might not be a sound algorithmic approach.

We would like to emphasize the point made by the argument above: the restriction that the algorithm for our task cannot simulate the circuit C is not just an artifact or weakness of our approach. Rather, it seems unavoidable for any white-box task in the cryptographic regime, since the lower bound we are trying to show is quantitatively stronger than the upper bound.

Indeed, as mentioned before, if we have full white-box access to C, we can unconditionally solve the algorithmic task we propose, by repeatedly sampling strings of length n and outputting the lexicographically smallest string accepted by C. Theorem 31 in Section 4 establishes this formally. The ability of the algorithm to read the description of C as well as to simulate C makes this argument work.

Now suppose we wish to extend the approach of Theorem 1 to uniform lower bounds for NP. We can define an analogous notion of time-efficient non-trivial sampling for dense $\mathfrak{C}$-satisfiability, and prove a connection similar to Theorem 1 where the consequence is a lower bound for NP. However, this notion of time-efficient non-trivial sampling is very restrictive, as the sampling algorithm will not even be able to read the entire input circuit, let alone simulate it. Ideally, we would like the sampling algorithm to be able to read the entire input, even if it isn't able to perform a simulation, as in our setting for PSPACE.

Therefore, we consider a *succinct* version of our algorithmic task. Our input now is 1^n together with a $\mathfrak{C}$-circuit C of size at most n which is a compressed representation of a larger $\mathfrak{C}$-circuit C' of size $\mathsf{poly}(n)$ on n variables. Here, by saying that C is a compressed representation of C', we mean that we can recover any specific bit in the representation of C' by evaluating C on some input. Alternatively, one can think of C as a circuit for the *direct connection language* of C'.

We say that there is *efficient non-trivial sampling with* PH *oracle for the succinct version of dense* $\mathfrak{C}$*-satisfiability* if for some large enough constant k, there is a constant d and a probabilistic algorithm A running in time n^d with a PH oracle such that, given 1^n and $\mathfrak{C}$-circuit C as input, where C is a compressed representation of a $\mathfrak{C}$-circuit C' of size $\mathsf{poly}(n)$ on n variables accepting at least a $2/3$ fraction of inputs, A outputs some fixed satisfying assignment y of C' with probability at least $n^k/2^n$. The notion of uniformity we use for all of our results on lower bounds for NP is LOGTIME-uniformity [12].

▶ **Theorem 2** (Informal Statement). *Let* $\mathfrak{C}$ *be any class of circuits that can be simulated by Boolean circuits of polynomial size. If there is efficient non-trivial sampling with* PH *oracle for the succinct version of dense* $\mathfrak{C}$*-satisfiability, then* NP *does not have uniform* $\mathfrak{C}$*-circuits of polynomial size.*

Note that we allow the sampling algorithm oracle access to an arbitrary PH oracle in the hypothesis. This is intended to make the algorithmic task easier to solve.

Now we justify our claim that our algorithmic approach fulfils the two criteria we stated in Section 1.1. The first criterion is that the relevant algorithmic task should be a conventional one. This is true in our case since the algorithmic task we consider is that of efficiently sampling, with non-trivial probability, a fixed satisfying assignment to a circuit with many satisfying assignments. We observe that the second criterion holds as well, under standard cryptographic assumptions.

▶ **Theorem 3** (Informal Statement). *If one-way functions secure against super-polynomial size circuits exist, then there is efficient non-trivial sampling for the succint version of dense* Circuit*-satisfiability.*

We show Theorem 3 by using cryptographic pseudo-random generators to give an efficient non-trivial sampling algorithm for the succinct version of dense Circuit-satisfiability. Note that the sampling algorithm yielded by the assumption does not need access to a PH oracle. Also, an efficient non-trivial sampling algorithm for the succinct version of dense Circuit-satisfiability trivially implies a space-efficient non-trivial sampling algorithm, therefore Theorem 3 additionally evidences the feasibility of the approach toward PSPACE $\neq$ P.

One might still wonder if the algorithmic approach we present is far stronger than is actually necessary for uniform lower bounds. We show that under plausible complexity assumptions (which seem *morally* weaker than the lower bounds we are trying to prove), a uniform version of our approach[3] is in fact *necessary* in that the algorithmic tasks we propose become feasible.

Our ideas give an unconditional *algorithmic characterization* of PSPACE $\neq$ P.

▶ **Theorem 4** (Informal Statement). PSPACE $\neq$ P *iff* E *has circuits of size* $2^{o(n)}$ *on infinitely many input lengths, or if there is a space-efficient non-trivial algorithm for the uniform version of dense* Circuit*-satisfiability on infinitely many input lengths.*

In other words, a central uniform lower bound question in complexity theory reduces to either showing surprising non-uniform algorithms exist for E, or to solving our sampling task for general uniform Boolean circuits in a space-efficient non-trivial way. While there are such algorithmic characterizations of lower bounds for NEXP in the complexity-theoretic regime [31, 57, 55], the characterization above seems to the first one in the cryptographic regime.

While the results above indicate that our approach is sound, it is unclear a priori whether our algorithmic approach is feasible, i.e., if it has any hope of yielding new lower bounds in the near future. We do believe that the connection from algorithms to uniform lower bounds is interesting in itself, but we would like the framework to be capable at least of proving some known uniform lower bounds.

We show that the framework does indeed have this capability. On the one hand, we use known unconditional results about hitting set generators for weak circuit classes to observe that our sampling tasks are solvable for these circuit classes. On the other hand, we show that our framework can be used to give alternative proofs of well-known uniform lower bounds such as versions of the space hierarchy theorem [51] and Allenders' lower bound for Permanent [3]. This evidences the flexibility of the framework - it accommodates techniques exploiting specific properties of circuit classes as well as techniques based on simulation and diagonalization.

Finally, we use our approach to prove a couple of new lower bounds, and hope that even stronger lower bounds will follow using more sophisticated algorithmic ideas.

▶ **Theorem 5** (Informal Statement). NP *does not have uniform polynomial-size* AC^0 *circuits with a bottom layer of* Mod$m$ *gates for any positive integer* m*, not does it have uniform polynomial-size* AC^0 *circuits with a bottom layer of threshold gates.*

We note that it is a longstanding open problem to prove non-uniform super-polynomial size lower bounds in NP against AC^0 circuits with a bottom layer of Mod6 gates or a bottom layer of threshold gates, despite much effort[4]. We show that uniformity can be exploited

[3] By this we mean that our algorithmic task is only required to be feasible on uniform sequences of circuits.

[4] However, such lower bounds are known for non-deterministic quasi-polynomial time NQP [59, 40], in the complexity-theoretic regime. We are interested here in lower bounds in the cryptographic regime.

to prove lower bounds for these classes. As far as we are aware, this is the first case of a super-polynomial circuit lower bound for NP that holds for a uniform circuit class but is not known to hold for the corresponding non-uniform circuit class[5].

1.3 Discussion

In this sub-section, we discuss various features of our approach. Some of these have been mentioned before, but it might be useful for the reader to see them discussed together.

Another Algorithmic Approach to Lower Bounds. Our work is the latest in the line of works which apply algorithmic approaches to lower bound problems. However, it is the first to apply an algorithmic approach to lower bound problems in the cryptographic regime, and in particular with relevance to problems such as NP vs P and PSPACE vs P. The algorithmic method of [57, 58] shows that non-trivial algorithms for CircuitSAT and Circuit Acceptance Probability Estimation imply super-polynomial circuit lower bounds for NEXP. Building on [21, 33], Oliveira and Santhanam [44] showed that non-trivial randomized learning algorithms with membership queries over the uniform distribution for a class $\mathfrak{C}$ of circuits implies lower bounds in BPEXP against polynomial-size $\mathfrak{C}$-circuits.

Previous works on algorithmic approaches require a non-trivial upper bound on the running time of the algorithm to derive lower bound consequences. In our case, in contrast, while it is important that the algorithm is efficient, what matters more is the probability of sampling some *fixed* solution - we need this to be non-trivial. Another difference between our work and previous works is that previous works all rely ultimately on hierarchy theorems. In contrast, we do not use hierarchy theorems, and this enables us to deal with the cryptographic regime of lower bounds.

Exploiting the Power of NP. As mentioned in Section 1.1, there are several works beginning in the 1980s that establish super-polynomial circuit lower bounds for weak circuit classes using various combinatorial and algebraic techniques. An interesting feature of these results is that in most cases, the best lower bounds we know are for problems that are in P. For example, the strongest lower bounds we know for constant-depth circuits are for the Parity function, which is easily seen to be solvable by linear-size circuits. Clearly, any lower bound technique that yields lower bounds for problems in P is not capable of proving super-polynomial lower bounds for general Boolean circuits. Our approach, in contrast, uses the power of NP, and perhaps this suggests that the approach, or variants in it, might be useful in the long run to prove strong lower bounds.

The Importance of Uniformity. Historically, there has been a divide in complexity theory between approaches to non-uniform lower bounds and approaches to uniform lower bounds. Approaches to non-uniform lower bounds are often tailored to the circuit class of interest, identifying a structural weaknesses of the circuit class (such as being simplified by random restrictions or being approximable by low-degree polynomials) and then exploiting the weakness mathematically or algorithmically. Approaches to uniform lower bounds tend to be more generic, employing clever combinations of simulation and diagonalization techniques. Our work bridges this divide to an extent in that it identifies stand-alone algorithmic tasks

[5] Allender's lower bound for Permanent is another example of this phenomenon, but Permanent is neither known nor believed to be in NP

such that solutions for these tasks have implications for uniform lower bounds. The hope is that these algorithmic tasks can be solved efficiently and non-trivially by exploiting properties of the circuit class.

Generality of the Approach. Our algorithmic approach is *general* in multiple respects. First, it is relevant to lower bounds for *any* circuit class $\mathfrak{C}$ contained in the class of Boolean circuits. We do not even require even weak closure properties of the circuit classes. In particular, this makes our approach potentially relevant to frontier lower bounds against *fixed-depth classes*, eg., the class of depth-two threshold circuits.

Secondly, our approach can be adapted to lower bounds for uniform classes other than NP, simply by modifying the resource requirements of the algorithm. Theorem 1 illustrates how this works in the context of lower bounds for PSPACE. The approach is also capable of being adapted to lower bounds for other problems such as $\mathsf{Permanent}$, as shown in Sections 3 and 5.

White-Box Algorithmic Tasks in the Cryptographic Regime. Our approach puts the spotlight on white-box circuit-based algorithmic tasks in the cryptographic regime, where the algorithm does not have the resources to simulate the circuit it gets as input. To the best of our knowledge, these kinds of algorithms have not been considered before. We are specifically interested in sampling algorithms, and the extent to which sampling can outperform simulation. There are some known results about the power of low-complexity samplers, such as the work of Applebaum, Ishai and Kushilevitz[8] using the technique of randomizing polynomials to show that in many contexts, NC^1-samplable distributions can be replaced by NC^0-samplable distributions, and the work of Viola [53] initiating a line of research on the complexity of distributions. Perhaps ideas from these works or related works could be useful in approaching our algorithmic tasks.

Relevance to Known Barriers. Several previous attempts to attack NP vs P and related problems have run into one or the other of various complexity barriers, including the relativization barrier [11], the Natural Proofs barrier [48] and the algebrization barrier [1]. So it is important to examine how our approach fares against these barriers. As of now, we envision our approach as relevant mostly to frontier questions such as separating NP from uniform ACC^0 and uniform TC^0_2. It does not seem as though the relativization and algebraization barriers are relevant to such weak circuit classes, as existing lower bounds for weak circuit classes exploit weaknesses of the gate sets, and hence don't work when oracle gates with large fan-in occur in the circuit. The natural proofs barrier is not known to have any relevance to uniform circuit lower bounds. Indeed, even in the case of non-uniform circuit lower bounds against these classes, the natural proofs barrier would only be operative if there are pseudo-random functions in ACC^0 or TC^0_2, and no compelling evidence exists for the existence of such low-complexity pseudo-random functions.

1.4 Proof Ideas

We discuss here the ideas behind our approach and sketch the proofs of the connections from algorithms to lower bounds. We first discuss the proof ideas behind Theorem 2, and then the ideas behind Theorem 1.

Recall that our goal is to develop an *algorithmic approach* to uniform lower bounds for NP and PSPACE. We would like our algorithmic approach to involve a conventional algorithmic task, and for there to be complexity-theoretic evidence that the algorithmic task is feasible. Ideally, the algorithmic task should require only a marginal improvement over known algorithms.

Our starting point is an elegant idea of Hirahara [28]. Hirahara was interested in the problem of proving uniform lower bounds for the problem R_{Kt} of determining whether an input string x has high Kt complexity. Recall that the Kt complexity of a string is the minimum of $|p| + \log(t)$ over programs p and time bounds t such that $U^t(p, \epsilon) = x$, i.e., a universal machine halts within t steps on input p and outputs x. R_{Kt} is known to be EXP-complete [6] but only with respect to *non-uniform* truth-table reductions or NP Turing reductions. It is a long-standing open problem whether R_{Kt} is in P. Hirahara showed that R_{Kt} does not have P-uniform ACC^0 circuits of polynomial size.

His idea is as follows: suppose that there is a non-trivial algorithm for satisfiability for a circuit class $\mathfrak{C}$. Namely, we can find a satisfying assignment y of length n for a satisfiable $\mathfrak{C}$-circuit C on n variables in deterministic time $2^n/n^{\omega(1)}$. Then it is not too hard to show that y has Kt complexity $n - \omega(\log(n))$ *conditioned* on C. Now, if C itself were a uniform circuit generated by an efficient procedure given input 1^n, that implies that $\mathsf{Kt}(C) = O(\log(n))$, and by first generating C and then generating y conditioned on C, we get that y has Kt complexity $n - \Omega(\log(n))$.

We can use this to derive a contradiction to the assumption that R_{Kt} has P-uniform $\mathfrak{C}$-circuits. We consider the uniform circuit ACC^0 C_n assumed to solve R_{Kt}. The satisfying assignments of C_n are precisely the hard Kt strings, and in particular we can assume that every satisfying assignment has Kt complexity $n - \Omega(\log(n))$. But then the argument in the previous para yields a contradiction, since y is a satisfying assignment to C_n and has Kt complexity $n - \omega(\log(n))$.

Since we know that satisfiability of ACC^0 circuits can be solved in non-trivial time [58], we derive a P-uniform ACC^0 lower bound for R_{Kt}. This is still quite far from the desired result showing $R_{\mathsf{Kt}} \notin \mathsf{P}$, but it constitutes some progress. Hirahara uses similar ideas to show that the set of K^t-random strings is not in P for any super-polynomial t.

While the idea of the proof is novel, the lower bound result for R_{Kt} is still in the complexity-theoretic regime of lower bounds, since we are trying to prove uniform super-polynomial lower bound for a language that is known to be complete for exponential time. However, we observe that this isn't *intrinsic* to the approach; rather it depends on which notion of resource-bounded Kolmogorov complexity we analyze. In this case, we analyzed Kt complexity, but in principle we could analyze some other resource-bounded Kolmogorov complexity measure.

In particular, let us imagine trying to upper-bound the $\mathsf{K}^{\mathsf{poly}}$ complexity of satisfying assignments to circuits. Our reason for doing this is that the language of hard $\mathsf{K}^{\mathsf{poly}}$ strings is in NP, hence if we are able to carry through an argument analogous to Hirahara's argument, we might be able to show a uniform circuit lower bound for NP. One obstacle that presents itself is that it is unclear what sort of algorithm we need to analyze in order to upper bound the $\mathsf{K}^{\mathsf{poly}}$ complexity of solutions. Another obstacle is that it is unclear even why there should be a solution of low $\mathsf{K}^{\mathsf{poly}}$ complexity.

Let us try to address the second obstacle first, and come up with an algorithmic task where there are likely to be solutions of non-trivial $\mathsf{K}^{\mathsf{poly}}$ complexity. Here we make a crucial observation: Hirahara considers the Satisfiability problem for general circuits, but in fact we can consider the Satisfiability problem for *dense* circuits instead. The reason is that if we are going to use the argument on a circuit that is presumed to decide the set of hard $\mathsf{K}^{\mathsf{poly}}$ strings correctly, the circuit will have many accepting inputs, since a random string is likely to be $\mathsf{K}^{\mathsf{poly}}$-hard.

Considering Dense Circuit Satisfiability makes our approach more plausible, since if we make cryptographic derandomization assumptions, we are at least likely to have solutions with low $\mathsf{K}^{\mathsf{poly}}$ complexity conditioned on the circuit. However, the first obstacle remains - it is not clear how to analyze $\mathsf{K}^{\mathsf{poly}}$ complexity of solutions for any natural algorithmic task.

Our idea is to use *probabilistic* notions of time-bounded Kolmogorov complexity. Several notions of probabilistic time-bounded Kolmogorov complexity have recently been defined and studied in [43, 35, 24, 36, 34]. A notion that is ideal for our purposes is the notion of $\mathsf{pK}^{\mathsf{poly}}$ [24, 36]. Intuitively, a string x has low $\mathsf{pK}^{\mathsf{poly}}$ string if for most random strings r of a given polynomial length, there is a small description p_r from which x can be reconstructed in polynomial time given r. This notion of probabilistic Kolmogorov complexity has two very appealing features. First, $\mathsf{pK}^{\mathsf{poly}}$ complexity turns out to be closely tied to a very natural algorithmic task, namely sampling solutions of search problems. This allows us to define a natural algorithmic task that makes no reference to Kolmogorov complexity notions. Second, $\mathsf{pK}^{\mathsf{poly}}$ complexity has a so-called Optimal Coding Theorem, which implies that strings sampled efficiently with probability p have $\mathsf{pK}^{\mathsf{poly}}$ complexity very close to $\log(1/p)$. This allows us to define an algorithmic task that involves just a non-trivial improvement in success probability over existing efficient algorithms.

The price we pay is that the language of hard $\mathsf{pK}^{\mathsf{poly}}$ strings is no longer in NP. However, we can define a promise version of the language of hard strings which is in AM, and this turns out to be sufficient for our purposes, by using an additional idea.

However, our assumption in Theorem 2 is for the succinct version of Dense Circuit Satisfiability. In order to use this version, we observe that we use the assumption of feasibility of algorithmic tasks only on uniform circuits, which are succinctly representable. In fact, when we are arguing by contradiciton, we can efficiently recover a succinct description of the circuit, where the description itself belongs to the class of circuits that are being described. This additional step allows us to complete the proof of Theorem 2.

To extend this proof idea to lower bounds for PSPACE in Theorem 1, we show that it is enough to simply change the resource requirements of the algorithmic tasks to a fixed polynomial space bound rather than a fixed polynomial time bound. The natural idea for analyzing this would be to consider a notion of space-bounded Kolmogorov complexity, but our argument by contradiction finds a short-cut. We use an argument by contradiction again to observe that if indeed PSPACE had small uniform circuits, then $\mathsf{PSPACE} = \mathsf{P}$. This implies that our small-space sampling algorithm can be simulated by a time-efficient sampling algorithm, and then we can use the same machinery as in Theorem 2 to complete the proof.

For Theorem 3, we use the fact [26] that one-way functions imply the existence of cryptographic pseudo-random generators (PRG) with seed length n^ϵ for any $\epsilon > 0$. A cryptographic PRG can be used to solve our sampling task efficiently by simply outputting a random element in the range of the PRG, which can be done in time a fixed polynomial in n. By the pseudo-randomness property, most elements of the range will be satisfying assignment of the dense circuit on which we are solving the sampling task, and each such element will be output with probability 2^{-n^ϵ}, which is non-trivial. Note that we do not even need to look at the circuit on which we are solving the sampling task.

For the algorithmic characterization of $\mathsf{PSPACE} \neq \mathsf{P}$, we use certain properties of the set of strings $L = \mathsf{MKSP}[\sqrt{n}]$ of Kolmogorov space-bounded complexity at most $\sqrt{n}$. It follows from [6] that L is hard on average for polynomial time in a zero-error sense if $\mathsf{PSPACE} \neq \mathsf{BPP}$, and it can be shown using ideas in [49] that the hardness on average of L implies the existence of a crytographic hitting-set generator against polynomial size circuits. A cryptographic hitting-set generator can be used to solve our sampling task space-efficiently using the observations in the previous paragraph. To complete our characterization, we use a standard result from derandomization [32], namely that either E has circuits of size $2^{o(n)}$ infinitely often or $\mathsf{BPP} = \mathsf{P}$.

The proof of Theorem 5 proceeds by designing efficient non-trivial sampling algorithms with PH oracle for the succinct version of Dense $\mathfrak{C}$ Satisfiability for $\mathfrak{C} = \mathsf{AC}^0 \circ (\mathsf{Mod}m)$ and $\mathfrak{C} = \mathsf{AC}^0 \circ \mathsf{Thr}$.

Several recent works in various areas of complexity theory, including learning theory, pseudorandomness, cryptography, structural complexity and proof complexity, have developed and exploited ideas from *meta-complexity*, i.e., the complexity of complexity. We refer to [5] for a recent survey. The ideas of our proof are another illustration of this phenomenon. Like the recent work of Hirahara [29] on average-case hardness of NP from exponential worst-case assumptions, our results use meta-complexity as a catalyst: the results make no reference to meta-complexity, yet the proofs use meta-complexity crucially.

2 Preliminaries

2.1 Standard Complexity Notions

The textbook by Arora and Barak [9] is an excellent reference for basic notions in complexity theory. Here we recall a few that are especially relevant to this paper.

Computational problems are typically modelled as decision problems, where each input is either accepted or rejected. Occasionally we are interested in *promise problems*, where the set of accepted inputs is disjoint from the set of rejected inputs, but some inputs might not belong to either category. Formally, a promise problem Γ over $\{0,1\}$ is a pair $(\Gamma_{YES}, \Gamma_{NO})$ where $\Gamma_{YES}, \Gamma_{NO} \subseteq \{0,1\}^*$ and $\Gamma_{YES} \cap \Gamma_{NO} = \varnothing$. The complement of a promise problem $(\Gamma_{YES}, \Gamma_{NO})$ is the promise problem $(\Gamma_{NO}, \Gamma_{YES})$. We say that a language L is consistent with a promise problem $\Gamma = (\Gamma_{YES}, \Gamma_{NO})$ if $\Gamma_{YES} \subseteq L$ and $\Gamma_{NO} \subseteq \bar{L}$.

We will need to be careful about which Turing machine model we consider, since we are often interested in computations that run in sub-linear time. We will use the random access Turing machine model, where each tape of a multi-tape Turing machine has a corresponding address tape. When the address tape for tape k has index i written on it, and the machine enters a special tape, the contents of the i'th tape cell of the k'th tape can be accessed in unit time. We also consider oracle Turing machines, where the random access Turing machine is provided with a separate oracle tape, on which queries to the oracle can be made, and answered in unit time.

We will be considering various standard circuit classes, including the class AC^0_d of constant-depth circuits of depth d with AND and OR gates, the class $\mathsf{AC}^0_d[p]$ of constant-depth circuits of depth d with AND, OR and MOD p gates for prime p, the class ACC^0 of constant-depth circuits of depth d with AND, OR and modular gates, the class TC^0_d of depth-d threshold circuits, the class Formula of Boolean formulas, and the class Circuit of Boolean circuit. By default, whenever we refer to a circuit class, we will mean the *non-uniform* version of the circuit class. However, as is standard terminology, NC^1 will refer to LOGTIME-uniform circuits of logarithmic depth. We will occasionally abuse notation and use the name of a circuit class to refer to the circuit class as well as to the class of languages decided by the circuit class.

We will mainly be using two standard notions of uniformity for circuits: LOGTIME-uniformity and LOGSPACE-uniformity. We refer to [12] for precise definitions of these notions as well as a detailed discussion on motivation. Briefly, LOGTIME-uniformity means that the *direct connection language* of a sequence $\{C_n\}$ of circuits, encoding types of gates and connections between them in a natural way, is decidable in time logarithmic in the size of the circuit. LOGSPACE-uniformity means that the direct connection language of $\{C_n\}$ is decidable in logarithmic space; equivalently, a description of C_n can be computed in logarithmic space given 1^n as input. The main property we will require of LOGTIME-uniformity is that any given bit of the description of a LOGTIME-uniform circuit C can be computed in time logarithmic in the size of the circuit. This is the case when a circuit is represented in a standard way, i.e., as a list of gate types and connections between gates in some pre-determined order.

We won't formally define direct connection languages, since the details depend on the circuit classes of interest, and we consider a wide variety of circuit classes. However, in each case, we will be able to answer questions about the gate type, about whether the i'th child of a gate is a certain other gate, and about whether a gate has more than i inputs using a single query to the direct connection language. Things get a bit subtle when considering circuits where the gates have weights, eg., threshold gates which check whether an integer-weighted sum of inputs is at least some integer value, or modular gates which check whether integer-weighted sums of the inputs belong to some set of values modulo a given integer. Indeed, we consider AC^0 circuits with a bottom layer of $\mathsf{Mod}m$ or Thr gates in Section 5, and we assume there that the gates are irredundant, i.e., that the weights aren't unnecessarily large. In particular, we assume that the weights for any $\mathsf{Mod}m$ gate are at most m, and that the weights for a threshold gate on n inputs is at most $n^{O(n)}$, which is true without loss of generality [39]. In these cases, the weights can be extracted using a fixed polynomial number of queries to the direct connection language - this will be crucial in the proofs of our new lower bounds.

We will also refer to $\mathsf{POLYLOG}$-uniformity, where the direct connection language is decidable in time poly-logarithmic in the size of the circuit.

We say that a circuit class $\mathfrak{C}$ is polynomially simulatable if there is a polynomial-time algorithm which, given a circuit C from $\mathfrak{C}$ and an input x to C, computes $C(x)$.

▶ **Proposition 6.** *Suppose a circuit class $\mathfrak{C}$ is polynomially simulatable, and let L be a language that has $\mathsf{LOGTIME}$-uniform (resp. $\mathsf{LOGSPACE}$-uniform) $\mathfrak{C}$-circuits of polynomial size. Then L has $\mathsf{LOGTIME}$-uniform (resp. $\mathsf{LOGSPACE}$-uniform) Boolean circuits of polynomial size.*

Recall that CH [54, 45] is the *counting hierarchy*, whose first level $\mathsf{CH}_1 = \mathsf{PP}$ and i'th level $\mathsf{CH}_i = \mathsf{PP}^{\mathsf{CH}_{i-1}}$. We will also need Toda's theorem [52].

▶ **Theorem 7** ([52])**.** $\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{PP}} = \mathsf{P}^{\#\mathsf{P}}$.

2.2 Meta-Complexity

Here we define various notions of Kolmogorov complexity that will be needed in this work.

Throughout, we fix a time-efficient universal Turing machine U. Notions of Kolmogorov complexity are defined relative to this universal machine U, but since the notions and results we use are robust to the precise choice of the universal machine, we suppress the dependence on U.

Given a string x, the Kolmogorov complexity $\mathsf{K}(x)$ is defined to be the size of the smallest program p such that $U(p, \epsilon)) = x$. Given a time bound $t : \mathbb{N} \to \mathbb{N}$ and a string x, the t-time bounded Kolmogorov complexity of x is defined as follows: $\mathsf{K}^t(x)$ is the size of the smallest program p such that $U^{t(|x|)}(p, \epsilon) = x$, where U^T means that that the universal machine is restricted to run for at most T steps.

K and K^t are *deterministic* notions of Kolmogorov complexity, in that a string is recovered from its compressed representation by a deterministic program. We require a probabilistic notion of Kolmogorov complexity recently introduced in [24].

Given a time bound $t : \mathbb{N} \to \mathbb{N}$, a string x and a number $\rho \in [0, 1]$, we say that x has ρ-confidence pK^t complexity at most k if for at least ρ fraction of random strings r of length $t(|x|)$, there is a program $p_r, |p_r| \leqslant k$, for which $U^{t(|x|)}(p_r, r) = x$.

▶ **Proposition 8.** *For any time bound $t : \mathbb{N} \to \mathbb{N}$, non-negative integers n, k and $\rho \in [0, 1]$, at most $2^{k+1}/\rho$ strings have ρ-confidence pK^t-complexity at most k.*

Let $t : \mathbb{N} \to \mathbb{N}$ be a time bound. Given a complexity parameter $s : \mathbb{N} \to \mathbb{N}$ and real numbers $\rho, \delta \in [0,1]$ such that $\delta < \rho$, we define the meta-complexity promise problem $R_{\mathsf{pK}^t}[s, \rho, \delta] = (\Gamma_{YES}, \Gamma_{NO})$, with $\Gamma_{YES}, \Gamma_{NO} \subseteq \{0,1\}^*$ and $\Gamma_{YES} \cap \Gamma_{NO} = \varnothing$, as follows. A string $x \in \Gamma_{NO}$ if the ρ-confidence pK^t complexity of x is at most $s(|x|)$. A string $x \in \Gamma_{YES}$ if the δ-confidence pK^t complexity of x is not at most $s(|x|)$.

▶ **Proposition 9.** *$R_{\mathsf{pK}^t}[s, \rho, \delta] \in \mathsf{coAM}$. Moreover, if $s(n) + \log(1/\delta) < n - 2$ for each $n \in \mathbb{N}$, then Γ_{YES} contains at least a $3/4$ fraction of all strings of length n.*

2.3 One-Way Functions and Pseudorandomness

We need the standard cryptographic notion of a non-uniformly secure one-way function. In fact, we define the length-preserving variant of the notion, which is without loss of generality.

▶ **Definition 10.** *Let $s : \mathbb{N} \to \mathbb{N}$. A function $f = \{f_n\}, f_n : \{0,1\}^n \to \{0,1\}^n$ is said to be an $s(\cdot)$-secure one-way function if for each sequence of circuits $\{D_n\}$ of size $\mathsf{poly}(s(n))$, we have that $\Pr_{x \sim \{0,1\}^n}[f(D(f(x))) = f(x)] = 1/n^{\omega(1)}$.*

Next we define the notion of a cryptographic pseudo-random generator. For ease of application, we define the notion slightly differently than the standard notion, with the computability of the PRG measured as a function of the output size.

▶ **Definition 11.** *Let $\ell : \mathbb{N} \to \mathbb{N}$ be a function such that $\ell(n) \leqslant n$ for all $n \in \mathbb{N}$. A cryptographic pseudo-random generator with seed length ℓ is a function $G = \{G_n\}, G_n : \{0,1\}^{\ell(n)} \to \{0,1\}^n$ computable in time $\mathsf{poly}(n)$ such that for any algorithm D running in time $\mathsf{poly}(n)$, $|\Pr_{y \in \{0,1\}^n} D(y) - \Pr_{z \in \{0,1\}^{\ell(n)}} D(G(z))| = 1/n^{\omega(1)}$.*

One of the foundational result in cryptography is that cryptographic pseudo-random generators with small seed length can be based on the existence of one-way functions with super-polynomial hardness.

▶ **Theorem 12** ([26])**.** *Suppose there is an $n^{\omega(1)}$-secure one-way function. Then there is a cryptographic pseudo-random generator with seed length $n^{o(1)}$.*

We will also need the notion of a cryptographic hitting set generator useful against a circuit class $\mathfrak{C}$.

▶ **Definition 13.** *Let $\ell : \mathbb{N} \to \mathbb{N}$ be a function such that $\ell(n) \leqslant n$ for all $n \in \mathbb{N}$, and let $\mathfrak{C}$ be a circuit class. A cryptographic hitting set generator with seed length ℓ against $\mathfrak{C}$ is a function $G = \{G_n\}, G_n : \{0,1\}^{\ell(n)} \to \{0,1\}^n$ computable in time $\mathsf{poly}(n)$ such that for any sequence of $\mathfrak{C}$-circuits $\{C_n\}$ such that C_n accepts at least a $1/n$ fraction of n-bit inputs for large enough n, there exists a sequence $\{y_n\}$ with $y_n \in \{0,1\}^{\ell(n)}$ for each n such that $C(G_n(y_n)) = 1$.*

2.4 Search Problems and Samplers

The algorithmic tasks we consider will involve solving search problems. We first define the notion of a promise search problem.

▶ **Definition 14.** *A promise search problem $\mathcal{S}$ is a pair (R, X) where $R \subseteq \{0,1\}^*$ is a polynomial-time computable binary relation and $X \subseteq \{0,1\}^*$ is a subset of inputs. A solution to the search problem $\mathcal{S}$ on input x is any string y such that $(x, y) \in R$. We say that an algorithm A solves the promise search problem $\mathcal{S}$ if for each $x \in X$, A outputs a solution to $\mathcal{S}$ on x if one exists.*

We will be interested in a specific promise search problem where the task is to find satisfying assignments of circuits that accept most of their inputs.

▶ **Definition 15.** *Let $\mathfrak{C}$ be a circuit class. The promise search problem* $\mathsf{Dense}-\mathfrak{C}-\mathsf{SAT}$ *is defined by the following binary relation R and input set X. R consists of all pairs (C, x), where C is (the encoding of) a $\mathfrak{C}$-circuit, and x is a satisfying assignment of C. X consists of all $\mathfrak{C}$-circuit C such that C accepts at least a $2/3$ fraction of its inputs.*

For technical reasons, we will also need the notion of a sampler. Note that our notion of sampler simply models an algorithm that samples a distribution, and is not related to the notion of sampler in the theory of randomness extraction.

▶ **Definition 16.** *A sampler is a polynomial-time randomized algorithm which, given 1^n as input for $n \in \mathbb{N}$, samples a distribution on n-bit outputs.*

3 From Algorithms to Uniform Lower Bounds

In this section, we show our main results about connections from circuit sampling tasks to uniform lower bounds for NP, PSPACE and Permanent.

3.1 An Algorithmic Approach to Uniform Lower Bounds for NP

We first define the algorithmic task we will consider in this sub-section.

▶ **Definition 17.** *Given circuits C and C', we say that C encodes C' if the Boolean function computed by C is the direct connection language of C'.*

▶ **Definition 18.** *Let $\mathfrak{C}$ be a circuit class. We say that there is efficient non-trivial sampling (resp. efficient non-trivial sampling with* PH *oracle) for the succinct version of* $\mathsf{Dense}-\mathfrak{C}-\mathsf{SAT}$ *if for every $k > 0$ there is a probabilistic algorithm A (resp. probabilistic algorithm A with* PH *oracle) which, given 1^n and a $\mathfrak{C}$-circuit C of size at most n, where C encodes a $\mathfrak{C}$-circuit C' of size at most n^b on n variables accepting at least a $2/3$ fraction of its inputs, runs in time n^d (for some constant d independent of b) and outputs some fixed satisfying assignment y to C' with probability at least $n^k/2^n$.*

The following is a version of the Optimal Coding Theorem for pK^t in [36] with slightly improved parameters[6] that are important in our application.

▶ **Lemma 19.** *Let S be a sampler that runs in time n^s on input of length n, where $s > 0$ is a constant. There is a constant β such that if S samples some $y \in \{0,1\}^n$ with probability p, then y has $3/4$-confidence $\mathsf{pK}^{n^{\beta s}}$ complexity at most $\log(1/p) + 3\log(n)$.*

In fact it can be shown for any search problem that sampling of a fixed solution to the search problem with non-trivial probability is equivalent to the existence of solutions that have non-trivial *conditional* $\mathsf{pK}^{\mathsf{poly}}$ complexity, for an appropriately defined notion of conditional $\mathsf{pK}^{\mathsf{poly}}$ complexity. We do not pursue this direction here to avoid detracting from the focus of the paper on approaches to uniform lower bounds.

We need an easy lemma to allow us to deal with the issue of promise problems.

[6] Specifically, the coding theorem in [36] involves an additive term that is logarithmic in the time bound of the sampler, while the additive term in our bound is logarithmic in the input length of the sampler

▶ **Lemma 20.** *Let Γ be a promise problem in* coAM*. If* $\mathsf{NP} = \mathsf{P}$*, then there is a language L_Γ consistent with Γ such that $L_\Gamma \in \mathsf{P}$.*

We are now ready to establish the main result of this sub-section. The following is a more formal version of Theorem 2.

▶ **Theorem 21.** *Let $\mathfrak{C}$ be a polynomially simulatable circuit class. Suppose that there is efficient non-trivial sampling with* PH *oracle for the succinct version of* $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$*. Then* NP *does not have* LOGTIME*-uniform $\mathfrak{C}$-circuits of polynomial size.*

Theorem 21 is very general in that there are no constraints on the circuit class $\mathfrak{C}$ apart from polynomial simulatability, and in particular we do not require any closure properties of $\mathfrak{C}$. We immediately obtain the following corollaries. The first two corollaries concern frontier questions in complexity theory, and the last two concern two of the central problems in the area.

▶ **Corollary 22.** *Suppose that for any $d \in \mathbb{N}$ there is efficient non-trivial sampling with* PH *oracle for the succinct version of* $\mathsf{Dense} - \mathsf{ACC}^0_d - \mathsf{SAT}$*. Then* NP *does not have* LOGTIME*-uniform* ACC^0*-circuits of polynomial size.*

▶ **Corollary 23.** *Suppose that there is efficient non-trivial sampling with* PH *oracle for the succinct version of* $\mathsf{Dense} - \mathsf{TC}^0_2 - \mathsf{SAT}$*. Then* NP *does not have* LOGTIME*-uniform* TC^0_2*-circuits of polynomial size.*

▶ **Corollary 24.** *Suppose that there is efficient non-trivial sampling with* PH *oracle for the succinct version of* $\mathsf{Dense} - \mathsf{Formula} - \mathsf{SAT}$*. Then* $\mathsf{NP} \neq \mathsf{NC}^1$*.*

▶ **Corollary 25.** *Suppose that there is efficient non-trivial sampling with* PH *oracle for the succinct version of* $\mathsf{Dense} - \mathsf{Circuit} - \mathsf{SAT}$*. Then* $\mathsf{NP} \neq \mathsf{P}$*.*

3.2 An Algorithmic Approach to Uniform Lower Bounds for PSPACE

Theorem 21 gives an algorithmic approach to showing uniform lower bounds for NP. It is natural to ask if there is a similar algorithmic approach involving an easier algorithmic task toward showing uniform lower bounds for larger classes such as PSPACE. We provide an affirmative answer in this sub-section.

We begin by defining a space-efficient notion of sampling.

▶ **Definition 26.** *Let $\mathfrak{C}$ be a circuit class. We say that there is space-efficient non-trivial sampling for* $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$ *if for every $k > 0$ there is a $d > 0$ and a probabilistic algorithm A such that for every $b > 0$, when A is given an instance C of* $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$*, where C is of size $m = n^b$ on n variables, A has space and randomness complexity n^d for some fixed constant d, and outputs some fixed satisfying assignment y to C with probability at least $n^k/2^n$.*

We would like to show that space-efficient non-trivial sampling for $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$ leads to lower bounds for PSPACE against LOGSPACE-uniform $\mathfrak{C}$-circuits of polynomial size. A natural strategy to achieve this is to define a new notion of probabilistic *space-bounded* Kolmogorov complexity and work in analogy to Section 3.2. But in fact we can short-circuit this process and adapt the argument in Section 3.2 more directly, while still working with time-bounded Kolmogorov complexity. We simply exploit the fact that our argument works by contradiction, and our initial assumption implies that $\mathsf{PSPACE} = \mathsf{P}$, which means that the space-bounded and time-bounded notions of Kolmogorov complexity essentially coincide.

The following is a more formal version of Theorem 1 from the Introduction.

▶ **Theorem 27.** *Let $\mathfrak{C}$ be a polynomially simulatable circuit class, and suppose that there is space-efficient non-trivial sampling for* Dense $-$ $\mathfrak{C}$ $-$ SAT*. Then* PSPACE *does not have* LOGSPACE*-uniform $\mathfrak{C}$-circuits of polynomial size.*

We immediately obtain the following corollary, which concerns a long-standing open problem in complexity theory . Note that analogues of Corollaries 22, 23, 24 are known to hold unconditionally for PSPACE by the space hierarchy theorem [51].

▶ **Corollary 28.** *Suppose that there is space-efficient non-trivial sampling for* Dense $-$ Circuit $-$ SAT*. Then* PSPACE $\neq$ P*.*

There are a couple of differences in the hypothesis of Theorem 27 as compared to Theorem 21. The first is that the sampling algorithm isn't given access to an oracle. However, this is an insignificant difference. Given a space-efficient sampling algorithm access to a PSPACE oracle doesn't really increase its power, as PSPACE is closed under polynomial-space reductions.

The second is that the sampling algorithm is given the entire circuit as input rather than a succinct representation of it. At first sight, this looks more like a hypothesis about a white-box algorithm rather than about a restricted white-box algorithm. But in fact, the hypothesis isn't fully white box, as the sampling algorithm doesn't have enough space to simulate the input circuit in general. The input circuit can be of size an arbitrary polynomial in n, while the sampling algorithm needs to run in space a fixed polynomial in n.

We could consider the succinct version of the sampling problem here too, and the connection would still go through, just as in the proof of Theorem 21. This would yield our desired conclusion under a weaker hypothesis, as space-efficient non-trivial sampling implies space-efficient non-trivial sampling for the succinct version. Indeed, suppose we have a space-efficient non-trivial sampling algorithm for the succinct version, where we are given a circuit C of size at most n encoding a circuit C' on n variables for which we want to sample a satisfying assignment. We could simulate any query to C' in the non-succinct version by running C in the succinct version, which costs at most a fixed polynomial overhead in space.

Our reason for stating the weaker result here (by using a stronger hypothesis) is that in some situations the stronger hypothesis seems more natural to attack, because it feels more similar in flavour to the white-box version. Indeed, when we reprove versions of the space hierarchy theorem using our approach in Section 5, we do establish the stronger hypothesis for the circuit class of interest there.

An analogue of the stronger hypothesis could also be defined and considered in the setting of uniform lower bounds for NP, but feels less achievable there, as it would involve solving the sampling task without even reading the entire input circuit. This might still be possible for weak circuit classes, such as depth-two circuits, but coming up with algorithmic approaches to the hypothesis for stronger circuit classes might be hard. In contrast, when considering the stronger hypothesis in the setting of uniform lower bounds for PSPACE, we are allowed to read the entire input circuit, just not to use a large amount of space when trying to sample from it.

An algorithmic approach to the problem of showing that Permanent is not in NC^1 can be developed along very similar lines to Theorem 21 and Theorem 27. Here Permanent is the problem of computing the permanent of a Boolean matrix over the integers, encoded in a standard way as a decision problem.

▶ **Theorem 29.** *Let $\mathfrak{C}$ be a polynomially simulatable circuit class that is closed under projections. Suppose that there is efficient non-trivial sampling with* CH *oracle for the succinct version of* Dense $-$ $\mathfrak{C}$ $-$ SAT*. Then* Permanent *does not have* LOGTIME*-uniform $\mathfrak{C}$-circuits of polynomial size.*

4 Soundness of the Approach

4.1 Solving the Algorithmic Tasks under Standard Cryptographic Assumptions

As discussed, one of the most important criteria for an algorithmic approach to a lower bound problem is that the approach should be sound, i.e., there should ideally be evidence that the relevant algorithmic task is feasible. We begin by providing cryptographic evidence that the algorithmic tasks discussed in Section 3 are feasible. The following is a more formal version of Theorem 3.

▶ **Theorem 30.** *Suppose there is an $n^{\omega(1)}$-secure one-way function. Let $\mathfrak{C}$ be any circuit class that is polynomially simulatable. Then there is efficient non-trivial sampling for* $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$. *Indeed, there is a probabilistic algorithm A which, given a $\mathfrak{C}$-circuit C of size $m = n^b$ on n variables accepting at least $2/3$ of its inputs, runs in time n^d (where d is independent of b) and outputs some fixed satisfying assignment of C with probability $2^{-n^{o(1)}}$.*

Note that the algorithm A in the proof of Theorem 30 is *oblivious*: it does not consult its input. Thus the standard cryptographic assumption of the existence of one-way functions implies an oblivious solution to our algorithmic task, while we only require a constrained white-box solution.

It turns out that if we are interested in a white-box solution to the algorithmic tasks that is not constrained, i.e., the algorithm is not required to be efficient, then the task is indeed solvable with non-trivial probability by a sampling argument.

▶ **Theorem 31.** *Let $\mathfrak{C}$ be any polynomially simulatable circuit class. For each $k > 0$ there is a probabilistic algorithm A which, given a $\mathfrak{C}$-circuit C of size $\mathsf{poly}(n)$ on n variables accepting at least $2/3$ fraction of its inputs, runs in time $\mathsf{poly}(n)$ and outputs a fixed satisfying assignment y of C with probability at least $n^k/2^n$ for large enough $n \in \mathbb{N}$.*

4.2 Necessity of the Approach

We have argued that our algorithmic approach is sound, but could it be that what we require algorithmically is much stronger than what is needed? Next we show that under plausible complexity-theoretic assumptions, a version of our algorithmic approach to lower bounds for NP and PSPACE is in fact *necessary*. Specifically, we define a *uniform* version of our algorithmic approach, where efficient non-trivial sampling is only required for each uniform sequence of circuits, rather than for circuits given as input to an algorithm. We observe that our proofs in Section 3 go through if the uniform version is feasible, and then show that under our complexity assumptions, $\mathsf{NP} \neq \mathsf{P}$ and $\mathsf{PSPACE} \neq \mathsf{P}$ actually imply the feasibility of the uniform versions of our assumptions.

We need a standard complexity-theoretic derandomization assumption, as well as an additional assumption about NP-hardness of a meta-complexity problem in the case of uniform lower bounds for NP. We discuss the case of uniform lower bounds for NP first, and then move on to the case of uniform lower bounds for PSPACE. First we define a uniform version of our algorithmic approach for NP.

▶ **Definition 32.** *Let $\mathfrak{C}$ be a circuit class. We say that there is efficient non-trivial sampling for the uniform version of* $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$ *if for every $k > 0$ and every* LOGTIME*-uniform sequence $\{C_n\}$ of $\mathfrak{C}$-circuits of size $\mathsf{poly}(n)$ on n variables such that C_n accepts at least $2/3$ fraction of inputs of length n, there is a probabilistic algorithm A which, given input 1^n, runs in time n^d (for some constant d independent of the exponent in the size of C_n) and outputs some fixed satisfying assignment y to C_n with probability at least $n^k/2^n$.*

Next we require a standard derandomization result.

▶ **Theorem 33** ([32]). *Suppose* E *requires exponential-size Boolean circuits. Then* BPP = P.

We need to define the meta-complexity problem MCSP and what it means for this problem to be average-case hard and to be hard to approximate.

▶ **Definition 34.** *Given a size function $s : \mathbb{N} \to \mathbb{N}$ where $s(N) \leqslant N$ for each positive integer N, we define the problem* MCSP$[s]$ *as follows. YES instances of* MCSP$[s]$ *are strings y of length N where $N = 2^n$ for some integer n and f_y has Boolean circuits of size at most $s(N)$, where f_y is the Boolean function whose truth table is y.*

We say that MCSP$[s]$ *is zero-error easy on average over the uniform distribution if there is a deterministic polynomial-time algorithm which, given input y, always outputs 0, 1 or '?'; always correctly classifies y with respect to* MCSP$[s]$ *when it outputs a Boolean value; and outputs a non-'?' value with probability at least $1/poly(N)$ over $y \sim \{0,1\}^N$. We say that* MCSP$[s]$ *is zero-error hard on average over the uniform distribution if it is not zero-error easy on average over the uniform distribution.*

Given a function $\gamma : \mathbb{N} \to \mathbb{N}$, we say that MCSP *is γ-hard to approximate (resp. γ-hard to approximate probabilistically) if there is no polynomial time algorithm (resp. probabilistic polynomial time algorithm) solving the following promise problem Γ. Γ_{YES} consists of tuples $(y, 1^s)$ such that y is the truth table of a function with Boolean circuits of size at most s. Γ_{NO} consists of tuples $(y, 1^s)$ such that y is the truth table of a function with no Boolean circuits of size at most $\gamma(|y|)s$.*

We say that it is NP*-hard to γ-approximate* MCSP *if there is a polynomial-time reduction from* SAT *to the promise problem Γ described above.*

We will use the following approximation to average-case reduction of Hirahara [27].

▶ **Theorem 35** ([27]). *Suppose that* MCSP *is $N^{1-\epsilon}$-hard to approximate probabilistically for each $\epsilon > 0$. Then for each $\delta > 0$,* MCSP$[N^\delta]$ *is zero-error average-case hard over the uniform distribution.*

Now we are ready to prove our result about necessity of the uniform version of our algorithmic approach in the case of uniform lower bounds for NP.

▶ **Theorem 36.** *Suppose that* E *requires exponential-size Boolean circuits, and moreover that* MCSP *is* NP*-hard to $N^{1-\epsilon}$-approximate for each $\epsilon > 0$. Then* NP $\neq$ P *iff there is efficient non-trivial sampling for the uniform version of* Dense $-$ Circuit $-$ SAT*.*

Given that the uniform version of the algorithmic approach suffices to show NP $\neq$ P, one might ask why we do not highlight this version in Section 3. The reason is that this algorithmic task is not very naturally defined, since it has a unary input and refers to a uniform circuit family. We find the algorithmic tasks defined and studied in Section 3 more natural, in that an arbitrary circuit from the class $\mathfrak{C}$ is provided as input.

Next we tackle the generality question for our algorithmic approach to lower bounds for PSPACE. We first define a uniform version of the algorithmic approach.

▶ **Definition 37.** *Let $\mathfrak{C}$ be a circuit class. We say that there is space-efficient non-trivial sampling for the uniform version of* Dense $-$ $\mathfrak{C}$ $-$ SAT *if for every $k > 0$ and every* LOGTIME*-uniform sequence $\{C_n\}$ of $\mathfrak{C}$-circuits of size* poly(n) *on n variables such that C_n accepts at least $2/3$ fraction of inputs of length n, there is a probabilistic algorithm A which, given input 1^n, has space and randomness complexity at most n^d (for some constant d independent of the exponent in the size of C_n) and outputs some fixed satisfying assignment y to C_n with probability at least $n^k/2^n$.*

We require the notion of KS complexity and the corresponding meta-complexity problem MKSP.

▶ **Definition 38.** *The* KS *complexity of a string x is defined as follows, relative to some space-efficient universal Turing machine U.* $\mathsf{KS}(x)$ *is the minimum over $|p| + s$ such that $U(p, \epsilon)$ halts and outputs x using space at most s.*

Given a function $s : \mathbb{N} \to \mathbb{N}$, $\mathsf{MKSP}[s]$ *is the set of strings x such that* $\mathsf{KS}(x) \leqslant s(|x|)$. *We say that* $\mathsf{MCSP}[s]$ *is zero-error easy on average over the uniform distribution if there is a deterministic polynomial-time algorithm which, given input y, always outputs 0, 1 or '?'; always correctly classifies y with respect to* $\mathsf{MCSP}[s]$ *when it outputs a Boolean value; and outputs a non-'?' value with probability at least $1/poly(n)$ over $y \sim \{0,1\}^n$. We say that* $\mathsf{MCSP}[s]$ *is zero-error hard on average over the uniform distribution if it is not zero-error easy on average over the uniform distribution.*

We will use the following result which establishes that MKSP is PSPACE-hard even on average.

▶ **Theorem 39** ([6])**.** *If* $\mathsf{PSPACE} \neq \mathsf{BPP}$*, then for each constant $\delta > 0$,* $\mathsf{MKSP}[n^\delta]$ *is zero-error hard on average under the uniform distribution.*

Finally we are ready to show our result in the case of uniform lower bounds for PSPACE.

▶ **Theorem 40.** *Suppose that* E *requires exponential-size Boolean circuits. Then* $\mathsf{PSPACE} \neq \mathsf{P}$ *iff there is infinitely-often space-efficient non-trivial sampling for the uniform version of* $\mathsf{Dense} - \mathsf{Circuit} - \mathsf{SAT}$.

As an easy corollary of Theorem 40, we derive an *algorithmic characterization* of $\mathsf{PSPACE} \neq \mathsf{P}$. We show that separating PSPACE and P, which is a lower bound question, is equivalent to the existence of at least one of two kinds of algorithms: a sub-exponential time non-uniform algorithm for E that works on infinitely many input lengths, or a space-efficient non-trivial sampling algorithm for the unfiorm version of $\mathsf{Dense} - \mathsf{Circuit} - \mathsf{SAT}$.

The following is a re-statement of Theorem 4.

▶ **Corollary 41.** $\mathsf{PSPACE} \neq \mathsf{P}$ *iff* E *has circuits of size $2^{o(n)}$ infinitely often or there is infinitely-often space-efficient non-trivial sampling for the uniform version of* $\mathsf{Dense} - \mathsf{Circuit} - \mathsf{SAT}$.

5 Feasibility of the Approach

In this section, we argue for the feasibility of our approach. We first show that uniform versions of most super-polynomial circuit lower bounds for NP can be captured within the framework, and then that some of the best-known uniform lower bounds proved using diagonalization, such as the space hierarchy theorem and Allender's lower bound for the Permanent, can be reproved using our approach. Then we show how to prove a couple of new lower bounds: NP does not have uniform polynomial-size AC^0 circuits with a bottom layer of Mod m gates, for any composite m, nor uniform polynomial-size AC^0 circuits with a bottom layer of threshold gates.

5.1 Capturing Known Lower Bounds

Complexity theorists have had success proving super-polynomial circuit lower bounds for NP against a variety of weak circuit classes, such as AC^0 and $\mathsf{AC}^0[p]$ (for primes p). We observe that the proofs of these lower bounds imply efficient non-trivial sampling for the

corresponding circuit classes, and hence our framework applies. Of course our framework only yields *uniform* lower bounds, which are weaker than the non-uniform lower bounds already known for these classes. However our hope is that the framework might be useful even for stronger circuit classes where non-uniform lower bounds in NP are *not* known, and in order for this to be credible, the framework should at least apply in cases where lower bounds *are* known.

We use the fact that super-polynomial size lower bounds for NP against AC^0 and $\mathsf{AC}^0[p]$ yield cryptographic hitting set generators against these classes with non-trivial seed length.

▶ **Theorem 42** ([41, 4, 19, 30])**.** *There is a cryptographic hitting set generator with seed length* $\mathsf{polylog}(n)$ *useful against* AC^0 *and, for any prime* p*, a cryptographic hitting set generator with seed length* $n - \sqrt{n}$ *useful against* $\mathsf{AC}^0[p]$*.*

In fact the work of [41, 4, 19, 30] gives cryptographic pseudo-random generators rather then just cryptographic hitting-set generators, but the weaker hitting property satisfies for our application.

▶ **Corollary 43.** *There is efficient non-trivial sampling for the succinct versions of* $\mathsf{Dense} - \mathsf{AC}^0 - \mathsf{SAT}$ *and* $\mathsf{Dense} - \mathsf{AC}^0[p] - \mathsf{SAT}$*.*

The corollary follows from Theorem 42 by just sampling a random output of the generator, which can be done in fixed polynomial time in n independent of the size of the circuit on n bits for which we are solving the sampling problem. Note that by the hitting property, at least one of the outputs of the generator will satisfy the circuit, and each such output is sampled with probability $2^{-\ell(n)}$, where $\ell(n)$ is the seed length of the generator, which is non-trivial by Theorem 42.

We next show that versions of some of the classical results on uniform lower bounds in the literature, shown using direct or indirect diagonalization, can be shown using our framework. First, we consider versions of the space hierarchy theorem.

▶ **Theorem 44.** *Let* $\mathfrak{C}$ *be the class of branching programs of polynomial size. There is space-efficient non-trivial sampling for* $\mathsf{Dense} - \mathfrak{C} - \mathsf{SAT}$*.*

▶ **Corollary 45** ([51])**.** $\mathsf{PSPACE} \neq \mathsf{LOGSPACE}$*.*

Corollary 45 follows from Theorem 44 by applying Theorem 27, since the class of branching programs of polynomial size is polynomially simulatable.

We remark that the standard proof of the space hierarchy theorem is a fairly simple direct diagonalization, so the proof of Corollary 45 does not have any advantage in terms of simplicity. In addition, Corollary 45 does not give the tight parameters of the space hierarchy theorem, i.e., separating space S from space S' for any space-constructible bounds S, S' where $S = o(S')$. The reason that Theorem 27 does not give the tight space hierarchy is that tighter separations correspond to simulation of circuit classes $\mathfrak{C}$ that are not known to be polynomially simulatable, i.e., branching programs of super-polynomial size. However, the tight space hierarchy can be recovered using the *ideas* of the proof of Theorem 27, by defining and applying an appropriate space-bounded version of Kolmogorov complexity. We omit the details, as we do not see how to obtain a new result on space hierarchies using these ideas.

Next, we show how to rederive Allender's celebrated uniform lower bound for the Permanent using our approach [3]. To show the efficient non-trivial sampling required to derive this lower bound, we need a lemma about the evaluation of succinctly described threshold circuits. For convenience, we will work with Majority circuits instead, and then use the fact that uniform depth d threshold circuits can be simulated by uniform depth $d + 1$ Majority circuits.

For any positive integer d, define the language Succinct-Maj_d^0-Eval to be the set of pairs $< C, x >$, where C is a Maj_d circuit of size n encoding a Maj_d circuit C' on n variables, and x is an input of length n, satisfying the condition that $C'(x) = 1$.

▶ **Lemma 46.** *Let d be any positive integer. Succinct-Maj_d-Eval is in* CH.

▶ **Theorem 47.** *Let d be any positive integer.There is efficient non-trivial sampling with* CH *oracle for the succinct version of* $\mathsf{Dense} - \mathsf{Maj}_d - \mathsf{SAT}$.

▶ **Corollary 48** ([3])**.** Permanent *does not have* LOGTIME*-uniform* TC^0 *circuits of polynomial size.*

We explain briefly how the proofs of Corollary 45 and Corollary 48 differ from the standard proofs. The standard proof of the space hierarchy [51] combines two ingredients: (i) The existence of a space-efficient universal Turing machine U that can simulate any Turing machine M with at most a constant factor overhead in space, and (ii) The idea of directly diagonalizing against all Turing machines operating in a given space bound by mapping inputs x to Turing machines M_x in a surjective way and doing the opposite of M_x on x. The proof of Corollary 45 replaces the simulation ingredient (i) with the existence of a non-trivial space-efficient sampling algorithm, and the direct diagonalization part (ii) with a different diagonalization argument based on resource-bounded Kolmogorov complexity.

The standard proof of Allender's lower bound [3] is an indirect diagonalization argument with the following parts: (i) A time hierarchy theorem for threshold Turing machines proved in an analogous way to the space hierarchy theorem, with a simulation step and a direct diagonalization step, and (ii) An inductive argument showing that if the Permanent has small uniform threshold circuits, so does every level of the counting hierarchy; then combining (i) and (ii) to derive a contradiction. The proof of Corollary 48 still uses the ingredient (ii), but replaces the simulation step of (i) with a sampling step, and the direct diagonalition step with a diagonalization argument based on resource-bounded Kolmogorov complexity.

The broader point we wish to make is that our framework is capable of capturing both direct and indirect diagonalization arguments, since the sampling condition we require seems weaker than the simulation conditions in previous diagonalization arguments, and hence potentially capable of proving a broader class of lower bounds. This is related to Theorem 31, which shows that efficient non-trivial sampling exists unconditionally in a white box setting.

5.2 New Lower Bounds

Ideally, we would like to be able to use our new approach to attack frontier open problems in uniform circuit lower bounds, such as separating NP from LOGTIME-uniform ACC^0 and separating NP from LOGTIME-uniform TC^0_2. While we are unable to do this, we are able to show lower bounds in NP against interesting subclasses of these circuit classes, namely against LOGTIME-uniform AC^0 circuits with $\mathsf{Mod}m$ gates at the bottom (for an arbitrary positive integer m), and against LOGTIME-uniform AC^0 circuits with Thr gates at the bottom. To show lower bounds in NP against the non-uniform versions of these classes is a longstanding open problem (though we do know lower bounds in NQP [59, 40]), and to the best of our knowledge, lower bounds in NP against the uniform versions have also been open so far.

▶ **Theorem 49.** *Let d, m be any positive integers.* NP *does not have polynomial-size* LOGTIME*-uniform* $\mathsf{AC}^0_d \circ (\mathsf{Mod}m)$ *circuits.*

Examining the proof of Theorem 49, the only fact we used about the bottom layer of gates is that they can be evaluated in fixed polynomial time in n. Hence the same proof also gives the following result.

▶ **Theorem 50.** *Let d be any positive integer.* NP *does not have polynomial-size* LOGTIME*-uniform* $\mathsf{AC}^0_d \circ \mathsf{Thr}$ *circuits.*

Theorems 49 and 50 above together capture the content of Theorem 5 from the Introduction.

We note that the simulation arguments used for the non-trivial sampling in the proofs of Theorem 49 and Theorem 50 are fairly generic. This leads us to believe that there might be alternate proofs of these results using a more standard indirect diagonalization approach. However, these results already seem new, and exploiting the fact that we only need *sampling* rather than *simulation* to apply Theorem 21 might lead to even stronger lower bounds.

6 Future Work

We describe here some directions for future work.

The main direction is to develop new algorithmic ideas for the sampling problems we consider, and use these to prove new lower bounds. In particular, it would be interesting to explore if the ideas and techniques of [8] and [53] are useful here. A particular circuit class of interest is the class of quasi-polynomial size SYM^+ circuits, i.e., depth-two circuits with a top symmetric gate and polylogarithmic fan-in AND gates at the bottom. Efficient non-trivial sampling for the succinct version of this class would imply that NP does not have LOGTIME-uniform ACC^0 circuits of polynomial size.

Another question is whether there is an analogous approach to separating NP and PSPACE from probabilistic uniform classes. For example, are there natural sampling tasks or similar tasks such that efficient solutions imply $\mathsf{NP} \not\subseteq \mathsf{BPP}$?

While we provide one potential algorithmic approach to uniform lower bounds, there might be others. It would be interesting to look into this, especially if these other approaches are more feasible with the algorithmic techniques we have at present.

References

1. Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC '08)*, 2008. To appear.
2. Miklos Ajtai. Σ^1_1-formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.
3. Eric Allender. The permanent requires large uniform threshold circuits. *Chic. J. Theor. Comput. Sci.*, 1999, 1999.
4. Eric Allender. When worlds collide: Derandomization, lower bounds, and kolmogorov complexity. *Foundations of Software Technology and Theoretical Computer Science*, 21, 2001.
5. Eric Allender. The new complexity landscape around circuit minimization. In *Language and Automata Theory and Applications - 14th International Conference, LATA 2020*, volume 12038 of *Lecture Notes in Computer Science*, pages 3–16. Springer, 2020.
6. Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from random strings. *SIAM Journal on Computing*, 35(6):1467–1493, 2006.
7. Josh Alman and Lijie Chen. Efficient construction of rigid matrices using an NP oracle. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019*, pages 1034–1055. IEEE Computer Society, 2019.
8. Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in nc^0. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 166–175. IEEE Computer Society, 2004.

9 S. Arora and B. Barak. *Complexity Theory: A Modern Approach*. Cambridge University Press, Cambridge, 2009.

10 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3(4):307–318, 1993.

11 Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P =? NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

12 David Barrington, Neil Immerman, and Howard Straubing. On uniformity within NC^1. *Journal of Computer and System Sciences*, 41, 1990.

13 Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. Rigid matrices from rectangular pcps or: Hard claims have complex proofs. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020*, pages 858–869. IEEE, 2020.

14 Peter Bürgisser, Christian Ikenmeyer, and Greta Panova. No occurrence obstructions in geometric complexity theory. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016*, pages 386–395. IEEE Computer Society, 2016.

15 Lijie Chen. Non-deterministic quasi-polynomial time is average-case hard for ACC circuits. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019*, pages 1281–1304. IEEE Computer Society, 2019.

16 Lijie Chen, Xin Lyu, and R. Ryan Williams. Almost-everywhere circuit lower bounds from non-trivial derandomization. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020*, pages 1–12. IEEE, 2020.

17 Lijie Chen and Hanlin Ren. Strong average-case lower bounds from non-trivial derandomization. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1327–1334. ACM, 2020.

18 Stephen Cook. The complexity of theorem-proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

19 Bill Fefferman, Ronen Shaltiel, Christopher Umans, and Emanuele Viola. On beating the hybrid argument. *Theory Comput.*, 9:809–843, 2013.

20 Lance Fortnow. The status of the P versus NP problem. *Communications of the ACM*, 52(9):78–86, 2009.

21 Lance Fortnow and Adam R. Klivans. Efficient learning algorithms yield circuit lower bounds. *Journal of Computer and System Sciences*, 75(1):27–36, 2009.

22 Merrick Furst, James Saxe, and Michael Sipser. Parity, circuits, and the polynomial-time hierarchy. *Mathematical Systems Theory*, 17(1):13–27, April 1984.

23 Eran Gat and Shafi Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electron. Colloquium Comput. Complex.*, TR11-136, 2011.

24 Halley Goldberg, Valentine Kabanets, Zhenjian Lu, and Igor Carboni Oliveira. Probabilistic kolmogorov complexity with applications to average-case complexity. In *37th Computational Complexity Conference, CCC 2022, July 20-23, 2022, Philadelphia, PA, USA*, volume 234 of *LIPIcs*, pages 16:1–16:60. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

25 Johan Håstad. Almost optimal lower bounds for small depth circuits. In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing*, pages 6–20, 1986.

26 Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

27 Shuichi Hirahara. Non-black-box worst-case to average-case reductions within NP. In *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018*, pages 247–258. IEEE Computer Society, 2018.

28 Shuichi Hirahara. Unexpected hardness results for kolmogorov complexity under uniform reductions. In *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020*, pages 1038–1051. ACM, 2020.

29 Shuichi Hirahara. Average-case hardness of NP from exponential worst-case hardness assumptions. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing,*, pages 292–302. ACM, 2021.

30 Shuichi Hirahara and Rahul Santhanam. On the average-case complexity of MCSP and its variants. In Ryan O'Donnell, editor, *32nd Computational Complexity Conference, CCC 2017, July 6-9, 2017, Riga, Latvia*, volume 79 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

31 Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: Exponential time vs. probabilistic polynomial time. *Journal of Computer and System Sciences*, 65(4):672–694, 2002.

32 Russell Impagliazzo and Avi Wigderson. P = BPP if E requires exponential circuits: Derandomizing the XOR lemma. In *Proceedings of the 29th Annual ACM Symposium on the Theory of Computing*, pages 220–229, 1997.

33 Adam R. Klivans, Pravesh Kothari, and Igor Carboni Oliveira. Constructing hard functions using learning algorithms. In *Proceedings of the 28th Conference on Computational Complexity, CCC 2013*, pages 86–97. IEEE Computer Society, 2013.

34 Zhenjian Lu and Igor Carboni Oliveira. Theory and applications of probabilistic kolmogorov complexity. *Bull. EATCS*, 137, 2022.

35 Zhenjian Lu, Igor Carboni Oliveira, and Rahul Santhanam. Pseudodeterministic algorithms and the structure of probabilistic time. In *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 303–316. ACM, 2021.

36 Zhenjian Lu, Igor Carboni Oliveira, and Marius Zimand. Optimal coding theorems in time-bounded kolmogorov complexity. In *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 92:1–92:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

37 Ketan Mulmuley. On p vs np and geometric complexity theory: dedicated to sri ramakrishna. *Journal of the Association of Computing Machinery*, 58(2), 2011.

38 Ketan Mulmuley and Milind Sohoni. Geometric complexity theory I: an approach to the P vs. NP and related problems. *SIAM Journal on Computing*, 31(2):496–526, 2001.

39 Saburo Muroga. *Threshold logic and its applications*. Wiley, 1971.

40 Cody D. Murray and R. Ryan Williams. Circuit lower bounds for nondeterministic quasi-polytime from a new easy witness lemma. *SIAM J. Comput.*, 49(5), 2020.

41 Noam Nisan. Pseudorandom bits for constant depth circuits. *Combinatorica*, 11(1):63–70, 1991.

42 Noam Nisan and Avi Wigderson. Hardness vs randomness. *Journal of Computer and System Sciences*, 49(2):149–167, 1994.

43 Igor Carboni Oliveira. Randomness and intractability in kolmogorov complexity. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019*, volume 132 of *LIPIcs*, pages 32:1–32:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

44 Igor Carboni Oliveira and Rahul Santhanam. Conspiracies between learning algorithms, circuit lower bounds, and pseudorandomness. In *32nd Computational Complexity Conference, CCC 2017*, volume 79 of *LIPIcs*, pages 18:1–18:49. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

45 Ian Parberry and Georg Schnitger. Parallel computation with threshold functions. *J. Comput. Syst. Sci.*, 36(3):278–302, 1988.

46 Alexander Razborov. Lower bounds for the monotone complexity of some boolean functions. *Soviet Mathematics Doklady*, 31:354–357, 1985.

47 Alexander Razborov. Lower bounds on the size of bounded-depth networks over the complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

48 Alexander Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, 1997.

49 Rahul Santhanam. Pseudorandomness and the minimum circuit size problem. In *11th Innovations in Theoretical Computer Science Conference, ITCS 2020*, volume 151 of *LIPIcs*, pages 68:1–68:26. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

50 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings of the 19th Annual Symposium on Theory of Computing*, pages 77–82, 1987.

51 Richard Stearns, Juris Hartmanis, and Philip Lewis. Hierarchies of memory limited computations. In *Proceedings of the Sixth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 179–190. IEEE, 1965.

52 S. Toda. PP is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991.

53 Emanuele Viola. The complexity of distributions. *SIAM J. Comput.*, 41(1):191–218, 2012.

54 Klaus W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986.

55 R. Ryan Williams. Natural proofs versus derandomization. *SIAM J. Comput.*, 45(2):497–529, 2016.

56 Richard Ryan Williams. Limits on representing boolean functions by linear combinations of simple functions: Thresholds, relus, and low-degree polynomials. In *33rd Computational Complexity Conference, CCC 2018*, volume 102 of *LIPIcs*, pages 6:1–6:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.

57 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. In *Proceedings of the 42nd Annual ACM Symposium on Theory of Computing*, pages 231–240, 2010.

58 Ryan Williams. Non-uniform ACC circuit lower bounds. In *Proceedings of 26th Annual IEEE Conference on Computational Complexity*, pages 115–125, 2011.

59 Ryan Williams. New algorithms and lower bounds for circuits with linear threshold gates. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pages 194–202, 2014.

60 Andrew Chi-Chih Yao. Separating the polynomial-time hierarchy by oracles (preliminary version). In *26th Annual Symposium on Foundations of Computer Science*, pages 1–10. IEEE Computer Society, 1985.

Colourful TFNP and Propositional Proofs

Ben Davis
School of Computer Science, McGill University, Montreal, Canada

Robert Robere
School of Computer Science, McGill University, Montreal, Canada

Abstract

Recent work has shown that many of the standard TFNP classes – such as PLS, PPADS, PPAD, SOPL, and EOPL – have corresponding proof systems in propositional proof complexity, in the sense that a total search problem is in the class if and only if the totality of the problem can be efficiently proved by the corresponding proof system. We build on this line of work by studying *coloured* variants of these TFNP classes: C-PLS, C-PPADS, C-PPAD, C-SOPL, and C-EOPL. While C-PLS has been studied in the literature before, the coloured variants of the other classes are introduced here for the first time. We give a family of results showing that these coloured TFNP classes are natural objects of study, and that the correspondence between TFNP and natural propositional proof systems is not an exceptional phenomenon isolated to weak TFNP classes. Namely, we show that:

- Each of the classes C-PLS, C-PPADS, and C-SOPL have corresponding proof systems characterizing them. Specifically, the proof systems for these classes are obtained by adding depth to the formulas in the corresponding proof system for the uncoloured class. For instance, while it was previously known that PLS is characterized by bounded-width Resolution (i.e. depth 0.5 Frege), we prove that C-PLS is characterized by depth-1.5 Frege (Res(polylog(n))).
- The classes C-PPAD and C-EOPL coincide exactly with the uncoloured classes PPADS and SOPL, respectively. Thus, both of these classes also have corresponding proof systems: unary Sherali-Adams and Reversible Resolution, respectively.
- Finally, we prove a *coloured intersection theorem* for the coloured sink classes, showing C-PLS $\cap$ C-PPADS = C-SOPL, generalizing the intersection theorem PLS $\cap$ PPADS = SOPL. However, while it is known in the uncoloured world that PLS $\cap$ PPAD = EOPL = CLS, we prove that this equality *fails* in the coloured world in the black-box setting. More precisely, we show that there is an oracle O such that C-PLS$^O \cap$ C-PPAD$^O \supsetneq$ C-EOPL^O.

To prove our results, we introduce an abstract multivalued proof system – the *Blockwise Calculus* – which may be of independent interest.

2012 ACM Subject Classification Theory of computation → Proof complexity

Keywords and phrases oracle separations, TFNP, proof complexity, Res(k), lower bounds

Digital Object Identifier 10.4230/LIPIcs.CCC.2023.36

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2023/068/

Funding B.D. and R.R. supported by NSERC.

1 Introduction

1.1 Introduction to TFNP and Proof Complexity

This work continues a recent line of research relating the theory of *total* NP *search problems* [22, 27] to the theory of *propositional proof complexity*.[1] A total NP search problem is a search problem S satisfying:

[1] This paper is an extended abstract. Please see the full version at https://eccc.weizmann.ac.il/report/2023/068/.

- **Totality.** On every input x, some solution y with $|y| \leq |x|^{O(1)}$ is guaranteed to exist.
- **Efficient Certification.** Checking if y is a valid solution for x is polynomial-time computable.

The class TFNP contains all such search problems, and many important computational problems lie inside of this class – such as the problem of computing a Nash equilibrium of a bimatrix game, or the problem of computing a prime factor of a given number.

Since the initial study of TFNP it has been known that no problem in TFNP can be NP-Hard unless NP = coNP [26]. As a result, in order to understand the internal structure of TFNP, researchers have defined subclasses of TFNP based on polynomial-time reducibility to fixed total search problems [27]. For example, some of the most well-studied subclasses of TFNP can be defined by reductions to the following problems:

- PLS. Given a directed acyclic graph, output a sink node.
- PPAD. Given a directed graph with an unbalanced node (in-degree $\neq$ out-degree), output another unbalanced node.
- PPADS. Given a directed graph with a *negatively* unbalanced node (in-degree $<$ out-degree), output a *positively* unbalanced node (in-degree $>$ out-degree).

The theory of TFNP has been an extraordinary success in capturing the complexity of many computational problems that have avoided classification in other settings. For example, the class PPAD captures the complexity of computing a Nash Equilibrium [15] along with other important problems in economics [13, 11, 12].

Black-Box TFNP Classes and Propositional Proof Systems

An important caveat in the definitions of the above classes is in the input representation. It is clear that all of the above problems are computationally easy (i.e. inside of P), if we are given the directed graphs in some standard encoding like an adjacency list or an adjacency matrix. Instead, in the definitions of the TFNP classes we assume that the inputs are given *implicitly*. For instance, we can represent an (exponentially large) $O(1)$-degree directed graph G by a polynomial-size boolean circuit C that, when given a node $u \in V(G)$ as input, outputs the list of in- and out-neighbours of u. When described in this implicit encoding, we can no longer exhaustively search through the graph to find a solution to the search problem, but, when given a potential solution we can still verify its correctness in polynomial time.

Another natural way of implicitly representing an input to a total search problem is by using *black-box* (also called *query*) access, where the input is represented by an oracle. Following the earlier example, now the graph G would be represented by an oracle which receives a node $u \in V(G)$ as input and outputs the list of neighbours of u. For now, we informally define TFNP^{dt} as the class of total search problems where the inputs are represented as black-boxes in this way. The seminal work of Beame et al. [3] demonstrated that this model is closely related to *oracle separations* between the standard TFNP classes. In particular, if we have two black-box TFNP subclasses A^{dt} and B^{dt}, then a containment $\mathsf{A}^{dt} \subseteq \mathsf{B}^{dt}$ by a sufficiently uniform simulation implies that $\mathsf{A} \subseteq \mathsf{B}$ – since we can always simulate the black-box by evaluating the circuit – but if $\mathsf{A}^{dt} \not\subseteq \mathsf{B}^{dt}$ then there is an oracle O such that $\mathsf{A}^{O} \not\subseteq \mathsf{B}^{O}$ [3]. Beame et al. [3] used this strategy to construct oracle separations between many pairs of TFNP classes that were not previously separated.

Another major contribution of Beame et al. [3] was pioneering the use of *propositional proof complexity* in the study of TFNP classes. They showed that if a total search problem S lies in the (black-box) class $\mathsf{PPA}^{dt} \subseteq \mathsf{TFNP}^{dt}$, then a particular unsatisfiable CNF formula F_S associated with S has efficient refutations in the well-studied algebraic Nullstellensatz

proof system. By combining this with a lower bound against Nullstellensatz proofs refuting the pigeonhole principle PHP_n^{n+1} they provided the first oracle separation between the classes PPP^O and PPA^O. Proof complexity was also employed as a crucial tool by Buresh-Oppenheim and Morioka [8], who used it to unify previous oracle separations in black-box TFNP and also provide new results about the class PLS^{dt}.

Very recently, the relationships between TFNP^{dt} subclasses and propositional proof systems have been revisited [10, 21, 20, 9]. One surprising outcome of the emerging work is that: not only can proof complexity lower bounds be used to construct oracle separations (as in [3]) but, proof complexity lower bounds in fact turn out to be *equivalent* to these oracle separations! More formally, for many of the most well-studied TFNP classes A (e.g. $\mathsf{A} = \mathsf{PLS}$, PPAD, PPADS, $\mathsf{CLS} = \mathsf{EOPL}$, SOPL), there is a corresponding propositional proof system P_{A} such that the following relationship holds:

A total search problem S lies in the class A^{dt}
if and only if
The propositional encoding of the totality of S can be efficiently proved in P_{A}.

These new equivalences led to a number of new results in both the theory of propositional proof complexity and the theory of TFNP. In the theory of TFNP, for example, for each of the classes listed above, such propositional proof systems not only exist, but they are natural proof systems that have been well-studied in the proof complexity literature (cf. Figure 1). In [20] these new equivalences led to the proofs of oracle separations between the TFNP classes PLS and PPP as well as between UEOPL and EOPL, finally resolving all oracle separations between the classical TFNP classes. On the other hand, the breakthrough collapse $\mathsf{CLS} = \mathsf{PLS} \cap \mathsf{PPAD}$ [16] and its followups $\mathsf{EOPL} = \mathsf{PLS} \cap \mathsf{PPAD}$ and $\mathsf{SOPL} = \mathsf{PLS} \cap \mathsf{PPADS}$ [19] led to brand-new *intersection theorems* in proof complexity. In particular, there are natural proof systems – P_1, P_2, P_3 – such that a formula F has an efficient proof in P_1 *if and only if* F has efficient proofs in P_2 *and* efficient proofs in P_3. This is illustrated by the work of [20], which showed that the proof system *Reversible Resolution* – which is closely related to Max-SAT solving – is the intersection of the classical *Resolution* proof system and the *Sherali-Adams* proof system. Section 2 outlines the formal definitions of these proof systems.

Next Steps

In light of these results a number of open problems – both concrete and conceptual – remain, namely:

- Do *all* TFNP subclasses defined by a syntactic existence principle admit a characterization by a natural proof system? The recent work of Buss, Fleming, and Impagliazzo [9] constructs a Cook-Reckhow proof system for *every* black-box TFNP class, but, it is not clear if these proof systems are equivalent to standard systems occurring in the literature.
- If the above is not true, what is special about these "weak" TFNP classes that do admit characterizations by proof systems?
- Are the intersection theorems $\mathsf{CLS} = \mathsf{EOPL} = \mathsf{PLS} \cap \mathsf{PPAD}$ and $\mathsf{SOPL} = \mathsf{PLS} \cap \mathsf{PPADS}$ a unique phenomenon, or do other instances of intersection theorems exist? If so, do they imply other intersection results for proof complexity?
- Do other well-studied TFNP classes not depicted above that correspond to natural proof systems? (Note that many other well-studied TFNP subclasses – such as the classes PPP and UEOPL – and other classes corresponding to the weak pigeonhole principle or Ramsey's theorem are currently not known to admit nice characterizations by proof systems).

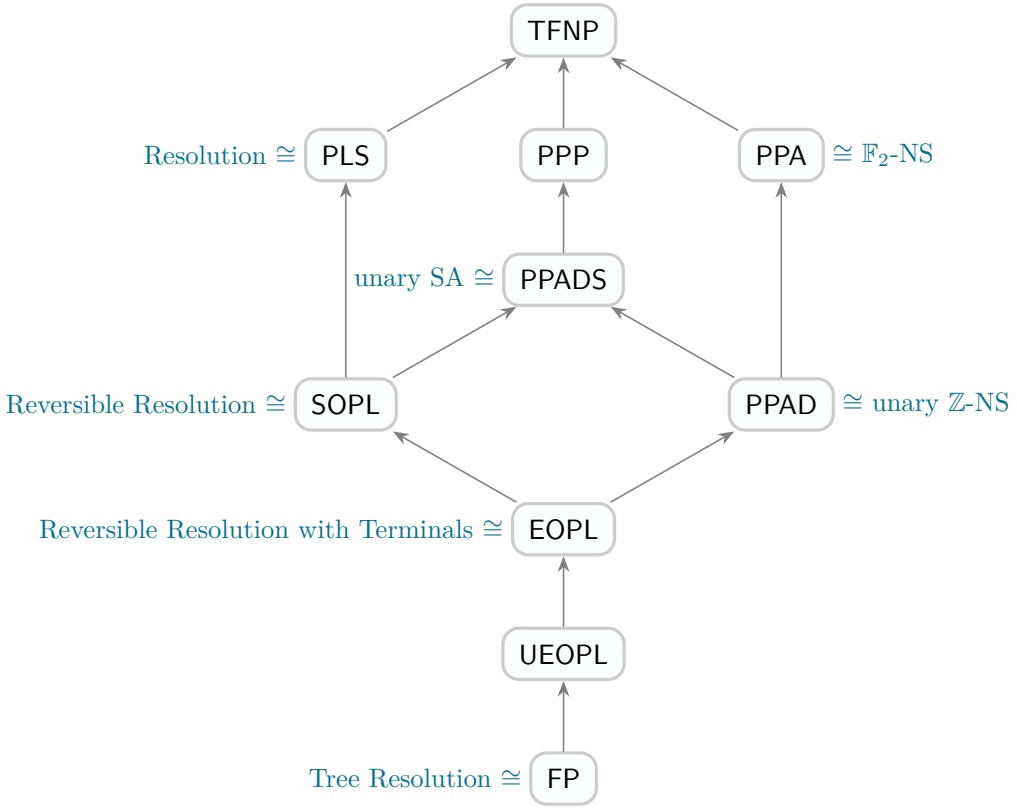


Figure 1 Class inclusion diagram for TFNP. An arrow A → B means A ⊆ B relative to all oracles. In the black-box model some classes can be captured using propositional proof systems, as indicated in blue. Above SA refers to the Sherali-Adams proof system [29], NS refers to the Nullstellensatz proof system [4], and "unary" refers to the fact that we measure size by the sum of all coefficients occurring in the proof.

1.2 Our Results

In this paper we introduce a new family of TFNP classes and demonstrate that they have natural corresponding propositional proof systems. Specifically, we consider a systematic way to generalize the TFNP classes PLS, PPAD, PPADS, EOPL, SOPL, obtaining their *coloured* generalizations C-PLS, C-PPAD, C-PPADS, C-EOPL, and C-SOPL. The formulas embodying the class C-PLS have previously been studied in proof complexity and bounded arithmetic, particularly in connection with *witnessing theorems* for the bounded arithmetic theory T^2_2 [24, 31]. For the other classes, however, the coloured variants are introduced and systematically studied here for the first time to the best of our knowledge. Before we discuss our results for these coloured classes, let us first describe to generalize a TFNP class to its coloured variant.

From Uncoloured to Coloured TFNP Classes

The key shared property between the classes PLS, PPADS, PPAD, EOPL, SOPL is the following: the input to each of these problems is a directed graph – enforced to be acyclic[2] in the case of PLS, EOPL, and SOPL – having distinguished source node s with at least one

[2] We can enforce acyclicity by adding in a decreasing potential function on the nodes of the graph, and requiring that edges must point from nodes of higher potential to nodes of lower potential.

outgoing edge. The goal of the search problem is to either output a *proper sink node* in the input graph (i.e. a sink node with at least one in-neighbour) or, in the case of PPAD and EOPL, one can also output a *proper source node* (i.e. a source node with at least one out-neighbour) other than the distinguished one[3].

In the coloured generalization of these problems, we receive a list of n *colours* $C_u \subseteq [n]$ for each node $u \in V(G)$ along with the directed graph G as input, and the solutions are updated as follows:

- Any proper source node *with a colour* is a solution,
- Any sink node *with no colour* (i.e. if $C_u = \emptyset$) is a solution, and
- A node u with an out-neighbour v is a solution if there is a colour $\lambda \in C_v$ such that $\lambda \notin C_u$.

To state the totality as an unsatisfiable system of constraints: the graph G has at least one proper source, all proper sources are colourless, all sinks have at least one colour, and colours propagate *backwards* across directed edges – if a node u has v as an out-neighbour then $C_v \subseteq C_u$. All of these constraints are obviously testable in polynomial time, with the possible exception of testing for a colourless sink. For this, we require that if a node is a sink node, then there is a polynomial-time function that points to some colour that is present at that sink. Note that knowing only the identity of a node, it is no longer simple to test whether it is colourless. This is unlike the analogous and easily-testable property in the uncoloured problems that a node has a successor, giving some intuition for the increased difficulty of the coloured problems. See Figure 2 for the hierarchy of these coloured problems and how they relate to classical TFNP classes, and Section 2.2 for formal definitions.

Statement of Results

Before stating our main results we require some formal definitions. A *query total search problem* is a sequence of relations $R_n \subseteq \{0,1\}^n \times O_n$, one for each $n \in \mathbb{N}$, such that $\forall x \in \{0,1\}^n \exists o \in O_n : (x, o) \in R_n$. We think of x as being provided to us via query access to its individual bits, and so an "efficient" algorithm would intuitively be provided by a $\mathsf{polylog}(n)$-depth decision tree solving the search problem. The search problem $\mathrm{R} = (R_n)_n$ is in TFNP^{dt} if, for each $o \in O_n$, there is a $\mathsf{polylog}(n)$-depth decision tree T_o such that $T_o(x) = 1$ iff $(x, o) \in R_n$. Furthermore, given a search problem R, we can define a corresponding subclass of TFNP^{dt}, denoted R^{dt}, obtained by taking all query total search problems that have low-depth decision-tree reductions to R (see Section 2.2 for the formal definition of a reduction in this model).

The canonical examples of total search problems in TFNP^{dt} come from low-width unsatisfiable CNF formulas. Any unsatisfiable CNF formula $F = C_1 \wedge \cdots \wedge C_m$ over variables $x_1, \ldots, x_n$ yields a closely related total search problem $S(F) \subseteq \{0,1\}^n \times [m]$: given an assignment x to the variables of F, output the index of a falsified clause of $F(x)$. Given a sequence of unsatisfiable CNF formulas $F = (F_n)_n$, the search problem $S(F) := (S(F_n))_n \in \mathsf{TFNP}^{dt}$ if and only if the width of (some unsatisfiable subformula of) F is $\mathsf{polylog}(n)$. Conversely, given any total search problem $R_n \subseteq \{0,1\}^n \times O_n$ we can define the unsatisfiable CNF formula $\bigwedge_{o \in O_n} \neg T_o$, where $\neg T_o$ is the encoding of the negation of the decision tree T_o as a CNF formula. It is easy to see that a query-efficient algorithm for R_n exists iff one exists for $S(\bigwedge_{o \in O_n} \neg T_o)$, and thus we can focus on search problems of the form $S(F) \in \mathsf{TFNP}^{dt}$ without loss of generality.

[3] For the interested reader, we note that this similarity was identified and formalized as a general Grid search problem in [19].

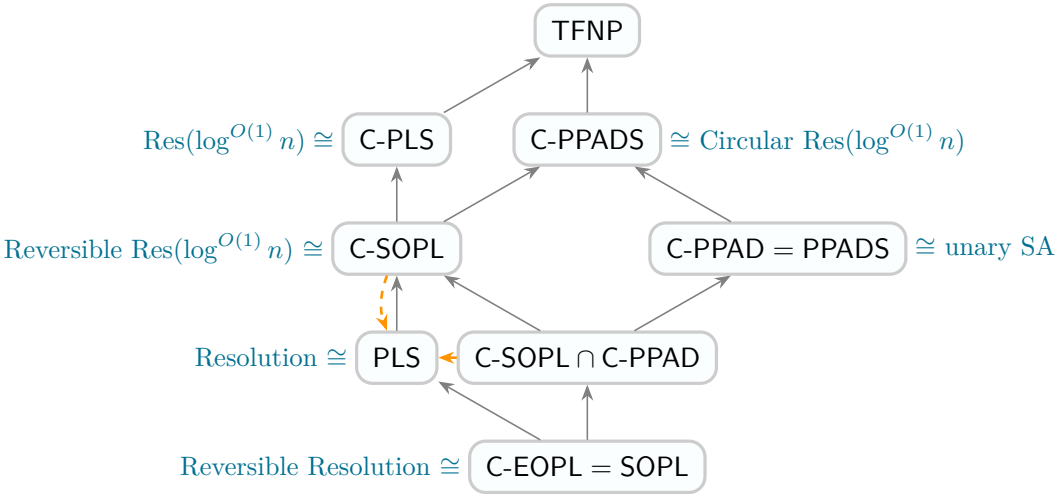


Figure 2 The coloured TFNP classes and the corresponding proof systems considered in this paper. A solid line from A to B indicates that A is contained in B relative to every oracle, while a red dashed line means A is not contained in B relative to some oracle.

Given these definitions we can state our main results, summarized in Figure 2. First, we show that *every* coloured class defined above has an equivalent propositional proof system. Moreover, these proof systems are closely related to the proof systems for the uncoloured variants. Given a black-box TFNP class A^{dt} and a proof system P, we write $\mathsf{A}^{dt} \cong P$ if the following holds: for every sequence of unsatisfiable CNFs $F = (F_n)_n \in \mathsf{TFNP}^{dt}$, $S(F) \in \mathsf{A}^{dt}$ if and only if there is a $n^{\mathsf{polylog}(n)}$-size, $\mathsf{polylog}(n)$-degree refutation of F_n in P.

▶ **Theorem 1.1.** *The following equivalences between* TFNP^{dt} *classes and proof systems hold:*

- $\mathsf{C\text{-}PLS}^{dt} \cong \mathrm{Res}(\mathsf{polylog}(n))$,
- $\mathsf{C\text{-}PPADS}^{dt} \cong \mathrm{CircRes}(\mathsf{polylog}(n))$,
- $\mathsf{C\text{-}SOPL}^{dt} \cong \mathrm{RevRes}(\mathsf{polylog}(n))$.

In the above theorem, $\mathrm{Res}(\mathsf{polylog}(n))$ is the extension of Resolution to DNF formulas with $\mathsf{polylog}(n)$-width conjunctions on the bottom level (see e.g. [23, 28, 1, 18]). The system $\mathrm{Res}(n)$ is equivalent to depth-2 Frege, and thus $\mathrm{Res}(\mathsf{polylog}(n))$ sits between Resolution and depth-2 Frege in power. The *Reversible* $\mathrm{Res}(\mathsf{polylog}(n))$ system (denoted $\mathrm{RevRes}(\mathsf{polylog}(n))$) is the natural extension of reversible Resolution to DNF formulas. The *Circular* $\mathrm{Res}(\mathsf{polylog}(n))$ system is exactly the *higher-depth analogue* of Sherali-Adams which is allowed to "operate" on DNF formulas. It is obtained by augmenting the $\mathrm{RevRes}(\mathsf{polylog}(n))$ system with a new rule that allows us to introduce any DNF formula D for free, as long as we (eventually) derive a copy of D later in the proof to make up for the introduced copy. This notion of a "circular, yet sound" proof was introduced by [2] in the setting of Resolution, where it was observed that Circular Resolution is exactly the same as Sherali-Adams. It is quite remarkable that augmenting the three TFNP classes PLS, PPADS, and SOPL with colours yields new natural classes whose corresponding proof systems are simply the proof systems for the uncoloured class where the lines have one greater depth!

Our second main result deals with the coloured "source-or-sink" classes C-PPAD and C-EOPL. Here, we show an *a-priori* unexpected collapse actually occurs: the *coloured* source-or-sink classes are exactly the same as the *uncoloured* sink classes. Consequentially, we obtain propositional proof systems equivalent to these TFNP^{dt} classes by relying on earlier work [20].

▶ **Theorem 1.2.** *The collapses* C-EOPL = SOPL *and* C-PPAD = PPADS *hold. As a consequence,* C-EOPL$^{dt} \cong$ *Reversible Resolution and* C-PPAD$^{dt} \cong$ *Unary Sherali-Adams.*

In order to prove the above collapses between TFNP classes, we actually proceed entirely through proof complexity. That is, we exploit the prior results SOPL$^{dt} \cong$ RevRes, as well as PPADS$^{dt} \cong$ uSA [20], and give *refutations* of the defining principles of C-EOPL and C-PPAD in the corresponding proof systems. By applying the known characterization results we then obtain the collapses between these TFNP classes immediately. While we do not see how to prove these collapses directly in the language of TFNP, this only further necessitates studying the relationship between the two areas.

Our third major result is a generalization of the intersection theorem SOPL = PLS∩PPADS to the coloured setting. This proves, as an immediate consequence, that the proof system RevRes(k) is the "intersection" of Res(k) and CircRes(k).

▶ **Theorem 1.3.** C-SOPL^{dt} = C-PLS^{dt} ∩ C-PPADS^{dt}.

▶ **Corollary 1.4.** *For any* polylog(n)*-width CNF formula F on n variables, there is a $n^{\text{polylog}(n)}$-size* RevRes(polylog(n)) *refutation of F if and only if there is a $n^{\text{polylog}(n)}$-size* Res(polylog(n)) *refutation of F and a $n^{\text{polylog}(n)}$-size* CircRes(polylog(n)) *refutation of F.*

Finally, and quite surprisingly, we show that the intersection theorem C-EOPL = C-PLS ∩ C-PPAD actually *fails* relative to an oracle. In other words, there is an oracle O such that C-EOPL$^O \neq$ C-PLS$^O \cap$ C-PPAD^O.

▶ **Theorem 1.5.** C-EOPL$^{dt} \subsetneq$ C-PLS$^{dt} \cap$ C-PPAD^{dt}*, or, equivalently* SOPL$^{dt} \subsetneq$ C-PLS$^{dt} \cap$ PPADS^{dt}.

We show this theorem as follows. Since C-EOPL^{dt} = SOPL$^{dt} \subseteq$ PLS^{dt}, the intersection theorem would imply that

$$\text{C-PLS}^{dt} \cap \text{C-PPAD}^{dt} = \text{C-PLS}^{dt} \cap \text{PPADS}^{dt} \subseteq \text{PLS}^{dt}.$$

However, we can actually show the (even stronger) separation that C-SOPL$^{dt} \cap$ PPADS$^{dt} \not\subseteq$ PLS^{dt}. The fact that PPADS$^{dt} \not\subseteq$ PLS^{dt} follows from [20], and we can prove directly that C-SOPL$^{dt} \not\subseteq$ PLS^{dt}. We then show that for PLS^{dt}, one can *combine* the adversary arguments from the previous two separations to create an adversary for PPADS$^{dt} \cap$ C-SOPL^{dt}. The combination of adversaries holds generically, and shows that PLS^{dt} is itself *not* a non-trivial intersection class (we discuss this more in the full version of the paper.) Taken together, our results paint a intriguing picture for how the coloured TFNP classes relate to the uncoloured classes.

The Blockwise Calculus

The results that we prove in this paper have the unfortunate property of becoming quite proof-theoretically technical when trying to proceed directly. Our primary technical innovation – when compared to the recent work between TFNP and proof complexity – is the use of *multivalued logic* to simplify these arugments. In particular, we found it useful to abstract out a generalized calculus – called the *Blockwise Calculus* – in which to implement our proofs. One can think of the Blockwise Calculus as the natural extension of the Resolution proof system to multivalued variables. In Section 3 we define the Blockwise Calculus and its Reversible and Circular variants, as well as discuss its basic properties. In particular, we show how to translate refutations in the Blockwise Calculus and its variants *automatically* into refutations in Resolution, Res(k), and their variants.

Open Problems

In general this work suggests that further investigation of the connection between TFNP subclasses and propositional proof complexity is an avenue ripe for exploration.

- As previously outlined, Atserias and Lauria showed that Sherali-Adams is polynomially equivalent to the Circular Resolution proof system [2]. Is there an analogue of this result for Circular Res(k)? That is, is there a natural semi-algebraic proof system that generalizes Sherali-Adams and is polynomially equivalent to Circular Res(k)?
- It was recently shown that Resolution does not polynomially-simulate unary Sherali-Adams and *vice-versa* [20]. Can we prove similar separations between Res(k) and CircRes(k)? Note that one direction of this separation is already known: Res(k) cannot simulate CircRes(k) as the retraction Pigeonhole Principle is easy for CircRes(k) [14] but hard for Res(k) [28].
- What combinatorial principles capture even higher-depth proof systems? We note that some principles (e.g. the *Game Induction principles*) are known using translations from bounded arithmetic [5, 30].

Paper Organization

The rest of the paper proceeds as follows. In Section 2 we introduce the formal definitions of the propositional proof systems and TFNP subclasses that we consider. In Section 3 we define the Blockwise Calculus and its variants, as well as prove our main technical theorems relating the Blockwise Calculus to the boolean proof systems introduced in Section 2. We refer to the full version of the paper for proofs of our new containment and separation results, respectively.

2 TFNP Classes and Propositional Proof Systems

2.1 Propositional Proof Systems

In this section we recall the definitions of some of the standard proof systems considered in this paper. First, we recall the simplest proof system, *Resolution*, and its variant *Reversible Resolution* [20]. The Reversible Resolution variant (and, in particular, the "reversible" rules presented below) were first introduced in the context of MaxSAT solving [7, 25, 17].

▶ **Definition 2.1.** *Let F be a CNF formula and let C be a clause. A* Resolution proof *of C from F is given by a sequence of clauses $C_1, C_2, \ldots, C_s = C$ where the sequence is generated as follows. Starting from the empty sequence we either choose a clause from F to append to the sequence, or, we choose earlier clauses in the sequence and apply one of the proof rules depicted below to generate new clauses to append to the sequence.*

$$\frac{C \vee \ell \quad C \vee \overline{\ell}}{C} \ (\textbf{\textit{Resolution}}) \qquad \frac{C}{C \vee \ell \quad C \vee \overline{\ell}} \ (\textbf{\textit{Reverse Resolution}})$$

The proof is a refutation *if $C = \bot$. The* length *of the proof is s, the number of clauses, and the* width *of the proof is the size of the widest clause in the proof. Finally, the proof is a* Reversible Resolution *proof if every clause is used as the hypothesis of at most one proof rule.*

We will also use *Sherali-Adams proofs*, which are one of the basic semi-algebraic proof systems studied in the literature. In particular we need its *unary* variant.

▶ **Definition 2.2.** *If* $C = \bigwedge_{i\in S} x_i \vee \bigwedge_{j\in T} \overline{x}_j$ *is a conjunction then we let* $p(C) := \prod_{i\in S} x_i \prod_{j\in T}(1-x_j)$ *denote the encoding of* C *as a real polynomial. A* conical junta *is a non-negative combination of conjunctions* $\sum_\lambda \lambda p(C)$ *where all coefficients are positive integers. A* Sherali-Adams refutation *of a CNF formula* $F = C_1 \wedge \cdots \wedge C_m$ *is given by a set of polynomials* $p_1, ..., p_m$ *and a conical junta* $\mathcal{J}$ *such that:*

$$\sum_{i=1}^{m} p_i \cdot p(\overline{C}_i) + \mathcal{J} = -1,$$

where all polynomial arithmetic is performed modulo the ideal generated by $\langle x_i^2 - x_i\rangle_{i=1}^n$*. The* unary size *of the refutation is the sum of all coefficients of all monomials in the expression above (after expansion), and the* degree *of the proof is the maximum degree of any monomial in the expanded expression above. We write* uSA *to denote the Sherali-Adams system where we measure size by unary size.*

The main focus of the present work is the higher-depth analogue of Resolution, known as Res(k), which operates on low-width DNF formulas. We consider three different variants of the Res(k) system (the *standard*, *reversible*, and *circular* variants), and for the sake of uniformity define them all using the same proof rules (cf. Figure 3).

$\wedge$-Introduction $\dfrac{D \vee A \qquad D \vee B}{D \vee (A \wedge B) \quad D \vee A \vee B}$ **Cut** $\dfrac{D \vee A \qquad D \vee \overline{A}}{D}$

Reverse Cut $\dfrac{D}{D \vee A \quad D \vee \overline{A}}$ **Axiom Introduction** $\dfrac{}{\ell \vee \overline{\ell}}$

Figure 3 The Res(k) Proof Rules. Above D is any DNF formula, A is a conjunction of boolean literals, ℓ is a boolean literal, and we use the convention that $\overline{A} = \bigvee_{\ell\in A} \overline{\ell}$.

▶ **Definition 2.3.** *Let* F *be a CNF formula, let* G *be a DNF formula, and let* k *be a positive integer. A* Res(k) proof *of* G *from* F *is a sequence of* k*-DNF formulas* $D_1, ..., D_s = G$ *where the sequence is generated as follows: starting from the empty sequence we either choose a clause from* F *to append to the sequence (interpreted as a width-*1 *DNF), or, we choose earlier DNFs in the sequence and apply any* Res(k)*-proof rule (cf. Figure 3) to generate new DNFs and append them to the sequence. The proof is a* refutation *of* F *if* $G = \bot$*, the empty disjunction. The* size *of the proof is* $\sum_{i=1}^{s} |D_i|$*, where* $|D_i|$ *represents the size of each DNF. The proof is* reversible *(or a* RevRes(k) proof*) if every DNF is used as a hypothesis of at most one proof rule.*

We now define *Unary Circular* Res(k) (or uCircRes(k)) proofs, which are a generalization of Res(k) in which the proofs can have cycles. As discussed in the introduction this is the higher-depth analogue of Sherali-Adams [6]. To define it we must introduce one additional proof rule called DNF Creation, defined next, that allows to create any DNF D in one proof step. While this rule is (obviously) not sound by itself, it turns out that one can make a sound proof system as long as we require that the proof eventually *derives* at least as many copies of D from other proof rules than were created by using the DNF Creation rule, and strictly more if D is the clause we wish to prove (cf. [2]).

$\dfrac{}{D}$ (**DNF Creation**)

▶ **Definition 2.4.** *Let F be a CNF formula. A* Unary Circular Res(k) proof *of a DNF G from F is a sequence of DNFs $D_1, D_2, \ldots, D_s = G$ that is generated as follows: starting from the empty sequence we either choose a clause C from F and append it to the sequence, we apply the DNF Creation rule to generate a new DNF and add it to the list, or we choose earlier DNFs in the sequence and apply a* Res(k) *proof rule to generate new DNFs and append them to the sequence. In addition, we make the following stipulations: each DNF D_i in the sequence is used as the hypothesis of at most one* Res(k) *rule, and every DNF D appearing in the proof is derived as the output of some proof rule at least as many times as it is created using DNF Creation, except the conclusion G which must be derived strictly more times than it is created with DNF-Creation. The* size *of the proof is $\sum_{i=1}^{s} |D_i|$. If $G = \bot$ then we call this a* uCircRes(k) *refutation of F.*

Both Res(k) and CircRes(k) can efficiently simulate RevRes(k), since RevRes(k) is a restriction of both systems – of the first system because of the fanout restriction, and of the second system because of its inability to apply DNF Creation.

2.2 Search classes

In this section we define the relevant background for TFNP. We follow the treatment of black-box TFNP used by [20].

▶ **Definition 2.5.** *A* total (query) search problem *is a sequence of relations $R = \{R_n \subseteq \{0,1\}^n \times O_n\}$, where O_n are finite sets, such that for all $x \in \{0,1\}^n$ there is an $o \in O_n$ so that $(x, o) \in R_n$. A total search problem R is in TFNP^{dt} if for each $o \in O_n$ there is a decision tree T_o with depth $\mathsf{poly}(\log n)$ such that for every $x \in \{0,1\}^n$, $T_o(x) = 1$ iff $(x, o) \in R$.*

As discussed in the introduction the canonical problems in TFNP^{dt} are the *false clause search problems* associated with an unsatisfiable $\mathsf{polylog}(n)$-width CNF formula $F = C_1 \wedge \cdots \wedge C_m$ defined as $S(F) \subseteq \{0,1\}^n \times [m]$ with $(x, i) \in S(F)$ iff $C_i(x) = 0$. Every problem in TFNP^{dt} is equivalent to $S(F)$ for some $\mathsf{polylog}(n)$-width CNF formula.

▶ **Definition 2.6.** *Let $R \subseteq \{0,1\}^n \times O$ and $S \subseteq \{0,1\}^m \times O'$ be total search problems. An S-*formulation of *R is a decision-tree reduction $(f_i, g_o)_{i \in [m], o \in O'}$ from R to S. Formally, for each $i \in [m]$ and $o \in O'$ there are functions $f_i \colon \{0,1\}^n \to \{0,1\}$ and $g_o \colon \{0,1\}^n \to O$ such that*

$$(x, g_o(x)) \in R \iff (f(x), o) \in S$$

where $f(x) \in \{0,1\}^m$ is the string whose i-th bit is $f_i(x)$. The depth *of the reduction is*

$$d \coloneqq \max\big(\{D(f_i) : i \in [m]\} \cup \{D(g_o) : o \in O'\}\big),$$

where $D(h)$ denotes the decision-tree depth of h. The size *of the reduction is m, the number of input bits to S. The* complexity *of the reduction is $\log m + d$. We write $S^{dt}(R)$ to denote the minimum complexity of an S-formulation of R.*

We extend these notations to sequences in the natural way. If R is a single search problem and $S = (S_m)$ is a sequence of search problems, then we denote by $S^{dt}(R)$ the minimum of $S^{dt}_m(R)$ over all m. If $R = (R_n)$ is also a sequence, then we denote by $S^{dt}(R)$ the function $n \mapsto S^{dt}(R_n)$.

Using the previous definition we can now define complexity classes of total search problems via reductions. For total search problems $\mathrm{R} = (R_n), \mathrm{S} = (S_n)$, we write

$$\mathrm{S}^{dt} \coloneqq \{\mathrm{R} : \mathrm{S}^{dt}(\mathrm{R}) = \mathsf{polylog}(n)\}.$$

Coloured TFNP Classes

With the definition of reductions established, we can define the search problems characterizing our coloured TFNP classes. We define the notation $[n]_0 := [n] \cup \{0\}$.

▶ **Definition 2.7 (Coloured Sink-of-Dag).** C-SoD$_n$ *is a total search problem defined on an* $n \times n$ *grid of nodes, where* $(1,1)$ *is a special distinguished node. As input, we receive the following parameters for each node* $(i,j) \in [n] \times [n]$*:*

- *An index* $s_{i,j} \in [n]_0$*, indicating that the successor of* (i,j) *is* $(i+1, s_{i,j})$*, or if* $s_{i,j} = 0$*, that* (i,j) *is a leaf.*
- *An indicator* $c_{i,j,\lambda} \in \{0,1\}$ $\forall \lambda \in [n]$*, indicating the presence of colours at each grid node*
- *An index* $e_{i,j} \in [n]$*, indexing a colour at each node*

Here the index $e_{i,j}$ *is used to ensure that sinks can be efficiently verified to have a colour.*

Any node on the final layer or any node with successor 0 *is called a leaf and the node* $(1,1)$ *is called the distinguished source. If the set of colours at each node contains the set of colours at its successor node, and there is at least one colour at each leaf, then clearly there must be at least one colour at the source node. The goal of the search problem is to find a witness of this fact. Formally, a solution to the* C-SoD$_n$ *search problem is*

- $((i,j),\lambda)$ *if* $s_{i,j} = k$*,* $c_{i+1,k,\lambda} = 1$*, and* $c_{i,j,\lambda} = 0$ *for some* k*.* (colour propagation)
- $((1,1),\lambda)$ *if* $c_{1,1,\lambda} = 1$ (distinguished source should be colourless)
- $((i,j),\lambda)$ *if* (i,j) *is a leaf,* $e_{i,j} = \lambda$*, and* $c_{i,j,\lambda} = 0$ (sinks should have a colour)

▶ **Definition 2.8 (Coloured Sink- and End-of-Line).** C-SoL$_n$ *is a search problem defined on a set of* n *nodes, denoted* $[n-1]_0$*, distinguishing the node* 0*. We define a graph on these nodes using the following parameters for each node* $u \in [n-1]_0$*:*

- *An index* $s_u \in [n-1]_0$ *indexing the successor of* u*.*
- *An index* $p_u \in [n-1]_0$ *indexing the predecessor of* u*.*
- *An indicator* $c_{u,\lambda} \in \{0,1\}$ *for each* $\lambda \in [n]$*, indicating the presence of the colour* λ *at* u*.*
- *An index* $e_u \in [n]$*, indexing a distinguished colour at each node.*

We define a graph G *on* $[n]$ *by including an edge* (u,v) *if and only if* $s_u = v$ *and* $p_v = u$*. Again, if the set of colours at each node contains that at its successor, and there is at least one colour at each sink, then each source must contain at least one colour. The goal of the search problem is to find a witness of this. A pair* (u,λ) *is then a solution to an instance of* C-SoL$_n$ *if:*

- $s_u = v$*,* $p_v = u$*,* $c_{v,\lambda} = 1$ *and* $c_{u,\lambda} = 0$ *for some node* $v \neq u$ (colour propagation)
- $u = 0$ *and* $c_{0,\lambda} = 1$ (distinguished source should be colourless)
- u *is a sink node,* $e_u = \lambda$*, and* $c_{u,\lambda} = 0$ (sinks should have a colour)

The C-EoL$_n$ *problem is obtained by adding the following solutions to the* C-SoL$_n$ *problem:*

- u *is a source node and* $c_{u,\lambda} = 1$ (sources should be colourless)

▶ **Definition 2.9 (Coloured Sink- and End-of-Potential-Line).** *The* C-SoPL$_n$ *and* C-EoPL$_n$ *problems are search problems combining the constraints of* C-SoD *and* C-EoL*. As with* C-SoD *they are defined on an* $n \times n$ *grid. We have the following parameters for each node* $(i,j) \in [n] \times [n]$*:*

- *An index* $s_{i,j} \in [n-1]_0$ *indicating that the successor of* (i,j) *is* $(i+1, s_{i,j})$*.*
- *An index* $p_{i,j} \in [n-1]_0$ *indicating that the predecessor of* (i,j) *is* $(i-1, p_{i,j})$*.*
- *An indicator* $c_{i,j,\lambda} \in \{0,1\}$ *for each* $\lambda \in [n]$ *indicating the presence of the colour* λ *at* (i,j)*.*
- *An index* $e_{i,j} \in [n]$ *indexing a distinguished colour at each node.*

As with C-SoL *and* C-EoL *we define a graph* G *on* $[n] \times [n]$ *by including an edge* $((i,j),(i+1,k))$ *if and only if* $s_{i,j} = k$ *and* $p_{i+1,k} = j$*. The solutions are then defined exactly as for* C-SoL$_n$ *and* C-EoL$_n$ *adapted to the* $n \times n$ *grid.*

We denote the TFNP^{dt} classes obtained by taking formulations of the above problems in San-Serif font, e.g. $\mathsf{C\text{-}SOPL}^{dt} = \text{C-SoPL}^{dt}$.

3 The Blockwise Calculus

3.1 Multivalued CNFs and Blockwise Calculus Proofs

The proof-theoretic results in this paper have the unfortunate property of becoming technical when proved directly in the boolean proof systems defined in the previous section. To aid exposition we have found it useful to abstract out a generalized calculus – the *Blockwise Calculus* – to phrase our proofs in. In this section we introduce the Blockwise Calculus and prove our main technical results illustrating its relationship with the proof systems introduced in the previous section. Intuitively the Blockwise Calculus is the extension of Resolution to variables in a wider range than $\{0,1\}$.

▶ **Definition 3.1.** *A* multivalued variable *is a pair* (x,n) *where* x *is a formal variable and* $n \in \mathbb{N}$ *is a positive integer representing the range* $[n-1]_0$ *that the variable* x *can take values in. We will suppress the range parameter* n *when it is obvious from context. An* atom *is an expression of the form* $[\![x \neq i]\!]$ *where* $i \in [n-1]_0$ *is an element of the range. Given an* $[n-1]_0$*-assignment to* x *the atom evaluates to true iff the inequality inside the atom is satisfied. A* clause *is a disjunction* ($\vee$) *of atoms, where each variable in the clause can be quantified over its own range. The* width *of a clause* C *is the number of atoms in it.*

Using multivalued variables we can introduce the notion of a multivalued CNF formula.

▶ **Definition 3.2.** *Let* $(x_1,r_1),(x_2,r_2),\ldots,(x_n,r_n)$ *be a collection of multivalued variables. A* multivalued CNF formula $F = C_1 \wedge \cdots \wedge C_m$ *over these variables is a conjunction of clauses of atoms over the same variables. We say that* F *is* unsatisfiable *if there is no assignment of each variable to their respective ranges such that the resulting CNF is satisfied, and define the corresponding search problem* $S(F) \subseteq [r_1-1]_0 \times \cdots \times [r_n-1]_0 \times [m]$ *in the natural way: given a multivalued assignment to the corresponding variables, output a false clause of* F*.*

While the Blockwise Calculus operates on multivalued CNF formulas, we ultimately want to convert everything back to refutations in boolean logic. For this, we introduce the *booleanization* of a multivalued CNF, which is obtained by encoding each multivalued variable (x,r) in binary.

▶ **Definition 3.3.** *Let* (x_i,r_i) *for* $i \in [n]$ *be a collection of multivalued variables, and let* $F = \bigwedge_{i=1}^{m} C_i$ *be a multivalued CNF formula over these variables. The* booleanization *of* F *is the following CNF formula* F_{bool}*. For each variable* (x_i,r_i) *we introduce* $t_i := \lceil \log r_i \rceil$ *boolean variables in a block, denoted* $\vec{x}_i := x_{i,1} \ldots x_{i,t_i} \in \{0,1\}^{t_i}$*, encoding the value of the variable* x_i *in binary. Then, for each clause in* F *we substitute each occurrence of an atom* $[\![x_i \neq k]\!]$ *with the disjunction on the variables* $\vec{x}_i$ *that is false exactly when* $x_i = k$*. Finally, for each* $i \in [n]$ *and each value* $\ell \in [2^{t_i}]$ *with* $\ell \geq r_i$*, we add a clause to* F_{bool} *over the variables* $\vec{x}_i$ *encoding that* $x_i \neq \ell$*.*

For each of the search problems defined in the previous section, there is a natural multivalued encoding of that search problem as an unsatisfiable multivalued CNF formula (cf. Section 3.2). Our current focus is to define the Blockwise Calculus and its variants. The rules of the Blockwise Calculus are shared among the three systems and defined in Figure 4, where C is a multivalued clause and (x, r) is a multivalued variable.

$$\textbf{(Reverse Cut)} \quad \frac{C}{C \vee [\![x \neq 0]\!] \quad C \vee [\![x \neq 1]\!] \cdots C \vee [\![x \neq r-1]\!]}$$

$$\textbf{(Cut)} \quad \frac{C \vee [\![x \neq 0]\!] \quad C \vee [\![x \neq 1]\!] \cdots C \vee [\![x \neq r-1]\!]}{C}$$

Figure 4 Proof Rules for the Blockwise Calculus.

▶ **Definition 3.4.** *Let F be a multivalued CNF formula and let C be a clause. A* Blockwise Calculus proof *of C from F is a sequence of clauses $C_1, C_2, \ldots, C_s = C$ where the sequence is generated as follows. Starting from the empty sequence we either choose a clause from F to append to the sequence, or, we choose earlier clauses in the sequence and apply one of the Blockwise Calculus proof rules (cf. Figure 4) to generate new clauses and append them to the sequence. The* length *of the proof is s, the number of clauses, and the* width *of the proof is the size of the largest clause in the proof. The proof is a* refutation *if $C = \bot$, the empty clause. Finally, the proof is a* Reversible Blockwise Calculus proof *if every clause is used as the hypothesis of at most one proof rule.*

As in the case of Res(k), we can also introduce the *Circular* variant of Blockwise Calculus. The analogous rule we need to introduce is the following, for any multivalued clause C:

$$\frac{}{C} \quad \textbf{(Clause Creation)}$$

▶ **Definition 3.5.** *Let F be a multivalued CNF formula and let C be a clause. A* Circular Blockwise Calculus proof *of C from F is a sequence of clauses $C_1, C_2, \ldots, C_s = C$ where the sequence is generated as follows. Starting from the empty sequence we can either choose a clause from F and append it to the end of the sequence, apply the Clause Creation rule to create an arbitrary clause C and append it to the sequence, or choose earlier clauses in the sequence and apply a Blockwise Calculus rule to generate new clauses and append them to the sequence. In addition, we make the following stipulations: each clause C_i in the sequence is used as the hypothesis of at most one Blockwise Calculus rule, and every clause C appearing in the proof is derived as the output of some proof rule more times than it is created using the Clause Creation rule. The length of this proof is s and the width of the proof is the maximum width of any clause C in the proof. The proof is a* refutation *if $C = \bot$.*

Similarly to the Res(k) systems, it is easy to see that Reversible Blockwise Calculus is a subsystem of both the Blockwise Calculus and the Circular Blockwise Calculus.

3.2 Encoding TFNP Problems as Multivalued CNFs

Given any of the total search problems introduced in Section 2, we can create a natural unsatisfiable multivalued CNF formula F expressing that the search problem has no solution. Intuitively, the negation of F encodes that the search problem is total.

Coloured Sink-of-Dag

We first show how to encode the Coloured Sink-of-Dag (C-SoD$_n$) problem. For each $i, j \in [n]$ we have a multivalued variable $(s_{ij}, n+1)$ expressing that the pointer of the node s_{ij} is either 0 or points to a node on the next level. For each $i, j \in [n]$ and each $\lambda \in [n]$ we have a multivalued variable $(c_{i,j,\lambda}, 2)$ expressing whether or not the colour λ is present at node (i, j). Finally, for each $i, j \in [n]$ we have a second multivalued variable (e_{ij}, n) indexing a colour at that node. We can now phrase the totality of the search problem using the following unsatisfiable multivalued CNF formula C-SoD, containing the following clauses:

- **Colourless Distinguished Source.** For each $\lambda \in [n-1]_0$, $[\![c_{11\lambda} \neq 1]\!]$.
- **Propagating Colours.** For each $i \in [n-1]$, each $j, k \in [n]$, and each $\lambda \in [n-1]_0$,

$$[\![s_{ij} \neq k]\!] \vee [\![c_{i+1,k,\lambda} \neq 1]\!] \vee [\![c_{i,j,\lambda} \neq 0]\!] .$$

- **Coloured Sinks.** For each $i \in [n-1], j \in [n], \lambda \in [n-1]_0$,

$$[\![s_{ij} \neq 0]\!] \vee [\![e_{ij} \neq \lambda]\!] \vee [\![c_{ij\lambda} \neq 0]\!] , \text{ and}$$

$$[\![e_{nj} \neq \lambda]\!] \vee [\![c_{nj\lambda} \neq 0]\!] .$$

Coloured Sink-of-Line and Coloured End-of-Line

The variables of both C-SoL$_n$ and C-EoL$_n$ are the same, but the two formulas differ on their defining constraints. For each $u, \lambda \in [n-1]_0$ we have multivalued variables $(s_u, n), (p_u, n), (e_u, n)$, and $(c_{u,\lambda}, 2)$ encoding successor pointers, predecessor pointers, colour pointers, and colours for each node. The nodes range in the set $[n-1]_0$ and we treat 0 as the distinguished source node. The clauses of the C-SoL$_n$ formula are defined as follows:

- **Colourless Distinguished Source.** For each $\lambda \in [n-1]_0$, $[\![c_{0,\lambda} \neq 1]\!]$.
- **Colour Propagation.** For each $u \neq v \in [n-1]_0$ and each $\lambda \in [n]$,

$$[\![s_u \neq v]\!] \vee [\![p_v \neq u]\!] \vee [\![c_{u,\lambda} \neq 0]\!] \vee [\![c_{v,\lambda} \neq 1]\!] .$$

- **Coloured Sinks.** For each $u, v, w, \lambda \in [n-1]_0$ with $u \neq w$:

$$[\![s_u \neq v]\!] \vee [\![p_v \neq w]\!] \vee [\![e_u \neq \lambda]\!] \vee [\![c_{u,\lambda} \neq 0]\!] .$$

The C-EoL$_n$ formula adds the following clauses to the C-SoL$_n$ formula:

- **Colourless Sources.** For each $u \in [n-1]$, $v, w, \lambda \in [n-1]_0$ with $u \neq w$:

$$[\![p_u \neq v]\!] \vee [\![s_v \neq w]\!] \vee [\![c_{u,\lambda} \neq 1]\!] .$$

Coloured Sink-of-Potential-Line and Coloured End-of-Potential-Line

The variables of C-SoPL$_n$ and C-EoPL$_n$ are the same. For each $i, j \in [n-1]_0$ and each $\lambda \in [n-1]_0$ we have variables $(s_{ij}, n), (p_{ij}, n), (e_{ij}, n), (c_{ij\lambda}, n)$ encoding successor pointers, predecessor pointers, colour pointers, and colours for each node. The clauses of the C-SoPL$_n$ formula are defined as follows:

- **Colourless Distinguished Source.** For each $\lambda \in [n-1]_0$, $[\![c_{0,0,\lambda} \neq 1]\!]$.
- **Colour Propagation.** For each $i \in [n-2]_0, j, k \in [n-1]_0$ and each $\lambda \in [n-1]_0$,

$$[\![s_{ij} \neq k]\!] \vee [\![p_{i+1,k} \neq j]\!] \vee [\![c_{i,j,\lambda} \neq 0]\!] \vee [\![c_{i+1,k,\lambda} \neq 1]\!] .$$

- **Coloured Sinks.** For each $i \in [n-2]_0, j, k, \ell \in [n-1]_0$ with $\ell \neq j$ and $\lambda \in [n-1]_0$

$$[\![s_{ij} \neq k]\!] \vee [\![p_{i+1,k} \neq \ell]\!] \vee [\![e_{i,j} \neq \lambda]\!] \vee [\![c_{i,j,\lambda} \neq 0]\!], \text{ and}$$

$$[\![e_{n-1,j} \neq \lambda]\!] \vee [\![c_{n-1,j,\lambda} \neq 0]\!].$$

The C-EoPL$_n$ formula adds the following clauses to the C-SoPL$_n$ formula:

- **Colourless Sources.** For each $i \in [n-1]$, $j, k, \ell, m \in [n-1]_0$ with $m \neq j$, and each $\lambda \in [n-1]_0$

$$[\![p_{ij} \neq \ell]\!] \vee [\![s_{i-1,\ell} \neq m]\!] \vee [\![s_{ij} \neq k]\!] \vee [\![p_{i+1,k} \neq j]\!] \vee [\![c_{ij\lambda} \neq 1]\!]$$

and

$$[\![s_{0,j} \neq k]\!] \vee [\![p_{1,k} \neq j]\!] \vee [\![c_{0,j,\lambda} \neq 1]\!].$$

3.3 Blockwise Calculus vs. Boolean Proof Systems

In this section we prove the main technical theorem necessary for our main results. Essentially, it says that if we have a refutation of a formula F in the Blockwise Calculus or one if its variations, then we can automatically obtain a refutation of any F-formulation in a related boolean proof system.

▶ **Theorem 3.6.** *Let F, G be any width-c multivalued CNF formulas for which there is a depth-d $S(F)$-formulation of $S(G)$. Then*

- *If there is a size-s Blockwise Calculus refutation of F, then there is a size-$s2^{O(d)}$ Res$(O(d))$-refutation of G_{bool}.*
- *If there is a size-s Circular Blockwise Calculus refutation of F, then there is a size-$s2^{O(d)}$ uCircRes$(O(d))$-refutation of G_{bool}.*
- *If there is a size-s Reversible Blockwise Calculus refutation of F, then there is a size-$s2^{O(d)}$ RevRes$(O(d))$-refutation of G_{bool}.*

Proving this theorem is much easier after we introduce some auxiliary technical lemmas for working with decision trees. For a given decision tree T, let $P_\ell(T)$ denote the set of all root-leaf paths ending in a leaf labelled by ℓ, and let $P(T) := \bigcup_\ell P_\ell(T)$. For a given path $p \in P(T)$ let $C_p := x_1 \wedge \cdots \wedge x_k$ and $\overline{C}_p := \overline{x}_1 \vee \cdots \vee \overline{x}_k$ where $x_1, ..., x_k$ are the queries made along p.

Let $D_T := \bigvee_{p \in P(T)} C_p$, intuitively encoding the fact that some branch of a decision tree must be followed under an input. We will rely on these formulas heavily, so we now demonstrate that they can be efficiently derived in Res(k). It is enough to prove this next lemma for RevRes(d) since both Res(d) and uCircRes(d) simulate RevRes(d).

▶ **Lemma 3.7.** *If T is a decision tree of depth d, then there is a size-2^{2d} RevRes(d) proof of the formula $\bigvee_{p \in P(T)} C_p$.*

Proof. If T consists of a single query of some literal ℓ, then $D_T = \ell \vee \overline{\ell}$, which can be derived in a single line as an axiom. Otherwise we proceed by induction, so let ℓ be the first literal queried by T. Let T_0 be the subtree of T followed when ℓ is falsified, and T_1 be the one followed when ℓ is satisfied. By induction, we can derive D_{T_0} and D_{T_1} with size $2 \cdot 2^{2(d-1)} = 2^{2d-1}$. We begin by $\vee$-weakening T_0 with ℓ and T_1 with $\overline{\ell}$, and introducing the axiom $\ell \vee \overline{\ell}$.

Now let $p_1, ..., p_k$ be the paths of T_0. Weaken the axiom $\ell \vee \overline{\ell}$ by C_{p_i} for all $1 < i \leq k$ to obtain $\bigvee_{1<i\leq k} C_{p_i} \vee \ell \vee \overline{\ell}$. $\wedge$-introducing this with $D_{T_0} \vee \overline{\ell}$ on C_{p_1} and $\overline{\ell}$, we obtain:

$$\bigvee_{1<i\leq k} C_{p_i} \vee C_{p_1 \cup \overline{\ell}} \vee \ell$$

Now weaken $\ell \vee \overline{\ell}$ by C_{p_i} for all $2 < i \leq k$, and by $C_{p_i \cup \overline{\ell}}$ for $1 \leq i < 2$ to obtain $\bigvee_{2<i\leq k} C_{p_i} \vee \bigvee_{1\leq i<2} C_{p_i \cup \overline{\ell}} \vee \ell \vee \overline{\ell}$. We again $\wedge$-introduce this, this time with $\bigvee_{1<i\leq k} C_{p_i} \vee C_{p_i \cup \overline{\ell}} \vee \ell$ on C_{p_2} and $\overline{\ell}$ to obtain:

$$\bigvee_{2<i\leq k} C_{p_i} \vee \bigvee_{1\leq i\leq 2} C_{p_i \cup \overline{\ell}} \vee \ell$$

Repeating this for the remaining paths p_j for $2 < j \leq k$, we obtain:

$$\bigvee_{p\in P(T_0)} C_{p \cup \overline{\ell}} \vee \ell$$

and we can repeat this process for T_1 to likewise derive:

$$\bigvee_{p\in P(T_1)} C_{p \cup \ell} \vee \overline{\ell}$$

Since $P(T) = \bigcup_{p\in P(T_0)}(p \cup \overline{\ell}) \cup \bigcup_{p\in P(T_1)}(p \cup \ell)$, we can finally cut these two formulas on ℓ to obtain $\bigvee_{p\in P(T)} C_p = D_T$. All conjunctions created in this process have width at most d, as they each correspond to a path or subpath of a path of T, and since there are 2^{d-1} paths in each subtree this process adds an additional $2 \cdot (2^{d-1})^2 = 2^{2d-1}$ lines to the proof. Thus the total size of the proof is $2 \cdot 2^{2d-1} = 2^{2d}$. Further, since all root–leaf paths are bounded in length by d, the proof has width $O(d)$. ◀

We now show that cutting and weakening along negated paths of decision trees can be done inside of Reversible Resolution.

▶ **Lemma 3.8.** *Let C be a width-w clause and let T be a depth-d decision tree. Then there is a size-2^d, width-$(w+d)$* RevRes *derivation of C from the set of clauses $\{C \vee \overline{C}_p \mid p \in P(T)\}$ and* vice-versa*.*

Proof. We proceed by induction on d. In the base case, $d = 1$ and a single variable x is queried by T; in this case we have the formulas $C \vee x$ and $C \vee \overline{x}$ and we resolve on x to obtain C.

By induction suppose that the claim holds for a decision tree of depth at most $d-1$. Let T_0 be the decision tree obtained by discarding all leaves of T (the new leaves may be labelled arbitrarily). For each path $p \in P(T)$, let x be the final variable it queries, let q be the path of T which differs from p only on x, and let p_0 be the path of T_0 obtained by truncating p before x. Then we may cut the formulas $C \vee \overline{p}_0 \vee x$ and $C \vee \overline{p}_0 \vee \overline{x}$, corresponding to p and q, on x to obtain $C \vee \overline{p}_0$. Repeating this for each such pair of paths in T yields $C \vee \overline{p}_0$ for each $p_0 \in T_0$ in 2^{d-1} steps, allowing us to apply the induction hypothesis to complete the derivation of C in a further 2^{d-1} steps, for a total size of 2^d as desired. Furthermore, this is reversible, as each path of T belongs to a single such pair. The width claims are also clear as all formulas are of the form $C \vee \overline{p}$ for some path p of a depth-d decision tree. ◀

Now, let F be a multivalued CNF formula on variables (x_i, r_i) for $i \in [n]$, let G be a multivalued CNF formula on variables (y_i, s_i) for $i \in [m]$, and suppose that we have a depth-d $S(F)$-formulation of $S(G)$. This means that each variable x_i is computed by a depth-d decision tree f_i which queries variables y_j and outputs a value in $[r_i - 1]_0$, and we also have, for each clause C in F, a decision tree g_C which queries y_j variables and outputs a clause of $S(G)$. For simplicity, we will identify the variable x_i with its decision tree f_i that computes it.

Suppose that we have a Blockwise Calculus refutation Π of F. Our goal is to give a Res($O(d)$) refutation of G. In order to prove this theorem we need to encode atoms $[\![x_i \neq j]\!]$ into boolean formulas. We introduce two such encodings: the *positive* and *negative* encoding. In the positive encoding we encode each atom as a d-DNF formula, while in the negative encoding we encode the atom as a family of width-d clauses. We emphasize that in the definitions below we identify the variable x_i of the formula F with the decision tree f_i outputting the value of x_i in the reduction from G.

$$\mathcal{D}^+([\![x_i \neq j]\!]) := \bigvee_{k \neq j} \bigvee_{p \in P_k(x_i)} C_p$$
$$\mathcal{D}^-([\![x_i \neq j]\!]) := \{\overline{C}_p : p \in P_j(x_i)\}$$

If C is a clause over multivalued atoms we write $\mathcal{D}^+(C)$ to denote the DNF formula obtained by substituting each atom A in C with its positive encoding $\mathcal{D}^+(A)$, and write $\mathcal{D}^-(C)$ to denote the CNF formula obtained by substituting $\bigwedge \mathcal{D}^-(A)$ for each atom in C and then re-writing the result in CNF by distributing the $\vee$ over the $\wedge$s.

The next lemma is arguably the main technical lemma used in the proof of Theorem 3.6. It shows that it is possible to derive positive encodings of multivalued clauses from negative encodings and vice-versa efficiently in RevRes(d).

▶ **Lemma 3.9.** *Suppose that x is computed by a depth-d decision tree and G is a DNF. Then there is a* RevRes(d) *proof of all the DNFs in $\{G \vee C \mid C \in \mathcal{D}^-([\![x \neq j]\!])\}$ from $G \vee \mathcal{D}^+([\![x \neq j]\!])$ and vice-versa in size $|G|^2 \cdot 2^{O(d)}$*

Proof. We begin by proving $G \vee \mathcal{D}^+([\![x \neq j]\!])$ from $\{G \vee C \mid C \in \mathcal{D}^-([\![x \neq j]\!])\}$. This direction is simpler. By applying Lemma 3.7 we can derive the DNF $\bigvee_{p \in P(x)} C_p$ in size 2^{2d} from axioms, and then by applying reverse cut repeatedly we can derive $G \vee \bigvee_{p \in P(x)} C_p$ in size $O(|G|^2 2^d)$. From $G \vee \bigvee_{p \in P(x)} C_p$ we can repeatedly cut on $G \vee C$ for each $C \in \mathcal{D}^-([\![x \neq j]\!])$ to in sequence to derive $G \vee \mathcal{D}^+([\![x \neq j]\!])$. The total size is $|G|^2 2^{O(d)}$.

We now prove the other direction. Without loss of generality suppose that $j = 0$ and let $\mathcal{D} := G \vee \mathcal{D}^+([\![x \neq 0]\!])$ for the sake of brevity. By definition we have $\mathcal{D} = G \vee \bigvee_{k \neq j} \bigvee_{p \in P_k(x)} C_p$. Let $\mathcal{P} = \bigcup_{k \neq 0} P_k(x)$ denote the set of all paths appearing in the above disjunction and write $\mathcal{P} = \{p_1, p_2, \ldots, p_s\}$.

We begin by applying reverse cut repeatedly along the variables in the decision tree computing x to derive the set of DNFs $\{\mathcal{D} \vee \overline{C}_q \mid q \in P(x)\}$. Fix an arbitrary path $q \in P_0(x)$. For each path $p_i \in \mathcal{P}$ there is a literal ℓ_i such that ℓ_i is queried positively in p_i and negatively in q. Therefore, by using an axiom-introduction we can introduce the clause $\ell_1 \vee \overline{\ell}_1$ and then repeatedly using reverse-cut we can derive $G \vee \bigvee_{i=2}^{s} C_{p_i} \vee \overline{C}_q \vee \overline{C}_{p_1}$ We can then cut this result with $\mathcal{D} \vee \overline{C}_q$ to derive $G \vee \bigvee_{i=2}^{s} C_{p_i} \vee \overline{C}_q$. We can now repeat this process: there is another literal ℓ_2 appearing positively in p_2 and negatively in q, and thus we can axiom-introduce $\ell_2 \vee \overline{\ell}_2$ and then use reverse cut to derive $G \vee \bigvee_{i=3}^{s} C_{p_i} \vee \overline{C}_{p_2} \vee \overline{C}_q$. Cutting this with the result of the previous stage yields $G \vee \bigvee_{i=3}^{s} C_{p_i} \vee \overline{C}_q$, and we can repeat this process s times in order to derive $G \vee \overline{C}_q$. We can then repeat this for each $q \in P_0(x)$ to derive $\mathcal{D}^-([\![x_i \neq j]\!])$.

We now estimate the size of the derivation. The first line has size at most $|G| + 2^d$, and we begin by deriving a set of 2^d DNFs, each of size $O(|G| + 2^d)$, and thus the cost of the first step is $O(|G|2^{2d})$. To cut each of the paths $C_{p_1}, C_{p_2}, \ldots, C_{p_s}$ we must pay $O(|G|^2 2^d)$ to derive the corresponding DNF to cut our preserved formula with, and this will repeat $s \leq 2^d$ times, for a total cost of $O(|G|^2 2^{2d})$. Finally, we must repeat this entire process $\leq 2^d$ times for each $q \in P_0(x)$, and thus the final size is $O(|G|^2 2^{3d}) = |G|^2 2^{O(d)}$. ◀

Using the lemma we can now prove Theorem 3.6.

Proof of Theorem 3.6. The basic idea of this proof is simple: for each clause $C \in \Pi$ we replace C with the width-d DNF encoding $\mathcal{D}^+(C)$, noting that the final clause $\bot$ remains empty. We prove two claims:

Claim 1. For each clause C in F we can deduce $\mathcal{D}^+(C)$ from the clauses of G_{bool} in RevRes($O(d)$).

Claim 2. For each proof rule of the Blockwise Calculus we can deduce the positive encodings of each consequent of the rule from the positive encodings of each antecedent of the rule efficiently in RevRes($O(d)$).

To prove the first claim, let $F = C_1 \wedge \cdots \wedge C_s$ and $G_{bool} = C'_1 \wedge \cdots \wedge C'_t$, let (x_i, r_i) for $i \in [n]$ denote the variables of F, and let $\vec{y}_1, \ldots, \vec{y}_m$ denote the (boolean block) variables of G_{bool}. By the definition of an $S(F)$-formulation, for each variable (x_i, r_i) of F we have a depth-d decision tree f_i querying variables of G_{bool} and outputting a value for x_i, as well as a decision tree g_k for each $k \in [s]$ such that $(f(y), k) \in S(F) \Rightarrow (y, g_k(y)) \in S(G)$. We can interpret this definition in terms of proofs as follows. Let $C_k = A_1 \vee \cdots \vee A_w$ be any clause of F and assume w.l.o.g. that $A_i := [\![x_i \neq \ell_i]\!]$ for some ℓ_i. For each $i \in [w]$ let $p_i \in P_{\ell_i}(x_i)$ be any path in the corresponding decision tree from the formulation outputting ℓ_i, and let $q \in P(g_k)$ be any path in the tree g_k. Then the clause $\bigvee_{i=1}^w \overline{C}_{p_i} \vee \overline{C}_q$ is a weakening of clause of G. Since there are at most 2^d paths in each decision tree and the width of C_k is w it follows that there are at most $2^{wd} \leq 2^{cd}$ such clauses, and each can be deduced from clauses of G using weakening rules. Next, we observe that from the collection of clauses $\{\bigvee_{i=1}^w \overline{C}_{p_i} \vee \overline{C}_q \mid q \in P(g_k)\}$ we can use reversible cuts up the decision tree g_k in order to deduce the family of clauses $\{\bigvee_{i=1}^w \overline{C}_{p_i}\}$, and taking the union over all such paths $p_i \in \mathcal{P}_{\ell_i}(x_i)$ yields exactly $\mathcal{D}^-(C_k)$. Finally, applying Lemma 3.9 yields $\mathcal{D}^+(C_k)$. Applying this strategy to all clauses of F we can deduce $\mathcal{D}^+(C_k)$ for each clause of F, as desired.

We move on to proving the second claim. We first consider the Cut rule

$$\frac{C \vee [\![x_i \neq 0]\!] \;\; C \vee [\![x_i \neq 1]\!] \;\; \cdots \;\; C \vee [\![x_i \neq r_i - 1]\!]}{C}$$

for which we need to show how to derive $\mathcal{D}^+(C)$ from $\mathcal{D}^+(C) \vee \mathcal{D}^+([\![x_i]\!] \neq \ell)$ for each $\ell \in [r_i - 1]_0$. We can apply Lemma 3.9 to $\mathcal{D}^+(C) \vee \mathcal{D}^+([\![x_i \neq \ell]\!])$ for each $\ell = 0, 1, \ldots, r_i - 1$ in order to derive the family

$$\bigcup_{\ell=0}^{r_i-1} \{\mathcal{D}^+(C) \vee D \mid D \in \mathcal{D}^-([\![x \neq \ell]\!])\} = \bigcup_{p \in P(x_i)} \{\mathcal{D}^+(C) \vee \overline{C}_p\}.$$

From this family we can apply Lemma 3.8 in order to derive $\mathcal{D}^+(C)$, as desired.

We now consider the Reverse Cut rule

$$\frac{C}{C \vee [\![x_i \neq 0]\!] \quad C \vee [\![x_i \neq 1]\!] \quad \cdots \quad C \vee [\![x_i \neq r_i - 1]\!].}$$

Starting from $\mathcal{D}^+(C)$ we must derive the family $\{\mathcal{D}^+(C) \vee \mathcal{D}^+([\![x_i \neq \ell]\!]) \mid \ell \in [r_i - 1]_0\}$. This direction is easy: since this rule is the reverse of the previous rule, and since we gave a RevRes(d) simulation of the previous rule, running the previous construction in reverse handles this case as well.

Using the two claims we can now complete the proof of the theorem. For Res(d) and RevRes(d) the result follows immediately by induction over the proof Π. For Circular Res(d) we can similarly apply induction over Π, additionally observing that if we ever apply the Clause Creation rule in a Circular Blockwise Calculus proof to create a clause C, we can simply apply the DNF creation rule in Circular Res(d) to create $\mathcal{D}^+(C)$. Since the Circular Blockwise Calculus proof must derive each clause C more times than it is introduced by a Clause Creation rule, the same property holds for the uCircRes(d) proof. This completes the proof of the theorem. ◀

A similar result can also be obtained for low-width Resolution, which we will use to show collapses to uncoloured classes. The main difference in this proof is that we do not use the positive encoding, only the negative encoding.

▶ **Theorem 3.10.** *Let F, G be any multivalued CNF formulas for which there is a depth-d $S(F)$-formulation of $S(G)$. Then*

- *If there is a size-s, width-$\log^{O(1)} s$ Blockwise Calculus refutation of F, then there is a size-$s^{O(1)}2^{O(d)}$, width-$d \cdot \log^{O(1)} s$ Resolution refutation of G_{bool}.*
- *If there is a size-s, width-$\log^{O(1)} s$ Circular Blockwise Calculus refutation of F, then there is a size-$s^{O(1)}2^{O(d)}$, width-$d \cdot \log^{O(1)} s$ uCircRes-refutation of G_{bool}.*
- *If there is a size-s, width-$\log^{O(1)} s$ Reversible Blockwise Calculus refutation of F, then there is a size-$s^{O(1)}2^{O(d)}$, width-$d \cdot \log^{O(1)} s$ RevRes-refutation of G_{bool}.*

Proof. Let Π be the size-s, width-$\log^{O(1)} s$ Blockwise Calculus refutation (potentially reversible or circular) of F. We construct a Resolution refutation of G from Π By first proving $\mathcal{D}^-(F) := \bigwedge_{C \in F} \mathcal{D}^-(C)$, then converting Π into a refutation of $\mathcal{D}^-(F)$, so we may finally combine these proofs into a refutation of G. Again, this requires us to show the following:

- For each clause C of F, there is an efficient proof of $\mathcal{D}^-(C)$ from G
- Each rule of the blockwise calculus can be efficiently simulated by Reversible Resolution using the negative encoding of blocks

Let $C_1, ..., C_s$ denote the clauses of F, and $C'_1, ..., C'_t$ the clauses of G_{bool}. Likewise, dnote the variables of F by $(x_1, r_1), ..., (x_n, r_n)$ and the variables of G_{bool} by $\vec{y}_1, ..., \vec{y}_m$. For each variable x_i of F, we have a depth-d decision tree f_i decision tree in the formulation over variables of G_{bool} computing it, and for each clause C_i of F, we have a depth-d decision tree g_i outputting a corresponding clause of G. By definition of a $S(F)$-formulation then, for each such clause C_i, each clause $C' \in \mathcal{D}^-(C_i)$, and each path $p \in P(g_i)$, the clause $C' \vee \overline{C}_p$ is a weakening of at least one clause of G – if C' and C_p are both falsified under some assignment $\vec{a}$ to the variables of G, then so too must the clause output by p be falsified under $\vec{a}$. Thus, for each such C', we can derive the clause $C' \vee \overline{C}_p$ from G for every $p \in P(g_i)$, at which point we may apply Lemma 3.8 to obtain $\mathcal{D}^-(C_i)$. Repeating for all clauses of F yields $\mathcal{D}^-(F)$.

We proceed now to show that we can simulate the rules of the blockwise calculus in Reversible Resolution:

- If some clause C was derived by cutting earlier clauses $C \vee [\![x \neq 0]\!], ..., C \vee [\![x \neq n-1]\!]$, then we have the set of clauses $C' \vee \overline{p}$ for each $C' \in \mathcal{C}$ and $p \in P(T_x)$, from which we wish to derive each $C' \in \mathcal{C}$. Thus, by Lemma 3.8 this can be done in 2^d steps with width $d + d \cdot \log^{O(1)} s$.

- If C was derived by weakening some earlier clause C_0 on some variable x, then begin with each $C' \in \mathcal{C}$, from which we wish to derive $C' \vee \overline{p}$ for each $C' \in \mathcal{C}$ and $p \in P(T_x)$. By reversibility of Lemma 3.8, this can be done in 2^d steps with width $d + d \cdot \log^{O(1)} s$.

Since all families of clauses $\mathcal{C}$ corresponding to an original clause C of Π have size $s^{O(1)}2^{O(d)}$ and each new clause requires 2^d additional steps to derive, this results in a proof of size $n^{O(1)}2^{O(d)}$ overall. Furthermore, all clauses in the new proof consist of $\log^{O(1)} s$ negated paths of depth-d decision trees, and thus have width $d \cdot \log^{O(1)} s$ overall. ◀

References

1 Albert Atserias and Maria Luisa Bonet. On the automatizability of resolution and related propositional proof systems. *Inf. Comput.*, 189(2):182–201, 2004. doi:10.1016/j.ic.2003.10.004.

2 Albert Atserias and Massimo Lauria. Circular (yet sound) proofs. In *Proceedings of the 22nd Theory and Applications of Satisfiability Testing (SAT)*, pages 1–18. Springer, 2019. doi:10.1007/978-3-030-24258-9_1.

3 Paul Beame, Stephen Cook, Jeff Edmonds, Russell Impagliazzo, and Toniann Pitassi. The relative complexity of NP search problems. *Journal of Computer and System Sciences*, 57(1):3–19, 1998. doi:10.1006/jcss.1998.1575.

4 Paul Beame, Russell Impagliazzo, Jan Krajíček, Toniann Pitassi, and Pavel Pudlák. Lower bounds on Hilbert's Nullstellensatz and propositional proofs. In *Proceedings of the 35th Symposium on Foundations of Computer Science (FOCS)*, pages 794–806, 1994. doi:10.1109/SFCS.1994.365714.

5 Arnold Beckmann and Samuel R. Buss. Characterising definable search problems in bounded arithmetic via proof notations. In *Ways of Proof Theory*, ONTOS Series in Mathematical Logic, pages 65–134, 2010.

6 Ilario Bonacina and Maria Luisa Bonet. On the strength of sherali-adams and nullstellensatz as propositional proof systems. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 25:1–25:12. ACM, 2022. doi:10.1145/3531130.3533344.

7 María Luisa Bonet, Jordi Levy, and Felip Manyà. Resolution for Max-SAT. *Artificial Intelligence*, 171(8-9):606–618, 2007. doi:10.1016/j.artint.2007.03.001.

8 Joshua Buresh-Oppenheim and Tsuyoshi Morioka. Relativized NP search problems and propositional proof systems. In *Proceedings of the 19th IEEE Conference on Computational Complexity (CCC)*, pages 54–67, 2004. doi:10.1109/CCC.2004.1313795.

9 Sam Buss, Noah Fleming, and Russell Impagliazzo. Tfnp characterizations of proof systems and monotone circuits. *Electron. Colloquium Comput. Complex.*, TR22-141, 2022. arXiv:TR22-141.

10 Samuel R. Buss and Alan S. Johnson. Propositional proofs and reductions between NP search problems. *Annals of Pure and Applied Logic*, 163(9):1163–1182, 2012. doi:10.1016/j.apal.2012.01.015.

11 Xi Chen, Decheng Dai, Ye Du, and Shang-Hua Teng. Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *Proceedings of the 50th Symposium on Foundations of Computer Science (FOCS)*, pages 273–282, 2009. doi:10.1109/FOCS.2009.29.

12 Xi Chen, Dimitris Paparas, and Mihalis Yannakakis. The complexity of non-monotone markets. *Journal of the ACM*, 64(3):20:1–20:56, 2017. doi:10.1145/3064810.

13 Bruno Codenotti, Amin Saberi, Kasturi Varadarajan, and Yinyu Ye. The complexity of equilibria: Hardness results for economies via a correspondence with games. *Theoretical Computer Science*, 408(2–3):188–198, 2008. doi:10.1016/j.tcs.2008.08.007.

14 Stefan S. Dantchev, Barnaby Martin, and Mark Nicholas Charles Rhodes. Tight rank lower bounds for the sherali-adams proof system. *Theor. Comput. Sci.*, 410(21-23):2054–2063, 2009. doi:10.1016/j.tcs.2009.01.002.

15 Constantinos Daskalakis, Paul Goldberg, and Christos Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009. doi:10.1137/070699652.

16 John Fearnley, Paul W. Goldberg, Alexandros Hollender, and Rahul Savani. The complexity of gradient descent: CLS = PPAD ∩ PLS. In *Proceedings of the 53rd Symposium on Theory of Computing (STOC)*, pages 46–59, 2021. doi:10.1145/3406325.3451052.

17 Yuval Filmus, Meena Mahajan, Gaurav Sood, and Marc Vinyals. MaxSAT resolution and subcube sums. In *Proceedings of the 23rd Theory and Applications of Satisfiability Testing (SAT)*, pages 295–311. Springer, 2020. doi:10.1007/978-3-030-51825-7_21.

18 Michal Garlík. Failure of feasible disjunction property for k-dnf resolution and np-hardness of automating it. *CoRR*, abs/2003.10230, 2020. arXiv:2003.10230.

19 Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. Further collapses in TFNP. In *Proceedings of the 37th Computational Complexity Conference (CCC)*, pages 33:1–33:15, 2022. doi:10.4230/LIPICS.CCC.2022.33.

20 Mika Göös, Alexandros Hollender, Siddhartha Jain, Gilbert Maystre, William Pires, Robert Robere, and Ran Tao. Separations in proof complexity and TFNP. *Electron. Colloquium Comput. Complex.*, TR22-058, 2022. arXiv:TR22-058.

21 Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in monotone complexity and TFNP. In *Proceedings of the 10th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 124, pages 38:1–38:19, 2018. doi:10.4230/LIPIcs.ITCS.2019.38.

22 David Johnson, Christos Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988. doi:10.1016/0022-0000(88)90046-3.

23 Jan Krajíček. On the weak pigeonhole principle. *Fundamenta Mathematicae*, 170(1-3):123–140, 2001.

24 Jan Krajícek, Alan Skelley, and Neil Thapen. NP search problems in low fragments of bounded arithmetic. *J. Symb. Log.*, 72(2):649–672, 2007. doi:10.2178/jsl/1185803628.

25 Javier Larrosa, Federico Heras, and Simon de Givry. A logical approach to efficient Max-SAT solving. *Artificial Intelligence*, 172(2-3):204–233, 2008. doi:10.1016/j.artint.2007.05.006.

26 Nimrod Megiddo and Christos Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2):317–324, 1991. doi:10.1016/0304-3975(91)90200-L.

27 Christos Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994. doi:10.1016/s0022-0000(05)80063-7.

28 Nathan Segerlind, Samuel R. Buss, and Russell Impagliazzo. A switching lemma for small restrictions and lower bounds for k-dnf resolution. *SIAM J. Comput.*, 33(5):1171–1200, 2004. doi:10.1137/S0097539703428555.

29 Hanif Sherali and Warren Adams. A hierarchy of relaxations and convex hull characterizations for mixed-integer zero–one programming problems. *Discrete Applied Mathematics*, 52(1):83–106, July 1994. doi:10.1016/0166-218x(92)00190-w.

30 Alan Skelley and Neil Thapen. The provably total search problems of bounded arithmetic. *Proceedings of the London Mathematical Society*, 103(1):106–138, 2011.

31 Neil Thapen. A tradeoff between length and width in resolution. *Theory Comput.*, 12(1):1–14, 2016. doi:10.4086/toc.2016.v012a005.